A Computationally Efficient Symmetric Diagonally Dominant Matrix Projection-based Gaussian Process Approach

Peng Wang[a,*], Lyudmila Mihaylova[a], Said Munir[b], Rohit Chakraborty[b], Jikai Wang[c], Martin Mayfield[b], Khan Alam[d], Muhammad Fahim Khokhar[e], and Daniel Coca[a]

[a] *Dept. of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield S10 3JD, UK*
[b] *Dept. of Civil and Structural Engineering, The University of Sheffield, Sheffield S10 3JD, UK*
[c] *Dept. of Automation, University of Science and Technology of China, Hefei 230027, P.R.China*
[d] *Department of Physics, University of Peshawar, KPK 25120, Pakistan*
[e] *Institute of Environmental Sciences and Engineering, National University of Sciences and Technology, Islamabad 44000, Pakistan*

Abstract

Although kernel approximation methods have been widely applied to mitigate the $\mathcal{O}(n^3)$ cost of the $n \times n$ kernel matrix inverse in Gaussian process methods, they still face computational challenges. The 'residual' matrix between the covariance and the approximating component is often discarded as it prevents the computational cost reduction. In this paper, we propose a computationally efficient Gaussian process approach that achieves better computational efficiency, $\mathcal{O}(mn^2)$, compared with standard Gaussian process methods, when using $m \ll n$ data. The proposed approach incorporates the 'residual' matrix in its symmetric diagonally dominant form which can be further approximated by the Neumann series. We have validated and compared the approach with full Gaussian process approaches and kernel approximation based Gaussian process

*Corresponding author
Email addresses: Peng.Wang@Sheffield.ac.uk (Peng Wang), l.s.mihaylova@Sheffield.ac.uk (Lyudmila Mihaylova), smunir2@sheffield.ac.uk (Said Munir), rohit.chakraborty@sheffield.ac.uk (Rohit Chakraborty), wangjk@ustc.edu.cn (Jikai Wang), martin.mayfield@sheffield.ac.uk (Martin Mayfield), khanalam@uop.edu.pk (Khan Alam), fahim.khokhar@iese.nust.edu.pk (Muhammad Fahim Khokhar), d.coca@sheffield.ac.uk (and Daniel Coca)

variants, both on synthetic and real air quality data.

Keywords: `Gaussian process methods, Symmetric diagonally dominant projection`, `Kernel approximation`, `Sustainable development`, `Air quality forecasting`

1. Introduction

Gaussian process (GP) methods are renowned for providing Bayesian non-linear and non-parametric solutions to regression and classification tasks [1, 2]. However, they have cubic computational complexity $\mathcal{O}(n^3)$ for the inversion of the kernel matrix of size $n \times n$ and its determinant [3]. This cubic computational cost has effectively limited applications of GPs to data with thousands of samples [4, 5]. This led to intensive studies of the scalability of GPs during the last decades [3], with particular interests in adapting GPs for various data processing and maintaining their capacity, ideally at the same level of full GPs.

The extensive review of Liu et al. [3] on GPs classifies scalable GPs into local approximations and global approximations. Local approximations follow the *divide-and-conquer* idea to first divide the whole dataset into sub-datasets, each with m samples. A 'local' GP model is next trained on each sub-dataset. These 'local' GP models are aggregated with each GP model responds to inputs that come from a certain 'local' area at the prediction stage. The computational cost is in the order of $\mathcal{O}(m^2 n)$ [3]. In the global GP algorithms, all data is available but sparsity is usually introduced by appropriately selected inducing points or by sub-sampling of the data. Thus for efficiency improvement, a lot of global approximation-based works [4, 6, 7] are dedicated to approximating the kernel matrix $\mathbf{K}_{nn}$ and can be further categorised into 1) the Subset of Data (SoD) method uses m out of n training samples, resulting in a smaller kernel matrix $\mathbf{K}_{mm}$; 2) the sparse kernel method sets to zero all entries smaller than a criterion value and hence leads to a sparse kernel matrix $\widetilde{\mathbf{K}}_{nn}$ [3]; 3) the low-rank method, also known as sparse approximation method, approximates $\mathbf{K}_{nn}$ with the eigendecompostion or the Nyström method [8] by a low-rank matrix $\mathbf{L}_{nn}$.

We also notice that Zhu et al. [9] apply the nonnegative matrix factorisation in the kernel matrix approximation, which obtains online performance with application in image processing. The SoD and low-rank approximations reduce the computational complexity to $\mathcal{O}(m^2 n)$ if we do not count the computation caused by eigendecomposition. The sparse kernel method reduces the cost to $\mathcal{O}(\alpha n^3)$, with α being a coefficient in the range $0 < \alpha < 1$.

The third solution is the most popular among all these three global approximations. Especially, the Nyström method [8] has achieved a balance between accuracy and efficiency and several variants have been proposed [2, 4]. We notice that these methods replace the kernel matrix $\mathbf{K}_{nn}$ with a low rank matrix $\mathbf{L}_{nn}$, which can be further factorised as $\mathbf{L}_{nn} = \mathbf{B}_{nm}\mathbf{B}_{nm}^{\mathrm{T}}$ [10]. The 'residual' matrix $\widetilde{\mathbf{A}} = \mathbf{K}_{nn} - \mathbf{L}_{nn}$ is usually discarded. This is intuitive from the Sherman-Morrison-Woodbury formula [11] (equation (2.7.12) in Chapter 2) perspective

$$\mathbf{K}_{nn}^{-1} = \widetilde{\mathbf{A}}^{-1} - \widetilde{\mathbf{A}}^{-1}\mathbf{B}_{nm}(\mathbf{I}_m + \mathbf{B}_{nm}^{\mathrm{T}}\widetilde{\mathbf{A}}^{-1}\mathbf{B}_{nm})^{-1}\mathbf{B}_{nm}^{\mathrm{T}}\widetilde{\mathbf{A}}^{-1}. \tag{1}$$

We can see that the computational cost of $\mathbf{K}_{nn}^{-1}$ remains at $\mathcal{O}(n^3)$ despite $(\mathbf{I}_m + \mathbf{B}_{nm}^{\mathrm{T}}\widetilde{\mathbf{A}}^{-1}\mathbf{B}_{nm})^{-1}$ holds the promise of reducing the overall computational cost of (1) such as when $\widetilde{\mathbf{A}}$ is diagonal. The consequence is that two questions remain unanswered: 1) Can we take the 'residual' matrix $\widetilde{\mathbf{A}}$ into consideration? 2) If we consider the 'residual' matrix, can we achieve comparable results with full GP models with a lower computational cost?

This paper aims to provide answers to these two questions. We show that by projecting $\widetilde{\mathbf{A}}$ to be a Symmetric Diagonally Dominant (SDD) matrix $\mathbf{A}$, we obtain an approximation of the kernel matrix $\mathbf{K}_{nn} \approx \mathbf{L}_{nn} + \mathbf{A}$ accurately and efficiently. The SDD matrix $\mathbf{A}$ shows appealing characteristics such as symmetry and diagonally dominant, but the challenge in (1) remains. However, the SDD matrix characteristics perfectly match the conditions of approximating its inverse matrix with Neumann series, hence avoiding calculating $\widetilde{\mathbf{A}}^{-1}$ directly. We therefore approximate $\widetilde{\mathbf{A}}^{-1}$ with Neumann series and cut the computational cost from $\mathcal{O}(n^3)$ down to $\mathcal{O}(n^2)$ [12, 13]. By doing so, we provide a way of com-

puting $\mathbf{K}_{nn}^{-1}$ and derive a new efficient GP implementation. To summarise, the main contributions of this paper are threefold:

1) The 'residual' matrix between the original kernel matrix and its approximation matrix is taken into account to approach full GPs performance;

2) The SDD projection and Neumann series are applied to reduce covariance matrix inversion computational complexity;

3) Comparisons of the proposed approach with various GP variants with different kernel settings on both synthetic data and air quality data are given, providing a reference for its potential applications in various fields.

The remaining part of this paper is organised as follows. Section 2 reviews some of the related works. Section 3 introduces the GP method and how traditional kernel approximation methods work. Section 4 details the proposed approach. Performance validation results and analysis are given in Section 5, and the paper is concluded in Section 6, with a brief discussion of future work.

2. Related Work

In literature, there are mainly three categories of low-rank approximations, i.e. the prior approximation, the posterior approximation, and the structured sparse approximation [3, 8, 14]. We are particularly interested in the prior approximation in this paper.

There are two main ways to approximate the prior kernel matrix. The first one is as we mentioned earlier, by applying an eigendecomposition or the Nyström method to reconstruct the kernel matrix with a low rank [15, 16, 17]. The second way is by applying variational inference to optimise the Kullback-Leibler divergence between the exact prior and a cluster of distributions that are easy to implement, such as Gaussian distributions [4, 18]. While the Nyström method can reduce the computational complexity, it faces the problem of how to determine the low-rank space such that the reconstructed kernel matrix would provide accurate results. This has stimulated research in two aspects: 1) low-rank approximation error analysis or finding the error bounds [15, 16, 19]; 2)

methods that can further improve the approximation accuracy such as greedy approaches and randomised algorithms [20, 21]. Particularly, Stein [17] reports that when neighboring observations are strongly correlated, the performance of the low-rank approximation becomes poor even if the sum of the m largest eigenvalues is much greater than the sum of the remaining eigenvalues. Ding et al. [22] find that bad performance would happen when the length scale of the kernel is small if we only focus on the high eigenvalue part of the spectrum of the kernel matrix. A multi-resolution kernel approximation approach is thus proposed which represents the entire kernel matrix, not just its eigenvectors with the greatest eigenvalues. This approach is capable of calculating the inverse matrix directly and has improved performance compared with other GP kernel approximations considered in [22]. They claim their method is considerably more flexible than existing hierarchical matrix decomposition or approximation [23] methods. Yao et al. propose the kernel-band-projection algorithm for anomaly detection in hyperspectral imagery. They take bands as mapping subjects, and by mapping bands into the kernel space, they construct a projection matrix. When the Gaussian kernel function is used, their method avoids the inversion calculation of the projection matrix, making the method efficient [24]. Burt et al. [19] adopt the Kullback-Leibler divergence to investigate how the number of inducing points m that could change along with the change of the dataset size n, to ensure a certain approximation accuracy. However, this work falls into the posterior approximation category [25].

While the approximation methods and approximation accuracy analysis have been extensively researched, there is little work on how to retain the 'residual' matrix $\widetilde{\mathbf{A}}$ in kernel methods. Particularly, methods that take the 'residual' matrix into account and achieve a balance between efficiency and accuracy, ideally at the same level of full GPs or even better, would make a good complementary to the GP community. Fig. 1 demonstrates how the covariance matrix is approximated.

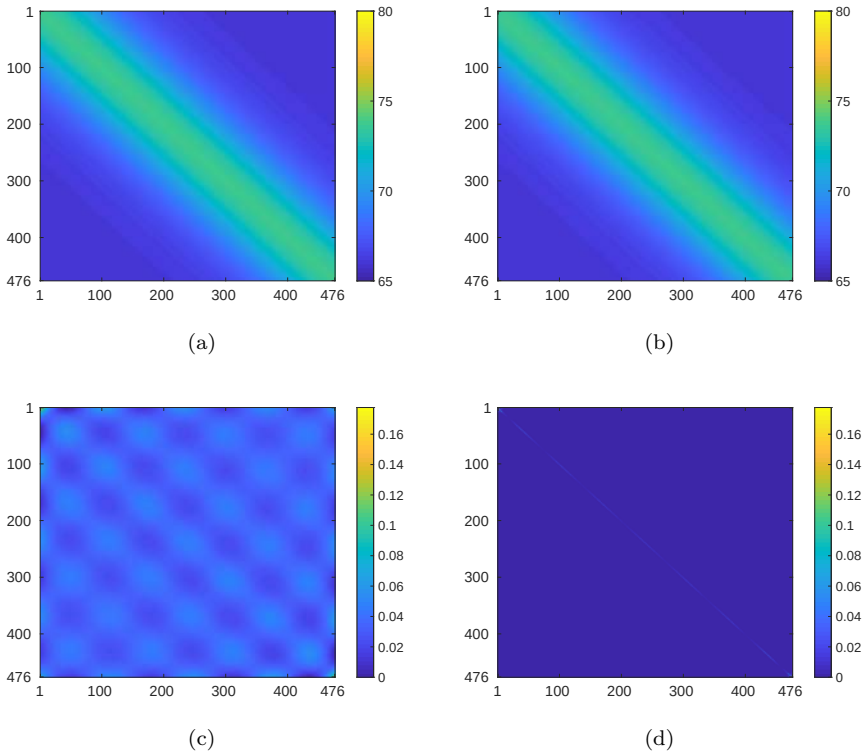

Figure 1: Matrices involved in the covariance matrix approximation: (a) $\mathbf{K}_{nn}$, (b) $\mathbf{L}_{nn}$, (c) the 'residual' matrix $\tilde{\mathbf{A}}$, (d) $\mathbf{A}$. The $\mathbf{L}_{nn}$ is usually used for approximating $\mathbf{K}_{nn}$, with $\tilde{\mathbf{A}} = \mathbf{K}_{nn} - \mathbf{L}_{nn}$ discarded. The $\mathbf{A}$ is the SDD projection matrix of $\tilde{\mathbf{A}}$, and we use $\mathbf{L}_{nn} + \mathbf{A}$ to approximate $\mathbf{K}_{nn}$. The figures are generated from Synthetic dataset 2 with $n = 476$ for demonstration.

3. The Gaussian Process Method and Kernel Approximations

3.1. Background Knowledge

Given a set of *training data* $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \cdots, n\}$ where $\mathbf{x}_i \in \mathcal{X}$ is the input and $y_i \in \mathbb{R}$ is the observation, we can determine a GP model $f(\cdot)$ to predict y_* for a new input $\mathbf{x}_*$. For instance, when the output is one dimensional, the GP model is formulated as

$$f \sim \mathcal{GP}\big(\bar{f}(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big), \quad y = f(\mathbf{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2), \tag{2}$$

where $\bar{f} : \mathcal{X} \to \mathbb{R}$ is the *mean function* defined as

$$\bar{f}(\mathbf{x}) = \mathbb{E}\big[f(\mathbf{x})\big], \tag{3}$$

and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the *kernel function* [19] defined as

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\big[(f(\mathbf{x}) - \bar{f}(\mathbf{x}))(f(\mathbf{x}') - \bar{f}(\mathbf{x}'))\big], \tag{4}$$

where ε is the additive independent identically distributed Gaussian measurement noise with variance $\sigma^2 \neq 0$, and $\mathbb{E}[\cdot]$ denotes the expectation of a random variable.

Given $\mathbf{x}_i$ a $D \times 1$ vector, the n inputs can be aggregated into a matrix $\mathbf{X}_{D\times n}$, or briefly $\mathbf{X}$ with the corresponding output vector $\mathbf{y}_{n\times 1}$, or $\mathbf{y}$. Similarly, the function values at the test inputs $\mathbf{X}_*$ with dimensions of $D \times N$ can be denoted as $\mathbf{f}_*$, and we next write the joint distribution of $\mathbf{y}$ and $\mathbf{f}_*$ as

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_{nn} + \sigma^2\mathbf{I} & \mathbf{K}_{nN} \\ \mathbf{K}_{Nn} & \mathbf{K}_{NN} \end{bmatrix}\right), \tag{5}$$

where $\mathbf{I}$ represents the identity matrix. $\mathbf{K}_{nn} + \sigma^2\mathbf{I}$ is the $n \times n$ prior covariance matrix of $\mathbf{y}$ with entry $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) + \sigma^2\delta_{ij}$, where δ_{ij} is one iff $i = j$ and zero otherwise, and $\mathbf{x}_i$ and $\mathbf{x}_j$ are column vectors from $\mathbf{X}$. The matrix $\mathbf{K}_{NN}$ denotes the $N \times N$ prior covariance matrix of $\mathbf{f}_*$ with entry $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are column vectors from $\mathbf{X}_*$. The matrices $\mathbf{K}_{Nn}$ and $\mathbf{K}_{nN}$ satisfy $\mathbf{K}_{Nn} = \mathbf{K}_{nN}^{\mathrm{T}}$, and the entry of the $N \times n$ prior covariance matrix of $\mathbf{f}_*$ and $\mathbf{y}$ is $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i$ is a column vector from $\mathbf{X}_*$ and $\mathbf{x}_j$ is a column vector from $\mathbf{X}$.

By deriving the conditional distribution of $\mathbf{f}_*$ from (5), where the prior mean is set to be zero for simplicity [6], we have the predictive posterior (also given in equation (2.22), Chapter 2 of [6]) at new inputs $\mathbf{X}_*$ as

$$\mathbf{f}_* | \mathbf{X}, \mathbf{y}, \mathbf{X}_* \sim \mathcal{N}\big(\bar{\mathbf{f}}_*, \mathrm{cov}(\mathbf{f}_*)\big), \tag{6}$$

where

$$\bar{\mathbf{f}}_* \triangleq \mathbb{E}\big[\mathbf{f}_*|\mathbf{X}, \mathbf{y}, \mathbf{X}_*\big] = \mathbf{K}_{Nn}\big[\mathbf{K}_{nn} + \sigma^2\mathbf{I}\big]^{-1}\mathbf{y}, \tag{7}$$

is the prediction at $\mathbf{X}_*$, and

$$\mathrm{cov}(\mathbf{f}_*) = \mathbf{K}_{NN} - \mathbf{K}_{Nn}\big[\mathbf{K}_{nn} + \sigma^2\mathbf{I}\big]^{-1}\mathbf{K}_{Nn}^{\mathrm{T}}, \tag{8}$$

denotes the covariance of $\mathbf{f}_*$.

The hyperparameter $\boldsymbol{\theta}$ incorporated in the mean and covariance functions underpin the predictive performance of GP models, and they are usually estimated by maximising the logarithm of the marginal likelihood

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^{\mathrm{T}}\big(\mathbf{K}_{nn} + \sigma^2\mathbf{I}\big)^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{nn} + \sigma^2\mathbf{I}| - \frac{n}{2}\log 2\pi. \tag{9}$$

3.2. Kernel Approximations

Scalable GPs have been extensively studied recently, which aims at alleviating the computational complexity while retaining the favourable prediction quality of GPs [3]. One of the most popular methods for kernel matrix approximation utilises a low-rank matrix to approximate $\mathbf{K}_{nn}$, hence decreasing the computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(m^2 n)$ with $m \ll n$, as introduced in [8]. For instance, in the eigendecomposition paradigm, the $\mathbf{K}_{nn}$ is presented as

$$\mathbf{K}_{nn} = \mathbf{U}_{nn}\boldsymbol{\Lambda}_{nn}\mathbf{U}_{nn}^{\mathrm{T}}, \tag{10}$$

where $\mathbf{U}_{nn}$ comprises all the eigenvectors and is orthonormal, and $\boldsymbol{\Lambda}_{nn} = \mathrm{diag}(\lambda_i)$, $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant 0$ is a diagonal matrix with eigenvalues being the diagonal entries. If we choose eigenvectors corresponding to the $m < n$ largest eigenvalues and build $\mathbf{U}_{nm} \in \mathbb{R}^{n\times m}$ and let $\boldsymbol{\Lambda}_{mm} = \mathrm{diag}(\lambda_1, \cdots, \lambda_m)$, we then have the approximation

$$\begin{aligned}\big(\mathbf{K}_{nn} + \sigma^2\mathbf{I}\big)^{-1} &\approx \big(\mathbf{U}_{nm}\boldsymbol{\Lambda}_{mm}\mathbf{U}_{nm}^{\mathrm{T}} + \sigma^2\mathbf{I}\big)^{-1} \\ &= \sigma^{-2}\mathbf{I}_n - \sigma^{-2}\mathbf{U}_{nm}\big(\sigma^2\boldsymbol{\Lambda}_{mm}^{-1} + \mathbf{U}_{nm}^{\mathrm{T}}\mathbf{U}_{nm}\big)^{-1}\mathbf{U}_{nm}^{\mathrm{T}}\end{aligned} \tag{11}$$

following the Sherman-Morrison-Woodbury formula. Clearly, with the existence of eigendecomposition, this approximation significantly reduces the computational complexity. However, the computational cost of the eigendecomposition is $\mathcal{O}(n^3)$, which makes the method less favourable.

The Nyström method [8] was then proposed to replace the eigendecomposition to approximate the covariance matrix

$$\mathbf{K}_{nn} \approx \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^{\mathrm{T}}, \tag{12}$$

where $\mathbf{K}_{nm}$ is obtained by randomly choosing m rows or columns of $\mathbf{K}_{nn}$ without replacement. Williams et al. [8] observe that even when $m \ll n$, there is no significant accuracy reduction when using (12) in the GP approach. With the Nyström method and the Sherman-Morrison-Woodbury formula, one can see from

$$\left(\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^{\mathrm{T}} + \sigma^2\mathbf{I}_n\right)^{-1} = \sigma^{-2}\mathbf{I}_n - \sigma^{-2}\mathbf{K}_{nm}\left(\sigma^2\mathbf{K}_{mm} + \mathbf{K}_{nm}^{\mathrm{T}}\mathbf{K}_{nm}\right)^{-1}\mathbf{K}_{nm}^{\mathrm{T}} \tag{13}$$

that the computational complexity is reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(mn^2)$.

4. Symmetric Diagonally Dominant Projection-based Gaussian Process

4.1. Symmetric Diagonally Dominant Projection

As discussed in Section 2 and shown in Fig. 1, most kernel approximation methods can be formulated as

$$\mathbf{K}_{nn} \approx \mathbf{L}_{nn}, \quad \text{with} \quad \mathbf{K}_{nn} = \mathbf{L}_{nn} + \tilde{\mathbf{A}}, \tag{14}$$

where $\mathrm{rank}(\mathbf{L}_{nn}) = K \ll n$, and $\tilde{\mathbf{A}}$ is the 'residual' matrix. The $\mathbf{L}_{nn}$ matrix is calculated by different methods [6], but the $\tilde{\mathbf{A}}$ matrix is typically discarded. Intuitively, full GPs should perform better than the kernel-based approximations in spite of the heavy computational loads. However, in some classification tasks,

the latter seems to outperform the full GPs [8]. This is believed to be caused by the high correlations of observations of the latent variables. Therefore, how the number of latent variables is determined would influence the performance of GPs variants, and leads to the phenomenon that full GPs sometimes perform poorer than kernel approximations. In this paper, we show that taking the 'residual' matrix $\tilde{\mathbf{A}}$ into consideration can mitigate the full GP challenges, which is performed by introducing the SDD projection into (14).

The SDD projection problem aims at finding an approximation of $\mathbf{K}_{nn}$ in the form

$$\mathbf{K}_{nn} \approx \mathbf{L}_{nn} + \mathbf{A}, \tag{15}$$

where $\mathbf{L}_{nn} = \sum_{k=1}^{K} \lambda_k \xi_k \xi_k^{\mathrm{T}}$ with $K = \operatorname{rank}(\mathbf{L}_{nn}) \ll n$, and $\mathbf{A}$ is a symmetric c-diagonally dominant matrix defined as

$$\begin{aligned} &\mathcal{SDD}_c^+ \\ &= \Big\{ \mathbf{A} = (a_{ij})_{n\times n} : \mathbf{A} = \mathbf{A}^{\mathrm{T}}, a_{jj} \geqslant c \textstyle\sum_{i:i\neq j} |a_{ji}| \quad \text{for all} \quad 1 \leqslant j \leqslant n \Big\}, \end{aligned} \tag{16}$$

with $c \in \mathbb{R}^+$ [26], when $c = 1$, we omit the subscript and denote (16) as $\mathcal{SDD}^+$. Ke et al. [26] also describe the problem of finding $\mathbf{A}$ as

$$\min_{(\mathbf{L}_{nn}, \mathbf{A})} \|\mathbf{K}_{nn} - \mathbf{L}_{nn} - \mathbf{A}\|_F, \tag{17}$$

where $\|\cdot\|_F$ is the matrix Frobenius norm. They also provide a nonconvex solution to (17), which can be achieved at the cost of $\mathcal{O}\big(n^2 \max\{\log(n), K\}\big)$.

In this paper, algorithm 1 is used to find $\mathbf{A} \in \mathcal{SDD}_c^+$, i.e. satisfying (16), which is next added to the low rank matrix $\mathbf{L}_{nn}$ to approximate $\mathbf{K}_{nn}$, as shown in (15). Please note that $\mathcal{P}_{\mathcal{DD}_c^+}(\cdot)$ in Algorithm 1 denotes the projection of an arbitrary matrix to a c-diagonally dominant cone indicated by $\mathcal{DD}_c^+$. When $c = 1$, we denote it as $\mathcal{DD}^+$ for brevity. Here we provide a three-dimensional example to help understanding Step 6, which is also known as Mendoza-Raydan-Tarazaga projection [27].

Algorithm 1 The Iterative SDD Projection

Input: The covariance matrix $\mathbf{K}_{nn}$ and a tolerance criterion (TOL) and TOL decreasing step η.

Output: $\mathbf{A} \in \mathcal{SDD}_c^+$.

Let $\mathbf{A}^{(0)} = \mathbf{K}_{nn} - \mathbf{L}_{nn}$, where $\mathbf{L}_{nn} = \mathbf{U}_{nm}\Lambda_{mm}\mathbf{U}_{nm}^{\mathrm{T}}$, with $m \leqslant n$ and $\Lambda_{mm} = \mathrm{diag}\{\lambda_1, \cdots, \lambda_m\}$, $\hat{\mathbf{A}}^{(0)} = \mathbf{A}^{(0)}$ and $\mathbf{J}^{(0)} = \mathbf{0}_{nn}$.
for $t = 1, 2, \cdots$ **do**
  $\hat{\mathbf{A}}^{(t)} = \mathcal{P}_{\mathcal{DD}_c^+}(\mathbf{A}^{(t-1)} - \mathbf{J}^{(t-1)})$. ⟵ Diagonally Dominant Projection
  $\mathbf{A}^{(t)} = \left(\hat{\mathbf{A}}^{(t-1)} + (\hat{\mathbf{A}}^{(t-1)})^{\mathrm{T}}\right)/2$. ⟵ Symmetric Projection
  $\mathbf{J}^{(t)} = \mathbf{J}^{(t-1)} + (\hat{\mathbf{A}}^{(t)} - \mathbf{A}^{(t-1)})$.
  if $\|\mathbf{J}^{(t)} - \mathbf{J}^{(t-1)}\|_F \leqslant$ TOL **then**
    $\mathbf{A} = \mathbf{A}^{(t)}$.
    if (32) is satisfied **then**
      stop.
    else
      TOL = TOL - η, continue.
    end if
  else
    continue.
  end if
end for

Example 1: Given a 3×3 matrix

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix}, \tag{18}$$

we can regard each row as a vector and they are depicted in Fig. 2. Without loss of generality, let's assume that $\mathbf{G}$ is entry-wisely positive and take $\mathbf{g}_3 = [g_{31}, g_{32}, g_{33}]$ an example. For some tasks such as dimension reduction, we need to project $\mathbf{g}_3$ to obtain $\tilde{\mathbf{g}}_3 = [0, g_{32}, g_{33}]$ as the best approximation of $\mathbf{g}_3$ in terms of the minimum Euclidean distance $\|\mathbf{g}_3 - \tilde{\mathbf{g}}_3\|_2$. Obviously, if $g_{32} \geqslant g_{33}$, we cannot guarantee that a diagonally dominant approximation of $\mathbf{G}$ can be obtained through this type of projection. In this paper, we follow the process introduced in [27] to guarantee that $\hat{\mathbf{G}} \in \mathcal{DD}^+$ is valid for approximating $\mathbf{G}$. To be specific, for the i-th row of a $h \times h$ matrix $\mathbf{G}$ ($i = 3$ and $h = 3$ in our

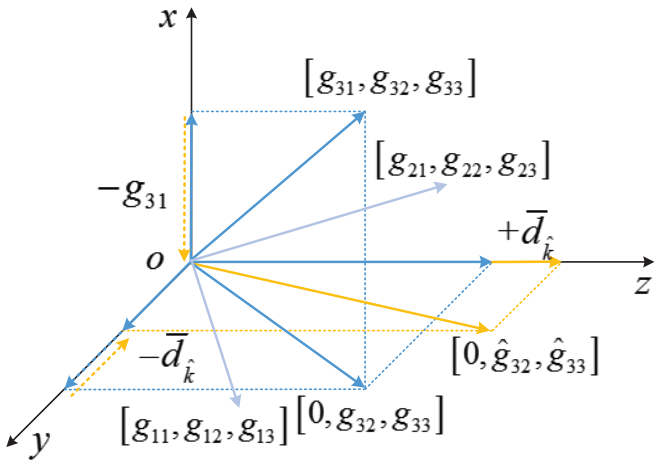

Figure 2: Illustration of Mendoza-Raydan-Tarazaga projection [27].

case), we first do

- $d_k = \sum_{k=j}^{h} g_{ik} - g_{ii}$, where $k \neq i$ and $1 \leqslant k \leqslant h$;
- $c_k = h - k + 1$ for $k < i$, and $c_k = h - k + 2$ for $k > i$;
- $\bar{d}_k = d_k / c_k$.

Next, we find $\hat{k}$ such that $g_{i\hat{k}} > 0$ and $\bar{d}_{\hat{k}} \leqslant g_{i\hat{k}}$, and finally the approximation matrix $\hat{\mathbf{G}}$ is generated as follows:

$$\hat{g}_{ik} = \begin{cases} g_{ik} + \bar{d}_{\hat{k}}, & k = i, \\ g_{ik} - g_{ik}, & 1 \leqslant k \leqslant \hat{k} - 1 \text{ and } k \neq i, \\ g_{ik} - \bar{d}_{\hat{k}}, & \hat{k} \leqslant k \leqslant h \text{ and } k \neq i. \end{cases} \tag{19}$$

Fig. 2 shows a geometrical example of how $\hat{\mathbf{g}}_3 = [0, \hat{g}_{32}, \hat{g}_{33}]$ is generated. In this case, we assume that $\hat{k} = 2$. Therefore, $\hat{g}_{31} = g_{31} - g_{31} = 0$, $\hat{g}_{32} = g_{32} - \bar{d}_{\hat{k}}$, and $\hat{g}_{33} = g_{33} + \bar{d}_{\hat{k}}$. By comparing $\tilde{\mathbf{g}}_3$ with $\hat{\mathbf{g}}_3$, one sees that they are both approximations of $\mathbf{g}_3$. However, $\hat{\mathbf{g}}_3$ is generated in a way to ensure that $\hat{g}_{33} \geqslant \hat{g}_{32}$ stands, such that $\hat{\mathbf{G}}$ obtained is diagonally dominant. Similarly to the

uniqueness of $\tilde{\mathbf{g}}_3$, $\hat{\mathbf{g}}_3$ is proved to be unique as well [27]. This ensures that if $\mathbf{G}$ has a diagonally dominant counterpart, it would be unique. We are interested in the latter as it leads to a diagonally dominant matrix $\hat{\mathbf{G}}$ that is usually positive definite, which helps us solving the matrix inverse problem in GPs.

4.2. Neumann Series for Diagonally Dominant Matrix Inversion Approximation

After solving (17) with Algorithm 1, we can next approximate $(\mathbf{K}_{nn}+\sigma^2\mathbf{I})^{-1}$ with

$$(\mathbf{K}_{nn} + \sigma^2\mathbf{I})^{-1} \approx (\mathbf{L}_{nn} + \mathbf{A} + \sigma^2\mathbf{I})^{-1} = (\mathbf{L}_{nn} + \mathbf{M})^{-1}, \tag{20}$$

where $\mathbf{M} = \mathbf{A}+\sigma^2\mathbf{I}$. Although the SDD matrix $\mathbf{A}$ ensures that $\mathbf{M} \in \mathcal{SDD}_c^+$, the computational cost of $\mathbf{M}^{-1}$ is still at $\mathcal{O}(n^3)$. This is not favourable. Therefore, we introduce the Neumann series to approximate $\mathbf{M}^{-1}$.

Given a matrix $\mathbf{M}$, the inverse $\mathbf{M}^{-1}$ of which can be expanded as the following Neumann series [13]

$$\mathbf{M}^{-1} = \sum_{i=0}^{\infty} \big(\mathbf{X}^{-1}(\mathbf{X} - \mathbf{M})\big)^i \mathbf{X}^{-1}, \tag{21}$$

which holds if $\lim_{i\to\infty} \big(\mathbf{I} - \mathbf{X}^{-1}\mathbf{M}\big)^i = \mathbf{0}$ is satisfied. In general, we can use (21) to approximate the inverse of any matrix. However, approximation with quick convergence requires $\mathbf{M}$ to be diagonally dominant [12, 13]. In our case, suppose

$$\mathbf{M} = \mathbf{A} + \sigma^2\mathbf{I} \triangleq \mathbf{D}_M + \mathbf{E}_M, \tag{22}$$

where $\mathbf{D}_M$ is the main diagonal of $\mathbf{M}$ and $\mathbf{E}_M$ is the hollow. If we substitute $\mathbf{X}$ in equation (21) by $\mathbf{D}_M$, we get

$$\mathbf{M}^{-1} = \sum_{i=0}^{\infty} \big(-\mathbf{D}_M^{-1}\mathbf{E}_M\big)^i \mathbf{D}_M^{-1}, \tag{23}$$

which is guaranteed to converge when $\lim_{i\to\infty} \big(-\mathbf{D}_M^{-1}\mathbf{E}_M\big)^i = \mathbf{0}$. When Neumann series given in (23) converges, we can then approximate $\mathbf{M}^{-1}$ with only

the first L terms. The L-term approximation is computed as follows:

$$\widetilde{\mathbf{M}}_L = \sum_{i=0}^{L-1} \big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^i \mathbf{D}_M^{-1}, \tag{24}$$

For instance, when $L = 1, 2, 3$, we have the approximations

$$\widetilde{\mathbf{M}}_L = \begin{cases} \mathbf{D}_M^{-1}, & L = 1 \\ \mathbf{D}_M^{-1} - \mathbf{D}_M^{-1}\mathbf{E}_M\mathbf{D}_M^{-1}, & L = 2 \\ \mathbf{D}_M^{-1} - \mathbf{D}_M^{-1}\mathbf{E}_M\mathbf{D}_M^{-1} + \mathbf{D}_M^{-1}\mathbf{E}_M\mathbf{D}_M^{-1}\mathbf{E}_M\mathbf{D}_M^{-1}. & L = 3 \end{cases} \tag{25}$$

From equations (24) and (25), we see that Neumann series approximation reduces the cost of $\mathbf{M}^{-1}$ from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ when $L \leq 2$, which is of particular favour when n becomes large. Wu et al. also mention that the calculation of the Neumann series can be accelerated by proper adjustment of the terms [13]. As mentioned earlier, when $\mathbf{M}$ is diagonally dominant [12, 13], the approximation would be both quick and accurate, this would help to improve the GPs performance.

4.3. Symmetric Diagonally Dominant Projection-based Gaussian Processes

By transforming the 'residual' matrix $\tilde{\mathbf{A}}$ into a diagonally dominant matrix $\mathbf{A}$, and approximating $\mathbf{M}^{-1}$ with (24), we know the inverse matrix involved in GPs can be approximated through

$$\begin{aligned} (\mathbf{K}_{nn} + \sigma^2\mathbf{I})^{-1} &\approx (\mathbf{L}_{nn} + \mathbf{A} + \sigma^2\mathbf{I})^{-1} \\ &= (\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^{\mathrm{T}} + \mathbf{M})^{-1} \\ &= \mathbf{M}^{-1} - \mathbf{M}^{-1}\mathbf{K}_{nm}\big(\mathbf{K}_{mm} + \mathbf{K}_{nm}^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{K}_{nm}\big)^{-1}\mathbf{K}_{nm}^{\mathrm{T}}\mathbf{M}^{-1}. \end{aligned} \tag{26}$$

The overall SDD projection-based GP (SDD GP) is described in Algorithm 2. For brevity, we denote the approximation of $\mathbf{K}_{nn}$ as $\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^{\mathrm{T}}$. One can observe from Algorithm 2 that with the proposed SDD GP, the computational cost is reduced overall from $\mathcal{O}(n^3)$ to $\mathcal{O}(mn^2)$, and we are still able to retain

Algorithm 2 The SDD projection-based GPs

Input: $\mathbf{X}$ the inputs, $\mathbf{y}$ the outputs, k the kernel function, σ^2 the observation noise level, TOL a tolerance criterion, and the test inputs $\mathbf{X}_*$

Output: $\bar{\mathbf{f}}_*$ and $\mathrm{cov}(\mathbf{f}_*)$

1: $\mathbf{K}_{nn} \approx \mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{nm}^{\mathrm{T}} + \mathbf{A}$, with $\mathbf{A}$ diagonally dominant $\longleftarrow$ Solving (17)
2: L-term Neumann Series Approximation: $\widetilde{\mathbf{M}}_L \approx \mathbf{M}^{-1}$, with $\mathbf{M} = \mathbf{A} + \sigma^2\mathbf{I}$
3: Cholesky factorisation: $\mathbf{C} := \mathrm{cholesky}(\mathbf{K}_{mm} + \mathbf{K}_{nm}^{\mathrm{T}}\widetilde{\mathbf{M}}_L\mathbf{K}_{nm})$
4: $(\mathbf{K}_{nn} + \sigma^2\mathbf{I})^{-1} \approx \widetilde{\mathbf{M}}_L - \mathbf{V}^{\mathrm{T}}\mathbf{V}$, with $\mathbf{V} = \mathbf{C} \setminus \mathbf{Y}_{mn}$ and $\mathbf{Y}_{mn} = \mathbf{K}_{nm}^{\mathrm{T}}\widetilde{\mathbf{M}}_L$.
5: Mean: $\bar{\mathbf{f}}_* = \mathbf{K}_{Nn}\mathbf{Q}\mathbf{y}$, with $\mathbf{Q} = \widetilde{\mathbf{M}}_L - \mathbf{V}^{\mathrm{T}}\mathbf{V}$
6: Covariance: $\mathrm{cov}(\mathbf{f}_*) = \mathbf{K}_{NN} - \mathbf{K}_{Nn}\mathbf{Q}\mathbf{K}_{Nn}^{\mathrm{T}}$, with $\mathbf{Q} = \widetilde{\mathbf{M}}_L - \mathbf{V}^{\mathrm{T}}\mathbf{V}$

the 'residual' matrix.

4.4. Theoretical Performance Analysis

By comparing the proposed approach with a full GP model, one can see that the major difference is demonstrated by

$$\begin{aligned}(\mathbf{K}_{nn} + \sigma^2\mathbf{I})^{-1} &= (\mathbf{L}_{nn} + \tilde{\mathbf{A}} + \sigma^2\mathbf{I})^{-1} \leftarrow \text{GP} \\ &\approx (\mathbf{L}_{nn} + \mathbf{A} + \sigma^2\mathbf{I})^{-1} \leftarrow \text{SDD GP}.\end{aligned} \tag{27}$$

Since $\mathcal{SDD}_c^+$ and Neumann series approximations are key in the proposed approach, we present a theoretical analysis of their impact on the algorithm performance as follows.

Lemma 1: Given $\mathbf{A}$ as the $\mathcal{SDD}^+$ approximation of $\tilde{\mathbf{A}}$, one can then take $\mathbf{M}$ as an $\mathcal{SDD}_c^+$ approximation of $\tilde{\mathbf{A}} + \sigma^2\mathbf{I}$, i.e. $\mathbf{M} \in \mathcal{SDD}_c^+$ with $c > 1$ satisfied.

Proof: According to the definition of $\mathbf{M}$, we know that the diagonal entries of $\mathbf{M}$ are $m_{jj} = a_{jj} + \sigma^2$, $j = 1, \cdots, n$, where a_{jj} are the diagonal entries of $\mathbf{A}$. The off-diagonal entries of $\mathbf{M}$ and $\mathbf{A}$ are identical. Therefore, we have

$$\begin{cases} m_{11} \geqslant c_1 \sum_{i:i\neq 1} |a_{1i}|, \\ \vdots \\ m_{jj} \geqslant c_j \sum_{i:i\neq j} |a_{ji}|, \\ \vdots \\ m_{nn} \geqslant c_n \sum_{i:i\neq n} |a_{ni}|, \end{cases} \tag{28}$$

where $c_j > 1,\ j = 1, \cdots, n$ hold for $\sigma^2 > 0$, which is normally the case as there is no point to consider a zero mean zero variance Gaussian noise. It is straightforward that by setting $c = \min\{c_1, c_2, \cdots, c_n\}$, **Lemma 1** stands. According to [26], $\mathbf{M}^{-1}$ can be bounded by

$$||\mathbf{M}^{-1}||_F \leqslant \frac{c}{c-1}||[\text{diag}(\mathbf{M})]^{-1}||_F, \tag{29}$$

which is the condition of making $(\mathbf{L}_{nn} + \mathbf{A} + \sigma^2\mathbf{I})^{-1}$ a good estimator of $(\mathbf{K}_{nn} + \sigma^2\mathbf{I})^{-1}$. More details can be found on the right half of Page 2 in [26].

Furthermore, because $\mathbf{M}^{-1}$ is approximated in the paper by the L-term Neumann series given in (24), we then investigate the error between (23) and (24) to demonstrate the approximation performance. Suppose the error is denoted by

$$\begin{aligned}
\Delta_{M|L} &= \sum_{i=L}^{\infty} \big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^i \mathbf{D}_M^{-1} \\
&= \big(-\mathbf{D}^{-1}\mathbf{E}_M \big)^L \sum_{i=0}^{\infty} \big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^i \mathbf{D}_M^{-1} \\
&= \big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^L \mathbf{M}^{-1}.
\end{aligned} \tag{30}$$

Let's assume that there exists a column vector $\mathbf{h}_M$, such that the l_2 norm of $||\mathbf{M}^{-1}\mathbf{h}_M||_2 \leq \infty$. Then according to [13] (Subsection B in Section III), the l_2-norm of $\Delta_{M|L}\mathbf{h}_M$ satisfies

$$\begin{aligned}
||\Delta_{M|L}\mathbf{h}_M||_2 &= ||\big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^L \mathbf{M}^{-1}\mathbf{h}_M||_2 \\
&\leqslant ||\big(-\mathbf{D}_M^{-1}\mathbf{E}_M \big)^L||_F ||\mathbf{M}^{-1}\mathbf{h}_M||_2 \\
&\leqslant ||\mathbf{D}_M^{-1}\mathbf{E}_M||_F^L ||\mathbf{M}^{-1}\mathbf{h}_M||_2.
\end{aligned} \tag{31}$$

We can see that if

$$||\mathbf{D}_M^{-1}\mathbf{E}_M||_F < 1 \tag{32}$$

holds, then the approximation error approaches zero as L increases [13], implying that the approximation of (23) by (24) is becoming better. It is also

demonstrated in [12] that the approximation accuracy of $\mathbf{M}^{-1}$ using (24) is fairly high when $\mathbf{M}$ is diagonally dominant, which is case in our paper.

By increasing L, the approximation performance of Neumann series can be improved. In addition, we can use the SDD projection to enhance the approximation performance. One can see that the matrices $\mathbf{A}$, $\mathbf{D}_M$, and $\mathbf{E}_M$ are linked through (22), which shows that $\mathbf{D}_M$ and $\mathbf{E}_M$ change along with $\mathbf{A}$. When $\mathbf{A}$ is updated by Algorithm 1, a new set of $\mathbf{D}_M$ and $\mathbf{E}_M$ are generated according to (22) subsequently and condition (32) is checked. When the condition is satisfied, one can see from (31) that the performance of approximating (23) by (24) can be further improved.

5. Performance Validation

5.1. Datasets and Baselines

To validate the proposed approach, we design two sets of experiments. In the first set, two synthetic datasets and the Mauna Loa CO_2 [1] data are processed. The synthetic datasets are generated by two deterministic functions that are perturbed by Gaussian noises, as shown in

$$y = \sin(x) + v_1, \tag{33}$$

with $v_1 \sim \mathcal{N}(0, 0.15)$ and $x \in [-5.0, 5.0]$, and

$$y = 5x^2 * \sin(12x) + (x^3 - 0.5) * \sin(3x - 0.5) + 4 * \cos(2x) + v_2, \tag{34}$$

with $v_2 \sim \mathcal{N}(0, 0.45)$ and $x \in [-0.2, 1.2]$. We adopt models (33) and (34) because 1) they are substantially nonlinear functions with function values known at any given inputs, and this comparison of the proposed approach with other approaches can demonstrate well their performance. 2) The impact of different noises on the solutions can be easily demonstrated. This helps to generate

[1] https://iridl.ldeo.columbia.edu/SOURCES/.KEELING/.MAUNA_LOA/

datasets with various noise levels, including outliers, to test the robustness of the proposed approach. 3) The number of samples can be easily controlled to test the proposed approach on datasets of different sizes.

In the second set, we separately process temperature and NO_2 concentration data from Sheffield, the United Kingdom, and from Peshawar, Pakistan. For the data in the second set, we converted the original time stamp (year-day-hour-minute) into decimal form first, then both the time and the observation are standardised for all the GP models. In our case, the data were collected every 15 minutes in both cities from June 22, 2019 to July 14, 2019. Therefore, four samples can be collected in each hour and 96 samples are accumulated per day. This information enables us to convert the year-day-hour-minute time stamps into decimal numbers through

$$t_{\text{dec}} = t_{\text{year}} + (t_{\text{day}} - 1) + (t_{\text{hour}} * 4 + t_{\text{minute}}/60 * 4)/96, \tag{35}$$

where t_{dec} is the decimal number, t_{year} is the year, t_{day} denotes the day, for example, the 173-th day of the year is June 22, 2019, t_{hour} is the hour, and t_{minute} is the minute. Note that since the data were collected every 15 minutes, t_{minute} can only be 0, 15, 30, and 45 and this is part of the current validation.

Since the proposed approach belongs to the group with prior approximation, we hence compare it with full GP variants with different kernels [6] and a sparse GP model, i.e. the Fully Independent Training Conditional (FITC) [2, 4] GP. We also compare the developed approach with the variants where the 'residual' $\tilde{\mathbf{A}}$ are discarded. We denote the model as SDD^{-} GP for short.

5.2. Performance Metrics

In order to assess the overall performance of different GP variants, we employ the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to evaluate the prediction performance. These two metrics are defined as:

$$MAE = \frac{1}{N_s} \sum\nolimits_{i=1}^{N_s} |y_i - \hat{y}_i|, \tag{36}$$

$$RMSE = \sqrt{\frac{1}{N_s}\sum_{i=1}^{N_s}(y_i - \hat{y}_i)^2}, \tag{37}$$

where y_i and $\hat{y}_i$ indicate the i-th true value and prediction, respectively, N_s is the number of samples in the testing set.

Note for Sheffield and Peshawar NO_2 and temperature datasets, and the Mauna Loa CO_2 dataset, we do not have 'exact' function values. To evaluate the performance, we compare observations with predictions of different GP models.

5.3. Implementation Details

We have implemented the SDD GP and the SDD$^-$ GP with Python based on GPy [2], and adjusted the full GP and the sparse GP from GPy to process our data for comparison.

For each dataset, we take N_t samples, which is 75% the number of samples as training data and the left as testing data, with size N_s. To be specific, for the two sets of synthetic data and the Mauna Loa CO_2 dataset, there are 635 samples for each. For the Sheffield and Peshawar temperature and air quality data, we use 2016 samples. This makes $N_t = 477$ for the Mauna Loa CO_2 and the synthetic datasets, and $N_t = 1512$ for the Sheffield and Peshawar temperature and air quality datasets. For sparse GP, we set the number of inducing points to roughly 1.5% the number of samples. For the SDD GP and SDD$^-$ GP, we take m eigenvectors for covariance matrix approximation to achieve good performance. We increase m from 5 up to half of the number of samples and record the corresponding RMSE and MAE, to investigate the impact of m on the performance. The iteration index t in Algorithm 1 is set to 15 for results generation. To study the algorithms' performance, we use various kernels as listed in Table 1 to design composition kernels to capture different data patterns. The Squared Exponential Automatic Relevance Determination (SE-ARD) kernel is also applied to each dataset for comparison. The kernel settings are given in Table 2. The hyperparameters are estimated by maximising the

[2] https://sheffieldml.github.io/GPy/

Table 1: Covariance kernels used in this paper.

Kernel Name	Covariance Function
Squared Exponential	$k_{\text{SE}}(x, x') = \sigma_s^2 \exp\left(-\frac{(x-x')^2}{2\ell_s^2}\right)$
Rational Quadratic	$k_{\text{RQ}}(x, x') = \sigma_r^2 \left(1 + \frac{(x-x')^2}{2\alpha\ell_r^2}\right)^{-\alpha}$
Periodic	$k_{\text{Per}}(x, x') = \sigma_p^2 \exp\left(-\frac{2\sin^2(\pi\lvert x-x'\rvert/p)}{\ell_p^2}\right)$
White	$k_{\text{W}}(x, x') = \sigma_{\text{noise}}^2$ if $x = x'$, else $k_{\text{W}}(x, x') = 0$
Squared Exponential ARD	$k_{\text{SE-ARD}}(x, x') = \sigma_{se}^2 \exp\left(-\sum_{i=1}^{N} \frac{(x_i - x_i')^2}{2\ell_i^2}\right)$

Table 2: Kernel settings for performance evaluation.

Datasets	Kernel Setting 1	Kernel Setting 2
Synthetic 1	$k_{\text{SE}} + k_{\text{Per}}$	$k_{\text{SE-ARD}}$
Synthetic 2	$k_{\text{SE}} + k_{\text{Per}}$	$k_{\text{SE-ARD}}$
Mauna CO_2	$k_{\text{SE}} + k_{\text{Per}} * k_{\text{SE}} + k_{\text{RQ}} + k_{\text{W}}$	$k_{\text{SE-ARD}}$
Sheffield/Peshawar Temp	$k_{\text{SE}} + k_{\text{Per}} * k_{\text{SE}} + k_{\text{RQ}}$	$k_{\text{SE-ARD}}$
Sheffield/Peshawar NO_2	$k_{\text{SE}} + k_{\text{Per}} * k_{\text{SE}} + k_{\text{RQ}}$	$k_{\text{SE-ARD}}$

logarithm marginal likelihood as shown in (9). We use 'GP-ARD' for brevity hereafter to indicate that the SE-ARD kernel is used.

5.4. Performance and Analysis

Fig. 3 shows the logarithm RMSE of different GP implementations on the synthetic dataset 1, synthetic dataset 2, and the Mauna Loa CO_2 dataset. We can see that the performance of the proposed SDD GP converges to that of the full GP, for both kernel setting 1 and 2. We also notice that kernels would affect the performance of GP variants as well as the proposed approach. For instance, Figs. 3(a) and 3(c) show that GP variants with the SE-ARD kernel (kernel setting 2) outperform their counterparts with kernel setting 1, whereas Fig. 3(b) shows that GP variants with kernel setting 1 perform better in terms of RMSE. To further generalise the results, we change the variances for both (33) and (34) and show how they affect the proposed approach in terms of RMSE. The results

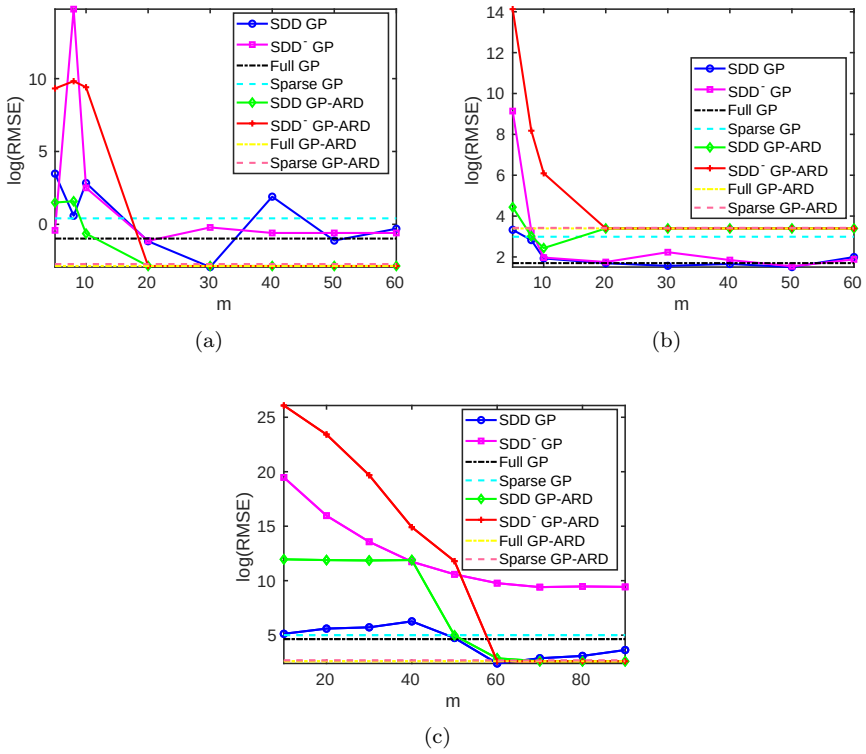

Figure 3: Logarithm RMSE of different methods on the Synthetic and the Mauna Loa datasets: (a) Synthetic dataset 1; (b) Synthetic dataset 2; (c) Mauna Loa dataset.

are given in Fig. 4, where Figs. 4(a) and 4(b) are the results generated with kernel setting 1, and Figs. 4(c) and 4(d) are generated with kernel setting 2. One can see that as the noise variances become big, the prediction RMSE of the proposed approach shows an increasing trend for both functions, for both kernel settings 1 and 2. This is intuitive as the proposed approach is essentially a regression model, whose performance could be degraded by noises.

Fig. 5 shows the RMSE of GP variants with different kernels on Sheffield and Peshawar NO_2 and temperature datasets, respectively. One can see from these figures that the SDD GP generally outperforms the SDD$^-$ GP and the sparse GP. We can also see that as m increases, the SDD GP outperforms or is comparable with the full GP when kernel setting 1 is used. When the SE-ARD

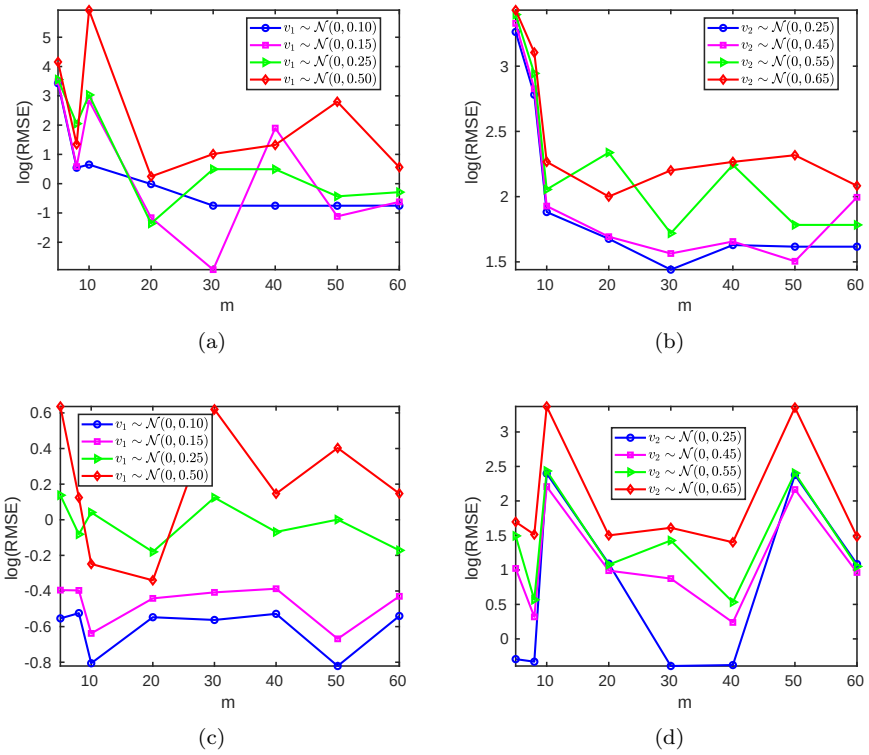

Figure 4: The impact of noise on the performance of the proposed approach: (a) Synthetic dataset 1 with kernel setting 1; (b) Synthetic dataset 2 with kernel setting 1; (c) Synthetic dataset 1 with kernel setting 2; (d) Synthetic dataset 2 with kernel setting 2.

kernel is used, we can see that the performance of SDD GP-ARD approaches the full GP-ARD as m increases till converge.

When focusing on the performance of the proposed approach with different kernels, i.e. SDD GP and SDD GP-ARD, we see that when m increases, their performance in terms of RMSE drops and then tends to converge. The same rule applies to the SDD$^-$ GP and SDD$^-$ GP-ARD. However, the latter does not always show the convergence trends. Particularly, the performance of the SDD$^-$ GP changes dramatically along with m as given in Fig. 5(c) and 5(d). This is because when the 'residual' matrix $\tilde{\mathbf{A}}$ is considered, a better approximation of the covariance matrix is achieved, hence leading to better performance of the SDD GP compared with the SDD$^-$ GP. When m keeps increasing, the

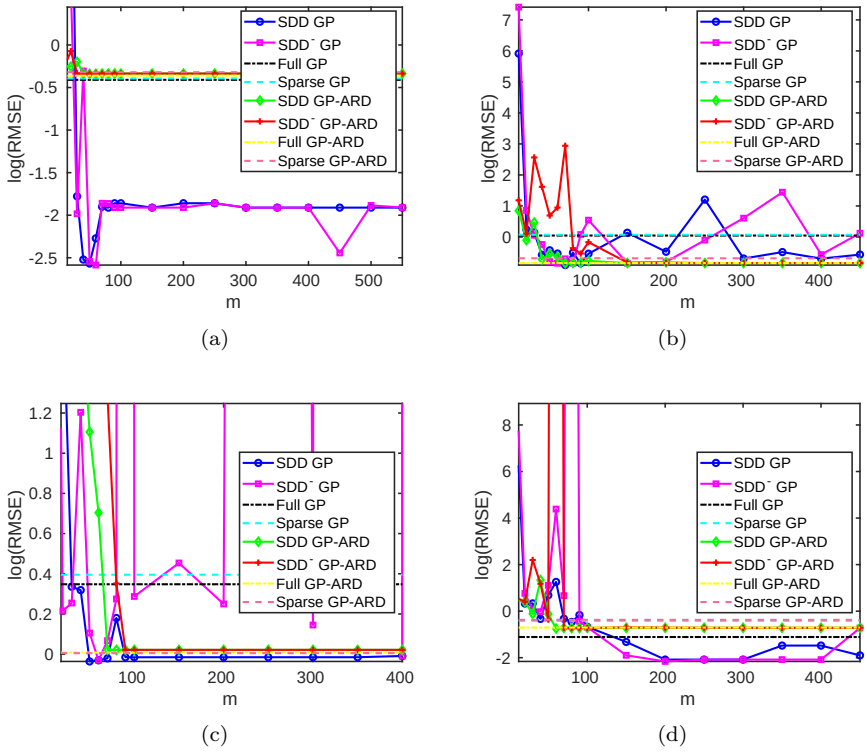

Figure 5: Logarithm RMSE of different methods on Peshawar and Sheffield NO_2 and temperature: (a) Peshawar NO_2; (b) Sheffield NO_2; (c) Peshawar temperature; (d) Sheffield temperature.

'residual' matrix can be neglected, and the performance of SDD GP and SDD$^-$ GP becomes similar. Similar trends can be observed for SDD GP-ARD and SDD$^-$ GP-ARD. It is worth mentioning that when the SE-ARD kernel is used, the RMSEs change less dramatically compared with the results generated by using the kernel setting 1.

To further demonstrate that taking the 'residual' matrix $\tilde{\mathbf{A}}$ into account would improve the performance, we have compared the RMSE and MAE of 1) SDD GP with the SDD$^-$ GP; 2) SDD GP-ARD with the SDD$^-$ GP-ARD on each dataset. Table 3 shows the percentage when the SDD GP (SDD GP-ARD) outperforms the SDD$^-$ GP (SDD$^-$ GP-ARD) in terms of RMSE and MAE, respectively. We see that the percentage of the SDD GP (SDD GP-

Table 3: The percentage of results with RMSE and MAE of the SDD GP smaller than the SDD$^-$ GP: ONE indicates kernel setting 1 in Table 2 is used; TWO indicates kernel setting 2 in Table 2 is used.

	Syn. 1	Syn. 2	M. L. CO_2	S. Temp	P. Temp	S. NO_2	P. NO_2
MAE ONE	50%	75%	100%	59%	59%	53%	69%
RMSE ONE	50%	88%	100%	59%	53%	53%	69%
MAE TWO	100%	100%	78%	94%	65%	74%	94%
RMSE TWO	100%	100%	78%	94%	65%	79%	100%

ARD) outperforms the SDD$^-$ GP (SDD$^-$ GP-ARD) is all equal to or bigger than 50%. This again demonstrates considering $\tilde{\mathbf{A}}$ would help to improve the proposed GP model's performance.

We also list the minimum and median RMSE and MAE of the GP variants on each dataset. The median RMSE and MAE are considered as they demonstrate the resilience of the proposed approach. To be specific, the minimum and median RMSEs and MAEs of the GP variants on synthetic dataset 1, synthetic dataset 2, and the Mauna Loa CO_2 dataset are given in Table 4 and 5, respectively. One can see from Table 4 that the SDD GP achieves the minimum RMSE on all three datasets, and the minimum MAE on synthetic dataset 2 and the Mauna Loa CO_2 dataset. When it comes to the median RMSE and MAE as shown in Table 5, we see that the full GP variants show better performance except on synthetic dataset 1, where the SDD GP-ARD achieves the best performance.

Table 4: The performance comparison among different methods and kernels on the synthetic and public datasets. We take the minimum RMSE and MAE for the SDD GP (SDD GP-ARD) and SDD$^-$ GP (SDD$^-$ GP-ARD).

	Syn. 1		Syn. 2		M. L. CO_2	
——	RMSE	MAE	RMSE	MAE	RMSE	MAE
SDD GP	**0.053**	0.187	**4.502**	**1.832**	**11.054**	**2.735**
SDD$^-$ GP	0.321	0.433	4.681	1.933	1.219e+04	99.151
Full GP	0.079	0.203	5.466	2.013	103.459	8.298
Sparse GP	0.579	0.641	19.876	3.718	148.698	11.168
SDD GP-ARD	0.057	0.194	11.410	2.932	13.444	3.049
SDD$^-$ GP-ARD	0.057	0.194	29.806	4.520	13.447	3.051
Full GP-ARD	0.195	**0.052**	29.805	4.520	13.563	3.063
Sparse GP-ARD	0.211	0.065	30.684	4.926	14.540	3.173

The corresponding results on Sheffield and Peshawar NO_2 and temperature datasets are separately given in Table 6 and 7. One can see that the SDD-GP performs the best in general on the Peshwar temperature and Sheffield

Table 5: The performance comparison among different methods and kernels on the synthetic and public datasets. We take the median RMSE and MAE for the SDD GP (SDD GP-ARD) and SDD$^-$ GP (SDD$^-$ GP-ARD).

	Syn. 1		Syn. 2		M. L. CO_2	
——	RMSE	MAE	RMSE	MAE	RMSE	MAE
SDD GP	1.267	0.871	6.155	2.049	114.708	8.362
SDD$^-$ GP	0.606	0.648	6.858	2.205	3.948e+04	184.845
Full GP	0.079	0.203	**5.466**	**2.013**	103.459	8.298
Sparse GP	0.579	0.641	19.876	3.718	148.698	11.168
SDD GP-ARD	**0.057**	**0.194**	29.806	4.520	144.676	10.577
SDD$^-$ GP-ARD	0.057	0.194	29.806	4.520	1.325e+05	348.360
Full GP-ARD	0.057	0.195	29.805	4.520	**13.563**	**3.063**
Sparse GP-ARD	0.065	0.211	30.684	4.926	14.540	3.173

NO_2 datasets. It also achieves the smallest MAE on the Sheffield temperature dataset. Otherwise, the SDD$^-$ GP slightly performs better than the SDD GP. When it comes to the median RMSE and MAE, the SDD GP outperforms the full GP variants except on the Sheffield temperature and NO_2 datasets.

Table 6: The performance comparison among different methods and kernels on Sheffield and Peshawar datasets. We take the minimum RMSE and MAE for the SDD GP (SDD GP-ARD) and SDD$^-$ GP (SDD$^-$ GP-ARD).

	S. Temp (℃)		P. Temp (℃)		S. NO_2 ($\mu g/m^3$)		P. NO_2 ($\mu g/m^3$)	
——	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SDD GP	0.125	**0.270**	**0.965**	**0.819**	**0.404**	**0.394**	0.077	0.212
SDD$^-$ GP	**0.115**	0.270	0.970	0.827	0.413	0.395	**0.075**	**0.209**
Full GP	0.329	0.482	1.416	1.005	1.0472	0.779	0.664	0.578
Sparse GP	0.661	0.695	1.485	1.025	1.074	0.762	0.675	0.652
SDD GP-ARD	0.473	0.586	1.021	0.880	0.427	0.436	0.712	0.722
SDD$^-$ GP-ARD	0.484	0.599	1.022	0.881	0.431	0.446	0.713	0.721
Full GP-ARD	0.494	0.604	1.008	0.870	0.431	0.446	0.697	0.731
Sparse GP-ARD	0.689	0.701	1.005	0.868	0.503	0.519	0.726	0.755

In general, as m increases, the performance of the developed approach reaches the performance of a full GP variant with the same kernel. When we adopt the minimum and median RMSE and MAE as metrics, we have shown that the performance of the proposed SDD GP (SDD GP-ARD) and the full GP (full GP-ARD) alternates. Fig. 3 and 5 also show that when m is small, the full GP variants outperform the proposed approach with the same kernels in general. This is because when m is too small, entries of the 'residual' matrix $\tilde{\mathbf{A}}$ are still significantly big, forcing it to be SDD would not improve the approximation accuracy of the covariance matrix.

In addition to the aforementioned validation results, we also test the pro-

Table 7: The performance comparison among different methods and kernels on Sheffield and Peshawar datasets. We take the median RMSE and MAE for the SDD GP (SDD GP-ARD) and SDD$^-$ GP (SDD$^-$ GP-ARD).

	S. Temp (°C)		P. Temp (°C)		S. NO_2 ($\mu g/m^3$)		P. NO_2 ($\mu g/m^3$)	
——	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SDD GP	0.629	0.664	**0.985**	**0.839**	0.603	0.532	**0.148**	**0.314**
SDD$^-$ GP	0.647	0.694	1.303	0.963	0.902	0.634	0.148	0.314
Full GP	**0.329**	**0.482**	1.416	1.005	1.047	0.779	0.664	0.578
Sparse GP	0.661	0.695	1.485	1.025	1.074	0.762	0.675	0.652
SDD GP-ARD	0.489	0.602	1.022	0.880	0.435	0.450	0.715	0.724
SDD$^-$ GP-ARD	0.490	0.603	1.022	0.880	0.697	0.548	0.715	0.724
Full GP-ARD	0.494	0.604	1.008	0.870	**0.431**	**0.446**	0.697	0.731
Sparse GP-ARD	0.689	0.701	1.005	0.868	0.503	0.519	0.726	0.755

posed approach with kernel setting 1 and 2 on data with outliers, in comparison with the corresponding full GP variants. The results are shown in Figs. 6 and 7, respectively. The outliers are generated by adding noises with prominent variances to both (33) and (34). To be specific, we set the variance of v_1 to 0.5 and the variance of v_2 to 1.65, which generates noise of comparable scales with the real value of (33) and (34).

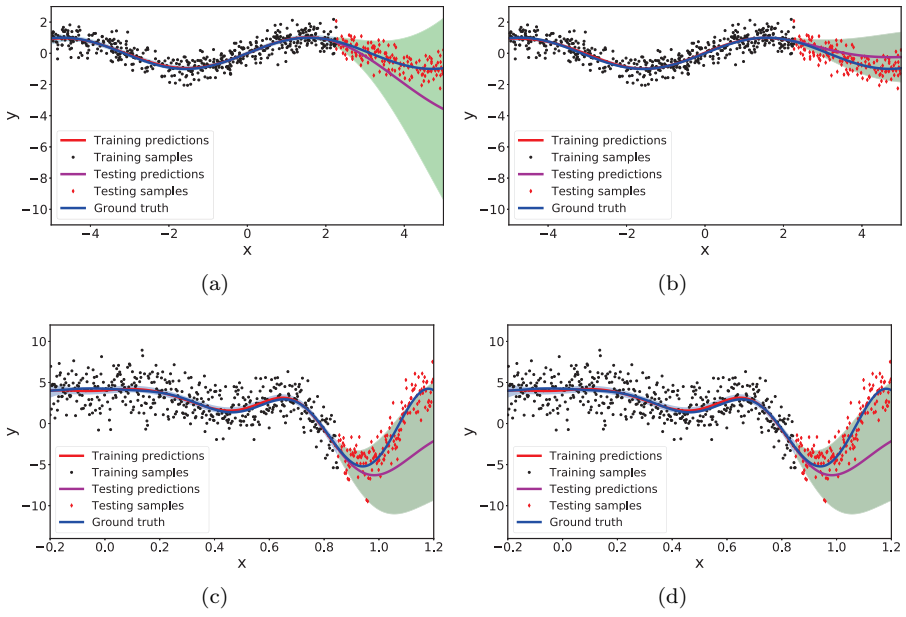

Figure 6: Impacts of outliers on (33) and (34), with kernel setting 1 in Table 2: (a) Full GP with $v_1 \sim \mathcal{N}(0, 0.50)$; (b) SDD with $v_1 \sim \mathcal{N}(0, 0.50)$; (c) Full GP with $v_2 \sim \mathcal{N}(0, 1.65)$; (d) SDD with $v_2 \sim \mathcal{N}(0, 1.65)$. The shaded areas indicate the 95% confidence interval.

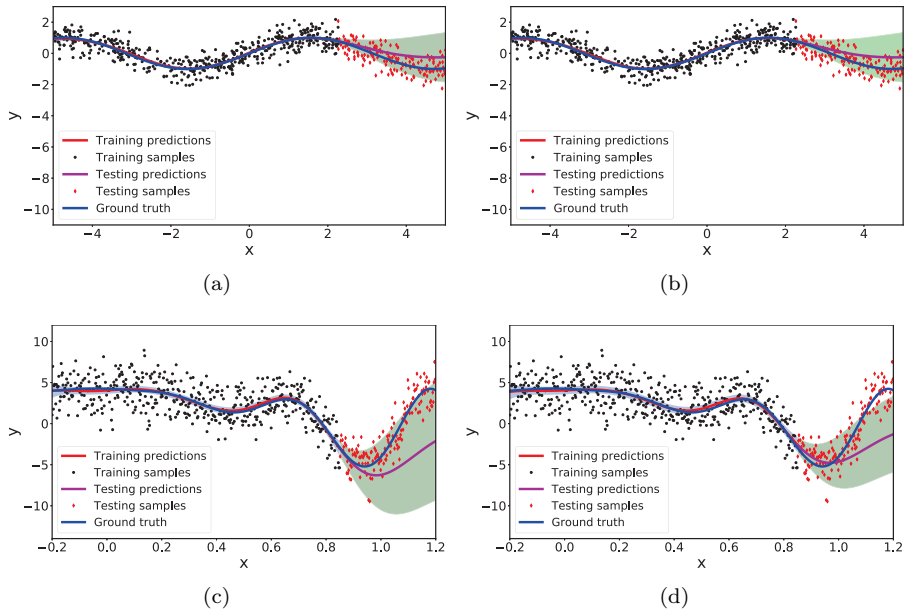

Figure 7: Impacts of outliers on (33) and (34), with kernel setting 2 in Table 2: (a) Full GP-ARD with $v_1 \sim \mathcal{N}(0, 0.50)$; (b) SDD GP-ARD with $v_1 \sim \mathcal{N}(0, 0.50)$; (c) Full GP-ARD with $v_2 \sim \mathcal{N}(0, 1.65)$; (d) SDD GP-ARD with $v_2 \sim \mathcal{N}(0, 1.65)$. The shaded areas indicate the 95% confidence interval.

We can see from Figs. 6 and 7 that the proposed approach is still able to produce comparable results as full GP variants with the same kernel despite the impact of outliers. It is worth mentioning that just like noises, kernels could affect the performance of GP models as well. This can be observed from the difference between Fig. 6(a) and Fig. 7(a), as well as from the difference between Fig. 6(d) and Fig. 7(d).

We then increase the number of samples and test the proposed approach over larger scale datasets compared with aforementioned settings. To be precise, we set the number of samples to 6,000 and 12,000 respectively for both (33) and (34). The proposed approach with kernel setting 1 and 2 are applied to process the data separately and the results are given in Fig. 8 and Fig. 9, respectively. Figs. 8(a) and 8(b) are the results from SDD GP with 6,000 and 12,000 samples from (33). Fig. 8(c) and 8(d) separately show the results of SDD GP with 6,000 and 12,000 samples from (34). The corresponding results obtained by using the

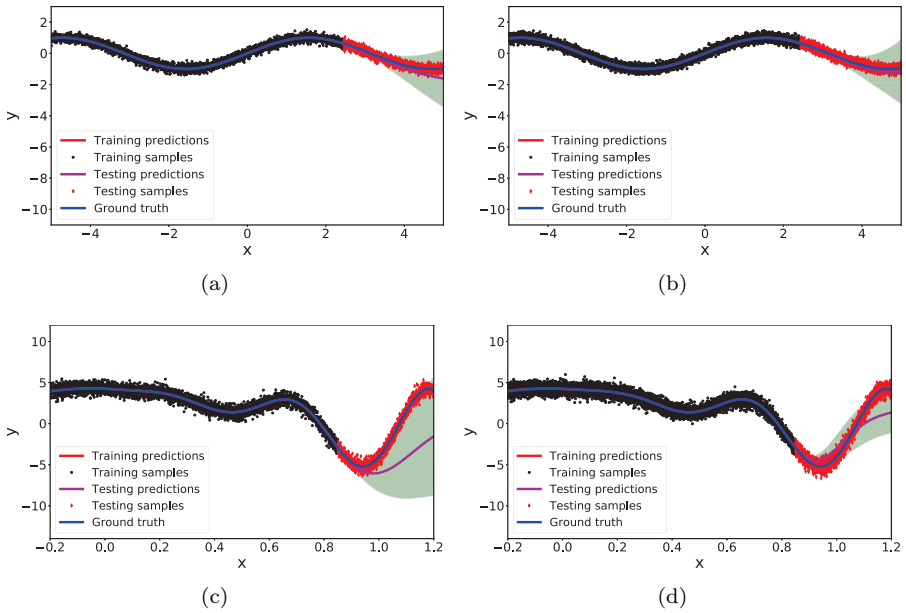

Figure 8: Performance of the proposed approach on large scale datasets, with kernel setting 1 in Table 2 used: (a) SDD GP with 6,000 samples from (33); (b) SDD GP with 12,000 samples from (33); (c) SDD GP with 6,000 samples from (34); (d) SDD GP with 12,000 samples from (34). $v_1 \sim \mathcal{N}(0, 0.15)$, $v_2 \sim \mathcal{N}(0, 0.45)$. The shaded areas indicate the 95% confidence interval.

SDD GP-ARD to process data from (33) are given in Figs. 9(a) and 9(b), whereas Figs. 9(c) and 9(c) show the results achieved by using SDD GP-ARD to process data from (34). As the number of samples increases, the full GP variants become slow for the new datasets, while the proposed approach still achieves comparable results more efficiently than the full GP variants as shown in Table 8. It is worth mentioning that GP variants using the kernel setting 2 are generally more efficient than those using kernel setting 1.

Table 8: Efficiency comparison of SDD GP (SDD GP-ARD) and full GP (full GP-ARD) with different kernel settings

	Syn. 1			Syn. 2		
Sample Number	635	6,000	12,000	635	6,000	12,000
SDD GP Time (s)	2.3	50.6	673.8	3.2	60.6	590.4
Full GP Time (s)	5.1	123.6	1566.2	6.5	126.0	1476.4
SDD GP-ARD Time (s)	1.6	38.9	200.2	1.5	52.2	239.2
Full GP-ARD Time (s)	3.4	83.2	454.5	3.5	108.6	509.5

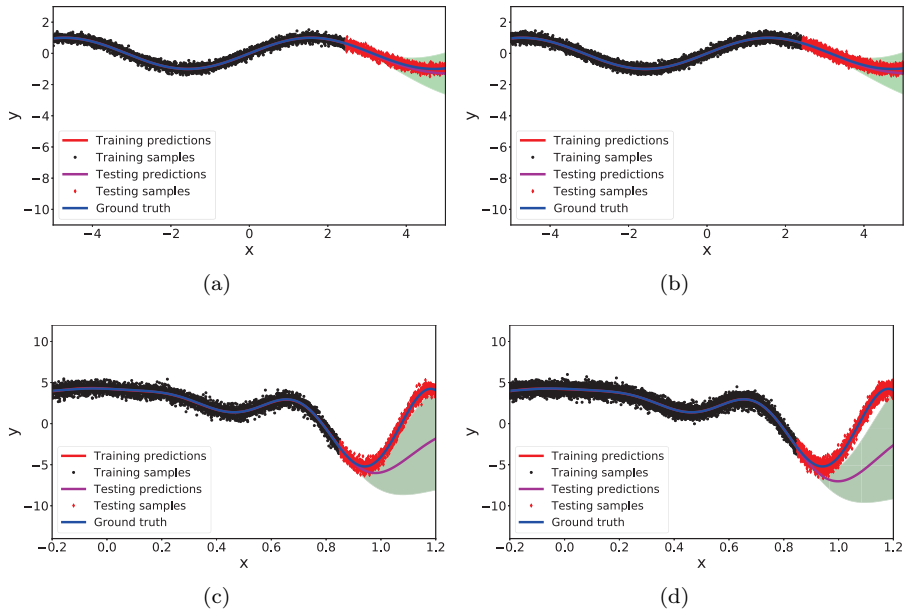

Figure 9: Performance of the proposed approach on large scale datasets, with kernel setting 2 in Table 2 used: (a) SDD GP-ARD with 6,000 samples from (33); (b) SDD GP-ARD with 12,000 samples from (33); (c) SDD GP-ARD with 6,000 samples from (34); (d) SDD GP-ARD with 12,000 samples from (34). $v_1 \sim \mathcal{N}(0, 0.15)$, $v_2 \sim \mathcal{N}(0, 0.45)$. The shaded areas indicate the 95% confidence interval.

6. Conclusion

In this paper, we propose a new kernel matrix approximation approach that considers the residual matrix and compares it with traditional kernel approximation methods. The key novelty of the paper stems from considering the residual matrix in covariance matrix approximation. The residual matrix is approximated by a symmetric diagonally dominant matrix whose inverse can be easily approached by the Neumann series. A new Gaussian process variant denoted as SDD GP is hence built upon the proposed approximation method, which achieves comparable or better performance compared with full GP on both synthetic datasets and real air quality datasets, with lower computational complexity. Furthermore, the SE-ARD kernel is applied in addition to the composition kernels, to demonstrate the generality of the proposed approach.

We have applied the Mendoza-Raydan-Tarazaga projection to help us achieve

a symmetric diagonally dominant projection of the residual matrix. We observed that the projection algorithm can affect the efficiency of the proposed approach, despite good prediction performance. Hence, we shall continue with the efficiency improvement of the proposed approach in the future.

Acknowledgements. This paper was supported by the National Science Foundation of China (Grant No. 61703387). We are also grateful to UK EPSRC for funding this work through EP/T013265/1 project NSF-EPSRC:ShiRAS. Towards Safe and Reliable Autonomy in Sensor Driven Systems. This work was also supported by the USA National Science Foundation under Grant NSF ECCS 1903466. We are also grateful to the Global Challenges Research Funds (QR GCRF - Pump priming awards (Round 2), entitled: "Collaborating with North Pakistan for monitoring and reducing the air pollution (X/160978))".

References

[1] C. Magnant, A. Giremus, E. Grivel, L. Ratton, B. Joseph, Bayesian non-parametric methods for dynamic state-noise covariance matrix estimation: Application to target tracking, Signal Processing 127 (2016) 135–150.

[2] E. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, in: Proceedings of Advances in Neural Information Processing Systems, 2006, pp. 1257–1264.

[3] H. Liu, Y.-S. Ong, X. Shen, J. Cai, When Gaussian process meets big data: A review of scalable GPs, IEEE Transactions on Neural Networks and Learning Systems (2020) 1–19.

[4] J. Quiñonero-Candela, C. E. Rasmussen, A unifying view of sparse approximate Gaussian process regression, Journal of Machine Learning Research 6 (12) (2005) 1939–1959.

[5] P. Wang, Y. Kim, L. Vaci, H. Yang, L. Mihaylova, Short-term traffic prediction with vicinity Gaussian process in the presence of missing data, in:

Proceedings from the Sensor Data Fusion: Trends, Solutions, Applications (SDF), IEEE, 2018, pp. 1–6.

[6] C. K. Williams, C. E. Rasmussen, Gaussian Processes for Machine Learning, MIT Press Cambridge, MA, 2006.

[7] H. Wang, X. Gao, K. Zhang, J. Li, Fast single image super-resolution using sparse Gaussian process regression, Signal Processing 134 (2017) 52–62.

[8] C. K. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: Proc. from Advances in Neural Information Processing Systems, 2001, pp. 682–688.

[9] F. Zhu, P. Honeine, Online kernel nonnegative matrix factorization, Signal Processing 131 (2017) 143–153.

[10] R. B. Abdallah, A. Breloy, M. N. El Korso, D. Lautru, Bayesian signal subspace estimation with compound gaussian sources, Signal Processing 167 (2020) 107310.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical Recipes in C, Cambridge University Press, Cambridge, 1988.

[12] D. Zhu, B. Li, P. Liang, On the matrix inversion approximation based on Neumann series in massive MIMO systems, in: Proceedings from the 2015 IEEE International Conference on Communications (ICC), IEEE, 2015, pp. 1763–1769.

[13] M. Wu, B. Yin, G. Wang, C. Dick, J. R. Cavallaro, C. Studer, Large-scale MIMO detection for 3GPP LTE: Algorithms and FPGA implementations, IEEE Journal of Selected Topics in Signal Processing 8 (5) (2014) 916–929.

[14] S. Ambikasaran, D. Foreman-Mackey, L. Greengard, D. W. Hogg, M. O'Neil, Fast direct methods for Gaussian processes, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (2) (2015) 252–265.

[15] K. Zhang, I. W. Tsang, J. T. Kwok, Improved nyström low-rank approximation and error analysis, in: Proceedings of the 25th international conference on Machine learning, 2008, pp. 1232–1239.

[16] L. He, N. Ray, H. Zhang, Error bound of nyström-approximated ncut eigenvectors and its application to training size selection, Neurocomputing 239 (2017) 130–142.

[17] M. L. Stein, Limitations on low rank approximations for covariance matrices of spatial data, Spatial Statistics 8 (2014) 1–19.

[18] D. M. Blei, A. Kucukelbir, J. D. McAuliffe, Variational inference: A review for statisticians, Journal of the American Statistical Association 112 (518) (2017) 859–877.

[19] D. R. Burt, C. E. Rasmussen, M. Van Der Wilk, Rates of convergence for sparse variational Gaussian process regression, arXiv preprint arXiv:1903.03571 (2019).

[20] M. Ouimet, Y. Bengio, Greedy spectral embedding, in: Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, 2005, pp. 253–260.

[21] C. Musco, C. Musco, Recursive sampling for the nyström method, in: Advances in Neural Information Processing Systems, 2017, pp. 3833–3845.

[22] Y. Ding, R. Kondor, J. Eskreis-Winkler, Multiresolution kernel approximation for Gaussian process regression, in: Advances in Neural Information Processing Systems, 2017, pp. 3740–3748.

[23] J. Li, A. Nehorai, Gaussian mixture learning via adaptive hierarchical clustering, Signal Processing 150 (2018) 116–121.

[24] X. Yao, C. Zhao, Kernel-band-projection algorithm for anomaly detection in hyperspectral imagery, in: Proceedings of the 14th IEEE International Conference on Signal Processing (ICSP), IEEE, 2018, pp. 300–303.

[25] M. Titsias, Variational learning of inducing variables in sparse Gaussian processes, in: Proceedings of Artificial Intelligence and Statistics, 2009, pp. 567–574.

[26] Z. T. Ke, L. Xue, F. Yang, Diagonally-dominant principal component analysis, Journal of Computational and Graphical Statistics (accepted) (2020) 1–16.

[27] M. Mendoza, M. Raydan, P. Tarazaga, Computing the nearest diagonally dominant matrix, Numerical linear algebra with applications 5 (6) (1998) 461–474.