SPARSE INVERSE GAUSSIAN PROCESS REGRESSION WITH APPLICATION TO CLIMATE NETWORK DISCOVERY

KAMALIKA DAS* AND ASHOK N. SRIVASTAVA**

ABSTRACT. Regression problems on massive data sets are ubiquitous in many application domains including the Internet, earth and space sciences, and finances. Gaussian Process regression is a popular technique for modeling the input-output relations of a set of variables under the assumption that the weight vector has a Gaussian prior. However, it is challenging to apply Gaussian Process regression to large data sets since prediction based on the learned model requires inversion of an order n kernel matrix. Approximate solutions for sparse Gaussian Processes have been proposed for sparse problems. However, in almost all cases, these solution techniques are agnostic to the input domain and do not preserve the similarity structure in the data. As a result, although these solutions sometimes provide excellent accuracy, the models do not have interpretability. Such interpretable sparsity patterns are very important for many applications. We propose a new technique for sparse Gaussian Process regression that allows us to compute a parsimonious model while preserving the interpretability of the sparsity structure in the data. We discuss how the inverse kernel matrix used in Gaussian Process prediction gives valuable domain information and then adapt the inverse covariance estimation from Gaussian graphical models to estimate the Gaussian kernel. We solve the optimization problem using the alternating direction method of multipliers that is amenable to parallel computation. We demonstrate the performance of our method in terms of accuracy, scalability and interpretability on a climate data set.

1. INTRODUCTION

In many application domains, it is important to predict the value of one feature based on certain other measured features. For example, in the Earth Sciences, predicting the precipitation at one location given the humidity, sea surface temperature, cloud cover, and other related factors is an important problem in climate modeling. For such problems, simple linear regression based on minimization of the mean squared error between the true and predicted values can be used for modeling the relationship between the input and the target features. In decision support systems which use these predictive algorithms, a prediction with low confidence may be treated differently than if the same prediction was given with high-confidence. Thus, while the predicted value from the regression function is clearly important, the confidence in the prediction is equally important. A simple model such as linear regression does not provide us with that information. Also, models like linear regression, in spite of being easy to fit and being highly scalable, fail to capture nonlinear relationships in the data. Gaussian Process regression (GPR) is one regression model that can capture nonlinear relationships and outputs a distribution of the prediction where the variance of the predicted distribution acts as a measure of confidence in the prediction. Moreover, the inverse kernel (or covariance) matrix has many interesting properties along the gaussian graphical model perspective, that can be exploited for better understanding relationships within the training examples. Depending on the nature of the data, these relationships can indicate dependencies (causalities) for certain models.

However, predictions based on GPR method, requires inversion of a kernel (or covariance) matrix of size $n \times n$, where n is the number of training instances. This kernel inversion becomes a bottleneck for very large datasets. Most of the existing methods for efficient computation in GPR involve numerical approximation techniques that exploit data sparsity. While this does speed up

*SGT Inc., NASA Ames Research Center, Kamalika.Das@nasa.gov, **NASA Ames Research Center, ashok.n.srivastava@nasa.gov
.

Published in Proceedings of the Conference on Intelligent Data Understanding, 2011.

GPR computations, one serious drawback of these approximations is that the resulting GPR model loses interpretability. Even if we get reasonably accurate predictions, we fail to unearth significant connections between the training points or identify the most influential training points for a specific set of test points.

In this paper we propose a sparse GPR algorithm which not only scales to very large datasets but also allows us to construct a complete yet sparse inverse covariance matrix, thereby facilitating interpretability. The method proposed in this paper induces sparsity by introducing a regularizer in a pseudo negative log likelihood objective used for covariance selection. This forces the algorithm to seek a parsimonious model for GPR prediction having excellent interpretability. One of the highlights of the solution technique used in this paper is a completely parallelizable framework for solving the inverse covariance estimation problem using the alternating direction method of multipliers (ADMM) that allows us to exploit modern parallel and multi-core architectures. This also addresses the situation where the entire covariance matrix cannot be loaded into memory due to size limitations.

The rest of the paper is organized as follows. In the next section (Section 2) we present some background material related to GPR and some existing methods of solving the GPR problems. In Section 3 we discuss the equivalence between inverse kernel and covariance matrices. Next we present our new sparse inverse covariance matrix using ADMM technique (Section 4). Experimental results are discussed in Section 5. We conclude the paper in Section 6.

2. Background: Gaussian Process Regression

Since this paper proposes a technique of model fitting using Gaussian Process regression, we start with a brief review of it here. Rasmussen and Williams [15] provide an excellent introduction on this subject. Gaussian Process regression is a generalization of standard linear regression. If $\mathbf{X}$ is the training data set having n multidimensional observations (rows) $\mathbf{x}_1, \ldots, \mathbf{x}_n$, with each $\mathbf{x}_i \in \mathbb{R}^D$ and the corresponding target is represented by a $n \times 1$ vector $\mathbf{y}$, then the standard linear regression model is:

$$f(\mathbf{x}) = \mathbf{x}\mathbf{w}^T, \qquad y = f(\mathbf{x}) + \epsilon$$

where $\mathbf{w}$ is a D-dimensional weight vector of parameters and ϵ is additive Gaussian noise such that $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Assuming that we choose the prior distribution of the weights to be Gaussian with mean zero and covariance Σ_p, the posterior distribution of the weights, following Bayesian inferencing techniques, can be written as:

$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) \sim \mathcal{N}\left(\frac{1}{\sigma^2} A^{-1}\mathbf{X}^T y, A^{-1}\right)$$

where $A = \sigma^{-2}\mathbf{X}^T\mathbf{X} + \Sigma_p^{-1}$. Given the posterior and the likelihood, the predictive distribution of a test input $\mathbf{x}^*$ is obtained by averaging over all possible models ($\mathbf{w}$) to obtain:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{y}) \sim \mathcal{N}\left(\frac{1}{\sigma^2}\mathbf{x}^* A^{-1} X^T \mathbf{y}, \mathbf{x}^* A^{-1} \mathbf{x}^{*T}\right)$$

Using a kernel (covariance) function $k(\mathbf{x}_i, \mathbf{x}_j)$ in place of a mapping from input space to an N-dimensional space, and applying some algebraic manipulations, we can write the predictive mean and variance of the posterior distribution as

$$\widehat{\mathbf{y}}^* = K^*(\sigma^2 I + K)^{-1}\mathbf{y} \tag{1}$$

$$C = K^{**} - K^*(\sigma^2 I + K)^{-1} K^{*T} \tag{2}$$

where the ij^{th} entry of K is $k(\mathbf{x}_i, \mathbf{x}_j)$ and K^* and K^{**} are similarly the cross covariance matrices involving the test point $\mathbf{x}^*$. Equations 1 and 2 pose significant computational challenge due to the requirement of inverting the covariance matrix K of size n^2. If the number of observations n is large, the $O(n^3)$ operation can be a bottleneck in the process of using Gaussian Process regression.

In the next section, we discuss several techniques that have been proposed in the literature for approximating the inverse matrix for large datasets.

2.1. **Existing methods for efficient GP computation.** Approximations are introduced in the Gaussian Process literature for either finding closed-form expressions for intractable posterior distributions or for gaining computational advantage for large data sets. Here we are interested in the second goal and, therefore, briefly discuss the existing research in this area. Smola and Bartlett [16] describe a sparse greedy method that does not require evaluating the full covariance matrix K and finds an approximation to the maximum aposteriori estimate by selecting an 'active' subset of columns of K by solving an expensive optimization problem. The running time of the numerical approximation is reduced from $O(n^3)$ to $O(nm^2)$ where m ($m \ll n$) is the rank of the matrix approximation.

A related approach of low rank matrix approximation called the subset of regressors method [21] involves selecting the principal sub-matrix of the unperturbed covariance matrix K by matrix factorization. Though this method has been found to be numerically unstable, recent research by Foster *et al.* [8] has shown that if we use partial Cholesky decomposition to factorize the covariance matrix and perturb the low rank factor such that independent rows and columns form the principal sub-matrix, then the approximation we get is numerically stable. The authors report excellent accuracy using their approximation calculations when the rank of the reduced matrix is a small factor (5) times the rank of the original data matrix X.

The generalized Bayesian committee machine [20] is another approach for reducing the computational complexity of any kernel-based regression technique, by dividing the data arbitrarily into M almost equal sized partitions, training a different estimator on each partition, and combining the estimates given by the different estimators using the inverse of the variance to ensure that least certain predictions are given the smallest weights in the final prediction. This method allows us to choose M to be equal to $K\alpha$ so that it becomes linear in K in computational complexity. The Bayesian Committee Machine weights the training data based on the test points using a block diagonal approximation and, therefore, the model needs to be retrained every time a new test set comes in. A related method recently proposed by Das and Srivastava [4] works for multimodal data. It partitions the input space into multiple clusters, with each one corresponding to one mode of the data distribution. Then, each cluster is modeled using a normal distribution and all points which are not modeled by any of the normal distributions are grouped using a separate cluster. Each cluster learns a separate GP model and a weighted sum based prediction is used for the gating.

A recent development is the ℓ_1 penalized GPR method (GPLasso) introduced by Yan and Qi [22] in which the authors explore sparsity in the output rather than the input. They propose a GPR technique that minimizes the Kullback-Leibler divergence between the posterior distributions of the exact and the sparse solutions using a ℓ_1 penalty on the optimization. They pose this problem as a LASSO optimization [19] and solve a rank reduced approximate version of this using the Least Angle Regression (LARS) method [7]. The authors present this work as a pseudo output analogy of the work by Snelson *et al.* [17]. Quiñonero-Candela and Rasmussen [14] provide a unifying view of all sparse approximation techniques for Gaussian Process regression by analyzing the posterior and reinterpreting each algorithm as an exact inferencing method using approximate priors.

All the methods discussed in this section apply some form of numerical approximation technique to reduce the rank of the kernel matrix for efficient matrix inversion. As a result, they often lose model interpretability — a value at any position of the reduced rank inverted matrix cannot be traced back to any cell of the original kernel. In many domains, however understanding the sparsity structure is important. For example, in Earth Sciences, it is not only important to get good predictions from the GPR model, but it is also important to understand how different geographical regions are connected and how these locations influence one another. Unfortunately, none of the efficient GPR techniques allow this. Our proposed technique in the next section not only learns a sparse GP model but also allows domain scientists to draw conclusions about the sparsity structure by studying the inverse covariance matrix.

3. SPI-GP:Sparse Gaussian Process using inverse covariance estimation

Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ be a set of multi-dimensional gaussian observations such that

$$\mathbf{x}_i \sim \mathcal{N}(\mu, \Sigma) \in \mathbb{R}^d$$

where $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d \times d}$ are the mean and covariance matrices. While the mean μ measures the center of the distribution, the covariance matrix Σ measures the pairwise (linear) relationship between the variables. It is well known that a value of 0 at any cell of Σ implies independence of the observations:

$$\Sigma_{i,j} = 0 \Rightarrow P(\mathbf{x}_i \mathbf{x}_j) = 0$$

which means $\mathbf{x}_i$ and $\mathbf{x}_j$ are independent. In many cases, we may be interested in studying how two variables influence each other when the information about the other variables are taken into consideration. One way of doing this is by studying the inverse covariance matrix, also known as the concentration matrix or precision matrix denoted by Σ^{-1}. Unlike Σ, a value of 0 in any cell of Σ^{-1} implies conditional independence among those variables [1]. For example, $\mathbf{x}_i$ and $\mathbf{x}_j$ are conditionally independent, given all the other variables, if Σ^{-1}=0. Mathematically,

$$\Sigma^{-1}_{i,j} = 0 \Rightarrow P(\mathbf{x}_i \mathbf{x}_j | \mathbf{x}_{-i,-j}) = 0$$

where $\mathbf{x}_{-i,-j}$ denotes all the variables other than $\mathbf{x}_i$ and $\mathbf{x}_j$. Note that independence of elements implies conditional independence but not vice-versa *i.e.* a value of 0 at any cell of Σ implies that the corresponding location of Σ^{-1} is also 0; but a non-zero value at any cell of Σ matrix does not imply that the corresponding cell of Σ^{-1} will also be non-zero. The reason for studying Σ^{-1} rather than Σ, is for many gaussian distributed variables, there is more sparsity in the inverse covariance matrix than in the covariance matrix and this sparsity reveals interesting data relationships. It has been shown in [9], that inverting a covariance matrix (with the additional assumption that the inverse is sparse) is equivalent to learning a graphical model, where each node in the model corresponds to a feature and the absence of an edge between any two signifies that those features are conditionally independent.

In the case of GPR, the kernel matrix between the observations (see Eqn. 1 and 2) can be viewed as a covariance matrix among the function outputs. Formally, a gaussian process is defined as a collection of random variables, any finite number of which is jointly gaussian. Hence, it is a distribution over functions, completely specified by its mean function and covariance function as,

$$f(\mathbf{x}_i) \sim GP(m(\mathbf{x}_i), k(\mathbf{x}_i, \mathbf{x}_j))$$

where $m(\mathbf{x}_i) = E[f(\mathbf{x}_i)]$ and $k(\mathbf{x}_i, \mathbf{x}_j) = E[f(\mathbf{x}_i) - m(\mathbf{x}_i)][f(\mathbf{x}_j) - m(\mathbf{x}_j)]$ are the mean function and covariance function of some real process $f(\mathbf{x}_i)$. Note that $f(\mathbf{x}_i)$ are random variables and GP fits a distribution over all possible $f(\mathbf{x}_i)$. In our case since $f(\mathbf{x}_i)$'s are linear functions $f(\mathbf{x}_i) = \mathbf{x}_i \mathbf{w}^T$, the mean and covariance of GP can be stated as,

$$m(\mathbf{x}_i) = E[f(\mathbf{x}_i)] = \mathbf{x}_i E[\mathbf{w}^T] = 0$$

$$k(\mathbf{x}_i, \mathbf{x}_j) = E[f(\mathbf{x}_i) f(\mathbf{x}_j)] = \mathbf{x}_i E[\mathbf{w}^T \mathbf{w}] \mathbf{x}_i^T = \mathbf{x}_i \Sigma_p \mathbf{x}_i^T$$

where $\mathbf{w} \sim N(0, \Sigma_p)$ denotes the prior distribution of the weights. The covariance function k, also known as the kernel function specifies the covariance between a pair of random variables

$$\text{cov}(f(\mathbf{x}_i), f(\mathbf{x}_j)) = E[f(\mathbf{x}_i) f(\mathbf{x}_j)] = k(\mathbf{x}_i, \mathbf{x}_j)$$

Therefore, a kernel function computed over the pairwise input points is equivalent to a covariance between the outputs. There are several choices of the kernel functions available. In this paper we have used the widely used gaussian radial basis function (rbf) kernel:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

where σ is known as the bandwidth parameter which is typically learned from the data.

In many GPR applications, it is not only important to get good prediction accuracy, but also understand the model. For example, in Earth Sciences teleconnections [11] reveal important symmetric and sometimes causal relationships among different events observed in geographically distant locations and can be studied by exploiting sparsity in the inverse kernel in GPR. Another possible application area is the study of climate networks [18]. Fig. 1 (left) shows the observed precipitation data of the world overlaid on a 360×720 grid. Figs. 1 (center and right) show a kernel or similarity matrix generated from the data and the corresponding inverse covariance matrix. Each cell in the kernel (except the diagonal) denotes the similarity between the precipitation values of a grid location (lower resolution). The highlighted row and column correspond to the location marked in white on the world map. In this paper we are interested in studying the sparsity pattern of the inverse covariance matrix, with the information that sparsity patterns in the inverse covariance matrix leads to conditional independence among the locations of interest.

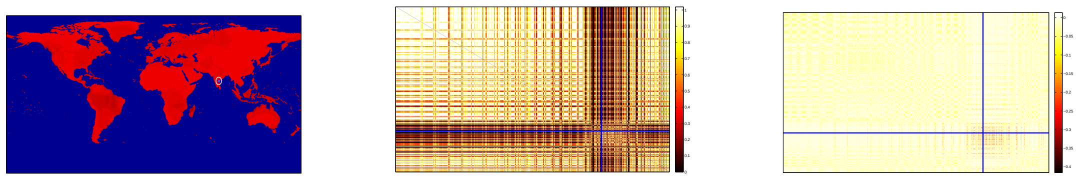


Figure 1. Precipitation data of the world map (top figure). Note that the data is only available for land (the ocean locations have fill values of -9999). The figure in the center shows a kernel in which similarity is computed between every pair of locations from the precipitation data. Note the location marked with a circle on the left figure corresponds to the row and column in blue on the center and right figure. The right figure shows the inverse kernel matrix.

4. Sparse covariance selection

There exist several techniques in the literature for solving the inverse covariance estimation problem also known as the covariance selection problem.

Given a dataset containing d features, Meinshausen *et al.* [13] infers the graphical model (and therefore the inverse covariance matrix) by taking one variable at a time and then finding all the connections of that variable with all of the other ones. For each variable d_i in the dataset, the method constructs a lasso regression problem by taking all the other variables as inputs and d_i as the target with an additional sparsity constraint on the solution weights. The non-zero entries of the weight vector signifies a connection between that feature and the target d_i. To deal with inconsistencies among the connections, the authors have proposed two schemes: (1) in the **AND** technique, an edge is established in the graphical model between any two features d_i and d_j iff both d_i and d_j have non-zero entries in the weight vector when they are each used as target in different lasso problems, and (2) in the **OR** scheme, an edge is established if either d_i or d_j has a non-zero weight when the other is taken as the target. One serious drawback of this method is the number of independent lasso problems increases linearly with the size of the feature space.

Banerjee *et al.* [1] propose a different solution to the inverse covariance selection problem. They show that based on Dempster's theory [5], estimating the inverse covariance matrix is equivalent to minimizing the pesudo negative log likelihood. The objective function takes the form:

$$\mathbf{Tr}(KS) - \log det(S)$$

where K is the empirical covariance (or kernel) matrix and S is the desired inverse of K i.e. $S = K^{-1}$, $\mathbf{Tr}(\cdot)$ is the trace of a matrix, and $det(\cdot)$ is the matrix determinant. Solution to the above equation is stable when an additional sparsity constraint is imposed on the inverse, *i.e.*

$$\mathbf{Tr}(KS) - \log det(S) + \lambda \|S\|$$

where λ controls the degree of sparsity. This is a convex optimization problem and in order to solve this, the authors propose a block-wise interior point algorithm.

Friedman *et al.* [9] generalizes both these papers and present a very efficient algorithm based on the lasso technique. Their objective function is the same as used by Banerjee *et al.* [1] *i.e.* they try to maximize the log likelihood of the model with the additional sparsity constraint. They show that the solution proposed by Meinshausen [13] is an approximation of the log likelihood estimate proposed by Banerjee *et al.* [1]. They propose a new algorithm based on coordinate descent to solve the same trace minimization problem. This algorithm is based on recursively solving lasso subproblems for each variable until convergence. The authors note that this new algorithm is at least 50 to 4000 times faster than existing techniques and therefore scales to much larger data sets.

However, there is one drawback common to all these optimization techniques. All these techniques assume that the data can be loaded in computer memory for the analysis. Unfortunately, in applications such as Earth Sciences, most datasets are massive — they contain millions of observations (locations) and therefore constructing a full covariance matrix in memory is itself impossible, leaving aside the computational power necessary to run these optimization techniques for inverse estimation. To solve the large scale inverse covariance estimation problems which do not fit into the memory of one machine, in this paper we propose our SPI-GP method which works by distributing the workload among a network of machines. The technique we follow is based on the method of Alternating Direction Method of Multipliers (ADMM) which is a distributable algorithm for solving very large convex optimization problems. We give a brief overview of ADMM technique in the next section.

4.1. **Alternating Direction Method of Multipliers for convex problems.** Alternating Direction Method of Multipliers (ADMM) [10][6][2] is a decomposition algorithm for solving separable convex optimization problems of the form:

$$\min \quad G_1(x) + G_2(y) \quad \text{subject to} \quad Ax - y = 0, \quad x \in \mathbb{R}^n, \quad y \in \mathbb{R}^m$$

where $A \in \mathbb{R}^{m \times n}$ and G_1 and G_2 are convex functions. The algorithm derivation is as follows. First, the augmented Lagrangian is formed:

$$L_\rho(x, y, z) = G_1(x) + G_2(y) + z^T(Ax - y) + \rho/2 \|Ax - y\|_2^2$$

where ρ is a positive constant known as the penalty parameter. ADMM iterations can then be written as:

$$x^{t+1} = \min_x \left\{ G_1(x) + z^{tT} Ax + \rho/2 \left\|Ax - y^t\right\|_2^2 \right\} \quad (3)$$

$$y^{t+1} = \min_y \left\{ G_2(y) - z^{tT} y + \rho/2 \left\|Ax^{t+1} - y\right\|_2^2 \right\} \quad (4)$$

$$z^{t+1} = z^t + \rho \left(Ax^{t+1} - y^{t+1} \right) \quad (5)$$

This is an iterative technique where t is the iteration counter, and the initial vectors y^0 and z^0 can be chosen arbitrarily. ADMM can be written in a different form (known as the scaled form) by combining the linear and quadratic terms of the Lagrangian:

$$z^T(Ax - y) + \rho/2 \|(Ax - y)\|_2^2 = \rho/2 \|(Ax - y) + (1/\rho)z\|_2^2 - 1/(2\rho) \|z\|_2^2$$

Now scaling the dual variable $p = (1/\rho)z$, the iterations of ADMM become:

$$x^{t+1} = \min_x \left\{ G_1(x) + \rho/2 \left\| Ax - y^t + p^t \right\|_2^2 \right\} \tag{6}$$

$$y^{t+1} = \min_y \left\{ G_2(y) + \rho/2 \left\| Ax^{t+1} - y + p^t \right\|_2^2 \right\} \tag{7}$$

$$p^{t+1} = p^t + \rho \left(Ax^{t+1} - y^{t+1} \right) \tag{8}$$

It has been argued [10] that ADMM is very slow to converge especially when high accuracy is desired. However, ADMM converges within a few iterations when moderate accuracy is desired. This can be particularly useful for many large scale problems similar to the one we consider in this paper.

Critical to the working and convergence of the ADMM method is the termination criterion. The primal and dual residuals are:

$$r_p^{t+1} = Ax^{t+1} - y^{t+1} \quad \text{(primal residual)}$$

$$r_d^{t+1} = \rho A(y^{t+1} - y^t) \quad \text{(dual residual)}$$

A reasonable termination criterion is when either the primal or the dual residuals are below some thresholds *i.e.*

$$\left\| r_p^{t+1} \right\|_2 \leq \epsilon_p \quad \text{and} \quad \left\| r_d^{t+1} \right\|_2 \leq \epsilon_d.$$

where ϵ_p and ϵ_d are the primary and dual feasibility tolerances. Using user-defined values for ϵ_1 and ϵ_2, these tolerances can be stated as,

$$\epsilon_p = \epsilon_1 \sqrt{m} + \epsilon_2 \max \left(\left\| Ax^{t+1} \right\|_2, \left\| y^{t+1} \right\|_2 \right)$$

$$\epsilon_d = \epsilon_1 \sqrt{n} + \epsilon_2 \left\| A^T p^{t+1} \right\|_2.$$

In the next section we discuss the ADMM update rules for the sparse inverse covariance estimation problem.

4.2. **Alternating Direction Method for sparse inverse kernel estimation.** We start with the prior assumption that the inverse kernel matrix K^{-1} is sparse. This is a reasonable assumption when studying climate data, because given a location *i.e.* any row of the inverse kernel matrix, there are few major locations which influence this location.

With such an assumption, the ADMM algorithm is as follows. Let K be the observed kernel matrix between the grid locations. For a moderate sized K, one can search over all sparsity patterns, since for a fixed sparsity pattern the log likelihood estimate of K is a tractable problem. However, this becomes very challenging for large K. One technique which has been used earlier for sparse covariance selection problem [1] is to minimize the negative log likelihood of $S = K^{-1}$ with respect to the observed data with a penalty term added to induce sparsity. This resulting objective function can be written as

$$\min \quad \mathbf{Tr}(KS) - \log det(S) + \lambda \left\| S \right\|_1$$

where $\left\| \cdot \right\|_1$ is the ℓ_1-norm or the sum of the absolute values of the entries of a matrix and λ is a constant which determines the amount of sparsity. Larger the value of λ, sparser is the solution S. The ADMM version of this problem can be written as follows:

$$\min \quad \mathbf{Tr}(KS) - \log det(S) + \lambda \left\| Y \right\|_1 \quad \text{subject to} \quad S - Y = 0$$

By constructing the augmented Lagrangian and using the derivations given in Section 4.1 for the scaled version of the problem, the ADMM updates for the above estimation problem are:

$$S^{t+1} = \min_x (\mathbf{Tr}(KS) - \log det(S) + \rho/2 \left\| S - Y^t + P^t \right\|_F) \tag{9}$$

$$Y^{t+1} = \min_y \left(\lambda \left\| Y \right\|_1 + \rho/2 \left\| S^{t+1} - Y + P^t \right\|_F \right) \tag{10}$$

$$P^{t+1} = P^t + \left(S^{t+1} - Y^{t+1} \right) \tag{11}$$

with $\|\cdot\|_F$ denoting the Frobenius norm of a matrix. These updates can be simplified further. Taking the derivative of Eqn. 9 and setting it to 0 we get,

$$\begin{aligned} & K - S^{-1} + \rho(S - Y^t + P^t) = 0 \\ \Rightarrow \quad & \rho S - S^{-1} = \rho(Y^t - P^t) - K \end{aligned}$$

Now let $Q\Lambda Q^T$ be the eigen decomposition of $\rho(Y^t - P^t) - K$. Therefore, continuing from the previous step,

$$\begin{aligned} & \rho S - S^{-1} = \rho(Y^t - P^t) - K \\ \Rightarrow \quad & \rho S - S^{-1} = Q\Lambda Q^T \\ \Rightarrow \quad & \rho Q^T S Q - Q^T S^{-1} Q = Q^T Q \Lambda Q^T Q \\ \Rightarrow \quad & \rho \widehat{S} - \widehat{S}^{-1} = \Lambda \quad [\text{since } Q^T Q = Q Q^T = I] \end{aligned} \tag{12}$$

where $\widehat{S} = Q^T S Q$. Solution to Eqn. 12 can easily be found noting that the right hand side is a diagonal matrix of the eigenvalues λ_i's. For each diagonal entry of $\widehat{S}_{ii}$, $\forall i = 1 : n$, we have

$$\rho \widehat{S}_{ii} - \widehat{S}_{ii}^{-1} = \lambda_i$$

which, using the formula of finding the roots of a quadratic equation is

$$\widehat{S}_{ii} = \frac{\lambda_i + \sqrt{\lambda_i^2 + 4\rho}}{2\rho}$$

Therefore, $S = Q\widehat{S}Q^T$ is the optimal value of the S minimization step.

Eqn. 10 can also be simplified further and can be written as the element-wise soft thresholding operation:

$$Y_{ij}^{t+1} = \Im_{\lambda/\rho}\left(S_{ij}^{t+1} + P_{ij}^t\right)$$

In the next section we describe the SPI-GP algorithm in details.

4.3. **SPI-GP: algorithm description.** The SPI-GP algorithm is based on the ADMM technique described in the earlier section. Alg. 1 presents the pseudo-code of the algorithm. The inputs are the kernel K, algorithm parameters λ and ρ, number of iterations *numIter* and the error tolerances ϵ_1 and ϵ_2. The output of the algorithm is the estimated inverse of K in $S = K^{-1}$. The algorithm proceeds in an iterative fashion. In every iteration, an eigen decomposition is performed of the matrix

$$[Q \quad \Lambda] = \rho(Y^{t-1} - P^{t-1}) - K.$$

The eigenvalues Λ and eigenvectors Q are used to update the S variable. The Y-update is a soft thresholding operation of $\left(S^t + P^{t-1}\right)$ with threshold λ/ρ. Finally, the P-update is a linear dual variable update. Also during each iteration, the primal and dual residuals r_p and r_d are computed along with the corresponding error thresholds. Whenever the residuals become less than the error thresholds, the algorithm stops. The result is returned in the matrix S. In our experiments we have chosen $rho = 1$

Running time of ADMM: Since the algorithm requires eigen decomposition for every S update, and the Y and P updates are constant time operations, the runtime complexity is $O(mn^3)$, where m is the number of iterations and n is the size of the dataset (training points).

Convergence of ADMM: In order to ensure convergence of ADMM, two basic assumptions are necessary: (1) the functions G_1 and G_2 are closed, proper and convex, and (2) the unaugmented Lagrangian has a saddle point. Based on these two conditions, it can be shown that [2]:

- primal residual approaches 0 i.e. $r^t \to 0$ as $t \to \infty$
- the objective function approaches the optimal value
- dual variable P approaches feasibility

```
Input: $K$, $\rho$, $\lambda$, $numIter$, $\epsilon_1$, $\epsilon_2$
Output: $S = K^{-1}$
Initialization: $Y^1 = 0, P^1 = 0$
begin
    for $t$=2 to $numIter$ do
        $[Q \quad \Lambda] = \mathbf{evd}[\rho(Y^{t-1} - P^{t-1}) - K]$;
        for $i$=1 to $n$ do
            $\widehat{S}_{ii} = \frac{\lambda_i + \sqrt{\lambda_i^2 + 4\rho}}{2\rho}$;
        end
        $S^t = Q\widehat{S}Q^T$;
        $Y^t = \mathbf{softThreshold}[\left(S^t + P^{t-1}\right), \lambda/\rho]$;
        $P^t = P^{t-1} + (S^t - Y^t)$;
        $r_p = \|S^t - Y^t\|_F$;
        $r_d = \left\|-\rho(S^t - Y^{t-1})\right\|_F$;
        $\epsilon_p = \epsilon_1\sqrt{n} + \epsilon_2 \max(||S^t||_F, ||Y^t||_F)$;
        $\epsilon_d = \epsilon_1\sqrt{n} + \epsilon_2||\rho P^t||_F$;
        if $(r_p < \epsilon_p)$ AND $(r_d < \epsilon_d)$ then
            break;
        end
    end
end
```

Algorithm 1: SPI-GP: ADMM for Sparse Kernel Inversion

In practice however, ADMM may be slow to converge. This type of algorithms, are therefore, more useful when moderate accuracy is necessary within a relatively few iterations. Although this algorithm is slow and sometimes has convergence issues, it is the only method that is amenable to parallel computing which is essential for many large data sets that do not fit in the main memory of a single machine.

4.4. **SPI-GP: distributed implementation.** As we have discussed earlier, ADMM is amenable to distributed computation in a network of machines. This becomes particularly important when the data does not fit into the memory of one machine. This form of ADMM is known as consensus optimization. In this form, the objective function G_1 needs to be decomposable across ℓ nodes $M_1, \ldots, M_\ell$ as follows:

$$\min \quad \sum_{i=1}^{\ell} G_1(x_i) + G_2(y) \text{ subject to } \quad Ax_i - y = 0, \quad x_i \in \mathbb{R}^n, \quad y \in \mathbb{R}^m$$

where x_i is the i-th block of data and is stored at machine M_i. The solution to this optimization is the same as given in Section 4.1. The update rules can be written as,

$$\begin{aligned} x_i^{t+1} &= \min_{x_i}\left\{G_1(x_i) + z^{tT}Ax_i + \rho/2\left\|Ax_i - y^t\right\|_2^2\right\} \\ y^{t+1} &= \min_{y}\left\{G_2(y) + \sum_{i=1}^{\ell}\left(-z_i^{tT}y + \rho/2\left\|Ax_i^{t+1} - y\right\|_2^2\right)\right\} \\ z_i^{t+1} &= z_i^t + \rho\left(Ax_i^{t+1} - y^{t+1}\right) \end{aligned}$$

Unfortunately, the above method cannot be applied for the optimization of the inverse covariance matrix in our case. This is because $\log det(S)$ is not a decomposable function.

Therefore, to solve this problem for large kernel matrices, we use the ScaLAPACK routine of Matlab. It allows the kernel matrix to be distributed across different machines, but still compute the eigen decomposition correctly. For a Matlab implementation, this is done using the co-distributed array data structure and an overloaded *eig* function. It should be noted here that this method *does* *not* attempt to speed up the GPR process. Instead, it makes GPR possible for extremely large data sets where the entire kernel matrix cannot be loaded in the main memory due to size limitations.

5. Experimental results

For the performance study of SPI-GP, the experimental results are reported on a synthetic multivariate Gaussian distribution data and a real life climate domain data set. For generating the multivariate Gaussian, we fix the number of dimensions and samples. We then generate a sparse inverse covariance matrix with all zeros and ones along the diagonal. We randomly insert 1 at certain locations in our inverse covariance. We make this inverse matrix symmetric and positive definite (by making the min eigenvalue positive). Finally we invert this matrix and draw Gaussian samples with zero mean which becomes our covariance matrix. Using this data set we demonstrate the scalability of the distributed SPI-GP method on a cluster of computing nodes.

Our second data set is a historical climate domain data set which consists of NCEP/NCAR features available at `http://www.cdc.noaa.gov/data/gridded/data.ncep.reanalysis.html` [12] and cross-matched normalized difference vegetation index (NDVI) data (NDVI) from the National Oceanic and Atmospheric Administrations Advanced Very High Resolution Radiometer (NOAA/AVHRR). The climate variables used in this study include pressure (hg1000 and hg500), sea surface temperature (sst), Temperature (temp) and precipitation (pre). We use this data set to demonstrate a Gaussian Process regression task where our goal is to take as inputs the first five variables and predict/model precipitation (output) using our SPI-GP method. We have used data from years 1982 - 2002 (21 years). Each variable is observed at a 0.5° resolution over the entire grid. The data used here are composites of observations over a month. Thus there are 360×720=259200 values for each variable vectorized and stored as a single row corresponding to a time point (a month). Therefore, each variable has $12 \times 21 = 252$ rows in the data set, each having 259200 columns. Note that some variables are observed only in land while others only in ocean. For any variable, the locations which do not contain any meaningful data has a fill value of -9999.0.

If we want to use all five variables (hg1000, hg500, sst, temp, ndvi) for predicting precipitation, then we have to create a GPR model which takes the five variables as input and precipitation as the output. Since there are missing values for each variable, the locations where all values are present are the coastlines of the continents (only approximately 8500 points). This means that if we build a model based on only these points, the other data points cannot be used in the model. Instead we use a multiple kernel approach as follows. Let K_{hg1000}, K_{hg500}, K_{ndvi}, K_{sst}, and K_{temp} be the kernels computed from each of the 5 input variables separately after removing the fill values. We create a global training kernel as,

$$K_{global} = K_{hg1000} + K_{hg500} + K_{sst} + K_{temp} + K_{ndvi}$$

Similarly we create the test kernel as,

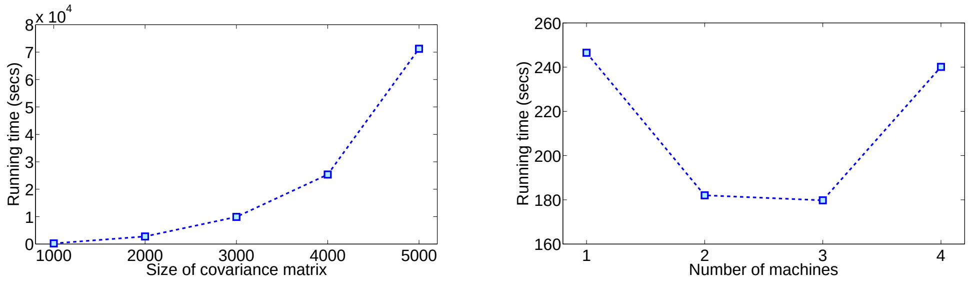
$$K^*_{global} = K^*_{hg1000} + K^*_{hg500} + K^*_{sst} + K^*_{temp} + K^*_{ndvi}$$

In both these global kernels, we normalize the values by making their range between 0 and 1. We then use the following two GPR equations

$$\widehat{\mathbf{y}}^* = K^*_{global}(\sigma^2 I + K_{global})^{-1}\mathbf{y} \tag{13}$$

$$C = K^{**}_{global} - K^*_{global}(\sigma^2 I + K_{global})^{-1} K^{*T}_{global} \tag{14}$$

using the kernels just computed by combining the individual kernels. By construction, K_{global} is a matrix on the entire set of grid locations ($n \times n$). K^*_{global} is a test kernel of size $m \times n$, where m is the number of test locations. y is the training output of size $n \times 1$. As a result $\widehat{y}^*$ becomes of size $m \times 1$. One issue is in using the entire y vector. For our prediction problem, this corresponds to precipitation and hence has only values on land. So when we use the y vector for the entire world's data, it contains missing values of -9999.0 (about 40% of the total size of y). To circumvent this problem, we replace the fill values with an average value of the feature y.



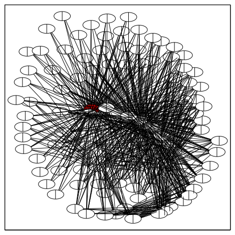
(a) Running time in seconds for different sizes of the training data on 4 processors

(b) Running time in seconds for different number of processors on a 1000×1000 matrix

FIGURE 2. Scalability study of SPI-GP on synthetic data

5.1. **Study 1: Scalability study on synthetic data.** In this study we report the scalability of the SPI-GP algorithm. The metric we use is running time (in seconds). We report results from two different experiments. In our first experiment we fix the number of cores on which we run our experiment and vary the size of the training data. Figure 2 (left) shows the result. We used the Matlab Parallel Computing toolbox and a local scheduler for multicore architecture. For this experiment we chose 4 processors on a single CPU to simulate the distributed computing environment. We experimented with five different sizes of the covariance matrix starting from 1000×10000 to 5000×50000 and notice that the growth in the running time is less than cubic in spite of the eigen decomposition step. This is due to the distributed eig function usage which makes the method complexity $O(nr^2)$ where r is the chunk (rank) of the matrix for the covariance matrix partition in any one of the processors. Figure 2 (right) reports the results of running SPI-GP on a 1000×10000 matrix on a varying number of processors starting from 1 to 4. The result is counter-intuitive since we see that a single processor takes the highest time while there is no clear trend in the time as we increase the number of processors, keeping the data fixed. This is because there is considerable overhead in distributing a job over the parallel computing framework and there is an optimal number of processors for a fixed partitioning of the data. The performance degrades with deviation from the optimal.

5.2. **Study 2: Precipitation prediction in the Indian subcontinent.** In the climate study, we observe which geographical regions are most similar to the precipitation pattern of India. We want to identify these points and study how these points change over a time period of 20 years. Since all climatic connections change very slowly with time, we construct the relevant network connections for Indian precipitation every 5 years. Fig. 4 shows the results. Each plot in Fig. 4 is for the average of one year's data. The variable shown in the figures is precipitation. The black markers are the locations in India. The yellow markers indicate the the top 10 areas which influence India. These are the points which have the highest values in the estimated inverse kernel matrix corresponding to test points for India. As Figure 4 shows, there are certain regions which remain similar to our test set for the entire period of 20 years, while others have a more disparate pattern. Some locations which show consistent influence pattern include the west coast of South America, west coast of Africa, and east coast of Australia. Some less consistent locations include areas in China. To illustrate how this method can be used for studying climate networks, we represent a portion of the precipitation-based inverse covariance matrix as a network. As can be seen in Figure 3, the true inverse covariance is difficult to understand or interpret, given the huge amount of network connections for any particular node in the graph. The reference node in this study is denoted in red in both the left and right subfigures in Figure 3. The right figure, which a sparse variant of the same graph shows only the important connections to the colored node, and enhances interpretability of learnt models like in Gaussian Process regression.


(a) Network representing a sub-matrix of the inverse covariance matrix

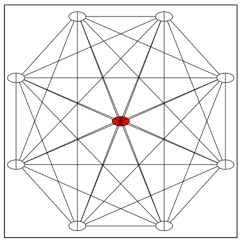

(b) Network representing a sub-matrix of the sparse inverse covariance matrix estimated using SPI-GP

FIGURE 3. Interpretability of sparse inverse covariance matrix

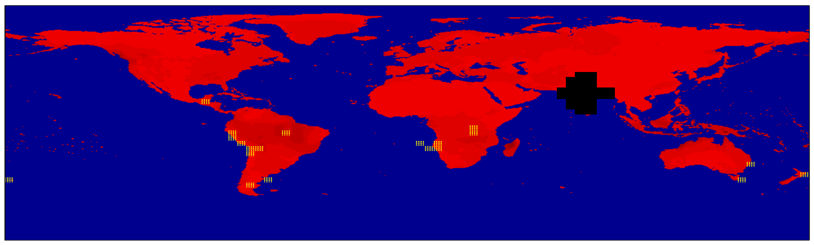
As we will observe in section 5.3, this regression problem performs poorly due to the immense amount of missing data in the different modalities used to predict precipitation. Therefore, for demonstrating the fact, that the poor regression results are only due to the nature of data available, and not due to the technique discussed here, we study a a different regression problem where we want to predict the precipitation in the Indian subcontinent based on only precipitation data from four weeks in advance. In this study, we use only precipitation data to predict precipitation for a delay of 1 month. This study is also performed for over a 20 year period at intervals of every 5 years. For every year we study the prediction problem quarterly.

5.3. **NMSE of SPI-GP.** If a set of points are very similar to the points representing rainfall in the Indian subcontinent, then it is intuitive that those points should be very good predictor of precipitation in India. Our next study tries to verify this intuition. For this, we choose the top k locations of the world that are most similar to the precipitation in the Indian subcontinent for each of the years 1982, 1986, 1990, 1994, and 1998 and build GPR models by taking only this subset as the training examples. We test on year 2002. As a baseline comparison, we train a separate GPR on the entire world's data (Full-GP). For both these methods, we use the same locations of India as test sets. We build these two GPR's separately for each of the five years mentioned before. The first row of Table 5.3 shows the normalized mean squared error (NMSE) values for these two GPR methods for each of the five years, where NMSE is defined as

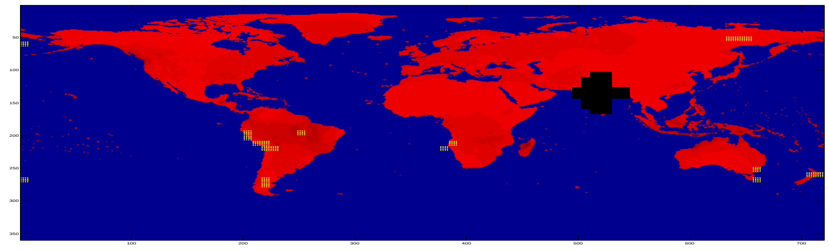
$$NMSE = \frac{\sum_{i=1}^{n}(\widehat{y}_i^* - y_i)^2}{n \times var(\widehat{\mathbf{y}}^*)}.$$

The value of k is chosen to be $n/2$ where n is kernel dimension. For this study, for each of the five years, the NMSE value for the GPR model of top k values from SPI-GP is better than the Full-GP. This happens because the most similar points capture more information and less of noise as has been verified earlier in [4]. However, as it can be noted the improvement in NMSE observed is not significant. Not only that, even for the improvement that is observed, the NMSE values are quite high (approximately 1). Now, a value of 1 for NMSE implies that the prediction is equal to the mean of the target. This explains the observed NMSE in our experiments. Since approximately 40% of the target data used in our experiments were actually fill values and were replaced by the mean of the target. Therefore, the NMSE that we see is largely an artifact of the data preprocessing for this data set since the mean-based smoothing technique applied here may have failed to capture the dynamics in the data.

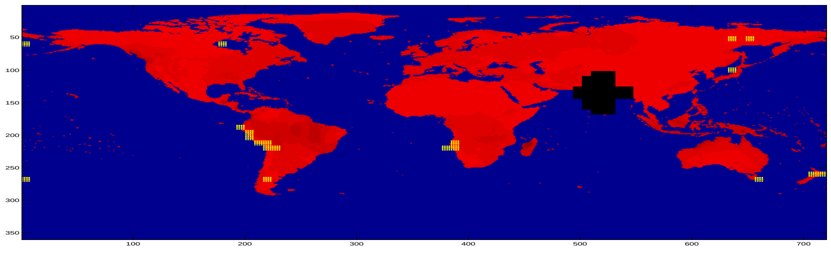
To verify that the high NMSE values are not an artifact of the technique, but the data, we perform similar experiments for the precipitation based regression study. The second row of Table 5.3 shows the NMSE values when we predict rainfall for August of 2002 based on rainfall in July for each of the years 1982, 1986, 1990, 1994, and 1998. We can notice that the NMSE values are much lower



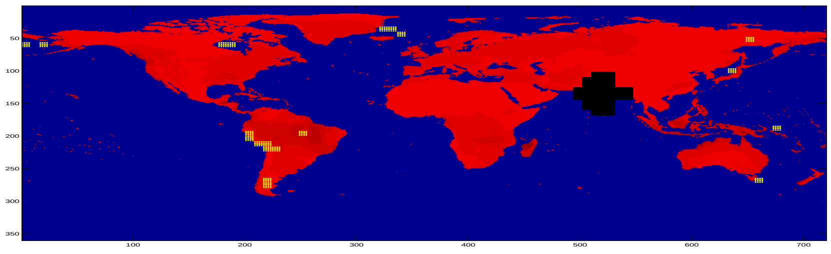
(a) Climate network for 1982 based on precipitation



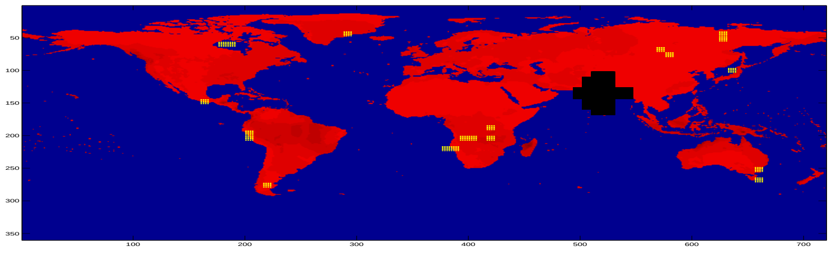
(b) Climate network for 1986 based on precipitation



(c) Climate network for 1991 based on precipitation



(d) Climate network for 1996 based on precipitation



(e) Climate network for 2001 based on precipitation

FIGURE 4. Evolution of the climate network over 20 years based on precipitation data.

		1982	1986	1990	1994	1998
Regression using all variables	Full-GP	1.085	1.218	1.115	1.883	1.138
	SPI-GP	0.902	0.811	1.05	1.072	0.979
Regression using precipitation	Full-GP	0.695	0.664	0.611	0.651	0.669
	SPI-GP	0.6912	0.664	0.605	0.650	0.667

Table 1. NMSE of GPR for 2002 when entire world's data is used (Full-GP) vs. top few similar points in SPI-GP. For the first regression scenario, each column shows the NMSE for that year. For the second scenario, each column shows the NMSE for prediction of rainfall in August for that year.

Training years	Training months							
	1		4		7		10	
	Full-GP	SPI-GP	Full-GP	SPI-GP	Full-GP	SPI-GP	Full-GP	SPI-GP
1982	0.237	0.283	0.454	0.439	0.426	0.426	0.371	0.361
1983	0.258	0.292	0.492	0.492	0.658	0.658	0.374	0.374
1984	0.261	0.273	0.451	0.451	0.818	0.819	0.374	0.368
1985	0.196	0.208	0.475	0.450	0.396	0.396	0.385	0.385

Table 2. NMSE of GPR for 1986 when entire world's data is used (Full-GP) vs. top few similar points in SPI-GP.

Training years	Training months							
	1		4		7		10	
	Full-GP	SPI-GP	Full-GP	SPI-GP	Full-GP	SPI-GP	Full-GP	SPI-GP
1982	0.311	0.293	0.554	0.563	0.706	0.706	1.23	1.22
1986	0.325	0.295	0.587	0.595	0.81	0.809	1.301	1.3
1991	0.281	0.278	0.564	0.586	0.782	0.781	1.15	1.15

Table 3. NMSE of GPR for 1996 when entire world's data is used (Full-GP) vs. top few similar points in SPI-GP.

compared to the first study. However, it should be noted that the precipitation prediction problem that we are studying is a difficult one since the data does not have reasonably high predictability. The linear correlations for different data subsets and different test sets can vary from -0.2 (very poor) to 0.88 (high correlation) accounting for the high variability in the NMSE values for the different test scenarios. Tables 5.3 and 5.3 document the NMSE values for predicting precipitation in India for months February, May, August and November for the years 1986 and 1996 respectively. NMSE values in the table range from as low as .19 to as high as 1.3 indicating the difficulty level of different prediction scenarios. For example year 1986 has reasonably good predictability and has lower variation in the NMSE values thatn year 1996. Although February, May and August have quite low NMSE values for 1996, the month of November does not have that since the prediction is working as poorly as random for the different training years. The year 2002 is worse than 1986 in terms of average predictability, but data is more consistent across the different training years.

6. Conclusion

In this paper we discuss a method for sparse inverse Gaussian Process regression that allows us to compute a parsimonious model while preserving the interpretability of the sparsity structure in the data. We discuss how the inverse kernel matrix used in Gaussian Process prediction gives valuable information about the regression model and then adapt the inverse covariance estimation from

Gaussian graphical models to estimate the Gaussian kernel. We solve the optimization problem using the alternating direction method of multipliers that is amenable to parallel computation. This sparsity exploiting GPR technique achieves two goals: (i)it provides valuable insight into the regression model and (ii)it allows for parallelization so that the entire kernel matrix need not be loaded into a single main memory, thereby removing the size related constraints plaguing large scale analysis. We perform experiments on historical climate data of 20 years. The climate network study shows evolution of the most influential points over time for predicting precipitation in the Indian subcontinent. The NMSEs reported are relatively high due to the mean-based smoothing adopted in the preprocessing. For future work, we plan to pursue other spatial smoothing processes such as the ones proposed by Cressie and Wikle [3]. We also want to pursue teleconnection study using different climate data for specific climate scenarios.

Acknowledgements

This work was supported by the Systemwide Safety and Assurance Technologies project of Aviation Safety Program under the NASA Aeronautics Mission Directorate.

References

[1] O. Banerjee, L. Ghaoui, A. d'Aspremont, and G. Natsoulis. Convex optimization techniques for fitting sparse Gaussian graphical models. In *Proceedings ICML-06*, pages 89–96, 2006.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 2011.

[3] N. Cressie and C. K. Wikle. *Statistics for Spatio-Temporal Data*. Wiley, 2011.

[4] K. Das and A. Srivastava. Block-GP: Scalable Gaussian Process Regression for Multimodal Data. In *The 10th IEEE International Conference on Data Mining, ICDM 2010*, pages 791–796, 2010.

[5] A. P. Dempster. Covariance Selection. *Biometrics*, 28:157–175, 1972.

[6] J. Eckstein and D. Bertsekas. An alternating direction method for linear programming. Technical Report LIDS-P ; 1967, MIT, 1990.

[7] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

[8] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, M. Way, P. Gazis, and A. Srivastava. Stable and Efficient Gaussian Process Calculations. *JMLR*, 10:857–882, 2009.

[9] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics Journal*, 9(3):432–441, 2008.

[10] M. Fukushima. Application of the alternating direction method of multipliers to separable convex programming problems. *Computational Optimization and Applications*, 1:93–111, 1992.

[11] M. H. Glantz, R. W. Katz, and N. Nicholls. *Teleconnections linking worldwide climate anomalies : scientific basis and societal impact*. Cambridge University Press, 1991.

[12] E. Kalnay, M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Candin, S. Saha, G. White, J. Woollen, Y. Zhu, M. Chelliah, W. Ebisuzaki, W. Higgins, J. Janowiak, K. C. Mot, C. Ropelewski, A. Leetmaa, R. Reynolds, R. Jenne, and D. Joseph. The NCEP/NCAR 40-Year Reanalysis Project. *B. Am Metrolo. Soc.*, 77(3):437–471, 1996.

[13] N. Meinshausen, P. Bhlmann, and E. Zrich. High Dimensional Graphs and Variable Selection with the Lasso. *Annals of Statistics*, 34:1436–1462, 2006.

[14] J. Quiñonero-Candela and C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *JMLR*, 6:1939–1959, 2005.

[15] C. E. Rasmussen. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[16] A. J. Smola and P. Bartlett. Sparse Greedy Gaussian Process Regression. In *Proc. of NIPS 13*, pages 619–625, 2000.

[17] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Proceedings of NIPS 18*, 2005.

[18] K. Steinhaeuser, N. Chawla, and A. Ganguly. An exploration of climate data using complex networks. *SIGKDD Explorations Newsletter*, 12:25–32, November 2010.

[19] R. Tibshirani. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[20] V. Tresp. The generalized bayesian committee machine. In *Proc. of KDD*, pages 130–139, 2000.

[21] G. Wahba. *Spline Models for Observational Data*. SIAM, 1990.

[22] F. Yan and Y. Qi. Sparse Gaussian Process Regression via ℓ_1 Penalization. In *Proceedings of ICML-10*, pages 1183–1190, 2010.