Scalable Gaussian Process Using Inexact ADMM for Big Data

Yue Xu†∗, Feng Yin∗, Jiawei Zhang∗, Wenjun Xu†, Shuguang Cui‡∗ and Zhi-Quan Luo∗

†Beijing University of Posts and Telecommunications

∗The Chinese University of Hong Kong, Shenzhen and SRIBD

‡Department of Electrical and Computer Engineering, University of California, Davis

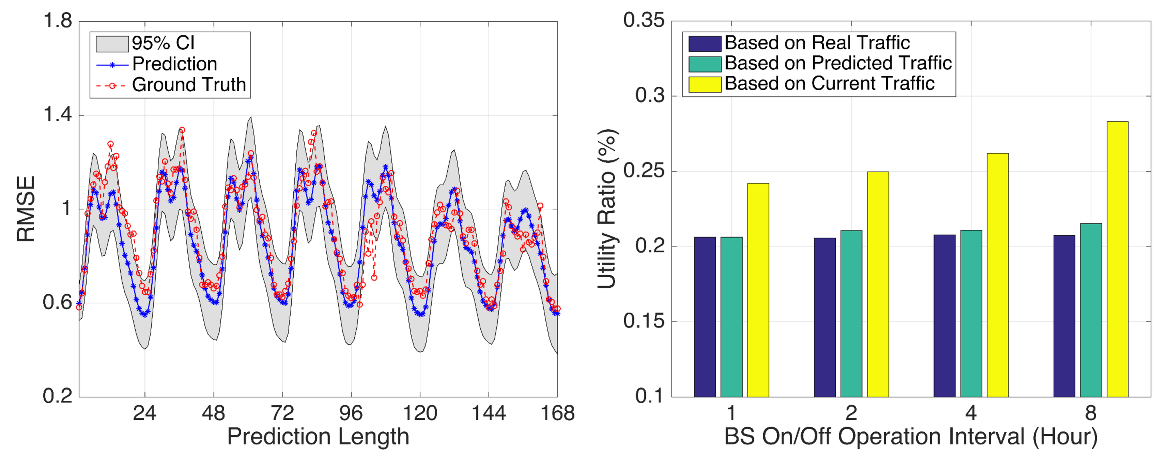
香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Background

- **What is GP?**

Gaussian process (GP) model is a class of important **Bayesian non-parametric models for machine learning.**

- **Application**



(a) Predicted Traffic by GP (b) Energy Saving Result

Figure 1. Performance of wireless traffic prediction based energy saving.

- **Challenge**

1) Standard GP suffers from the high complexity of **hyper-parameter optimization**, which scales as $\mathcal{O}(n^3)$ with n the number of training samples.

2) Existing low-complex GP model reduces the complexity based on certain approximations, e.g., the subset-of-data (SOD) model based on sparse GP, the Bayesian committee machine (BCM).

Main Result

- **A Scalable GP Model for Processing Big Datasets**

A novel **scalable GP regression model**, which is **parallelizable** over a large number of computation units and **does not involve any approximation** essentially.

- **Faster Hyper-parameter Optimization**

A practical implementation with the Gauss-Seidel method, which reduces the complexity to $\mathcal{O}(n^3/k)$ with k the number of parallel computing units.

Simulation

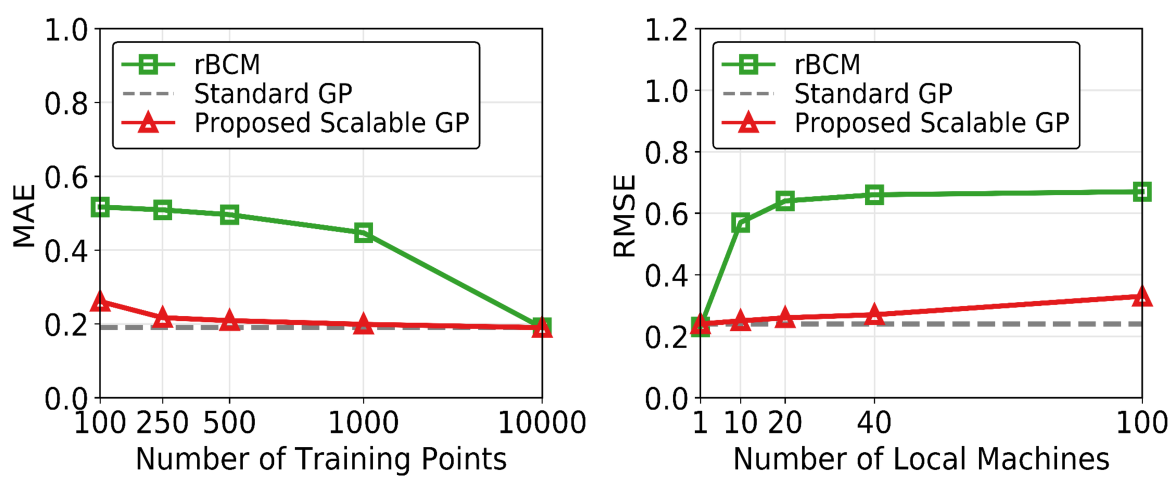


Figure 2: The Prediction Performance.

- **Baseline Model**
- The state-of-art robust BCM (rBCM) model.
- The standard GP model (Optimizes the GP parameters as in $\mathcal{P}_0$)

- **Simulation Setting**
- Artificially generated datasets with SE kernel.
- Using 10000 points as the training dataset to predict the next upcoming data point.
- Repeat 300 times (iteratively update the training set) to average the performance.
- Run at a workstation with eight Intel E3 Xeon CPU cores at 3.50 GHz.

Regression Model

- **GP Definition**

A GP is a collection of random variables, any finite number of which follow a Gaussian distribution.

- **GP Function**

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})),$$

where

- $\boldsymbol{x}$: continuous-value input;
- $m(\boldsymbol{x})$: mean function (zero in practice);
- $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$: kernel function (e.g., SE, periodic).

- **GP-based Regression Model**

$$y = f(\boldsymbol{x}) + e,$$

where

- y: continuous-value output;
- e: noise (estimated independently).

Standard GP

- **Standard GP Hyper-parameter Optimization**

$$\mathcal{P}_0: \quad \arg\min_{\theta} g(\theta) = y^T C^{-1}(\theta) y + \log|C(\theta)|\,,$$
$$s.t. \quad \theta \in \Theta,$$

where

- $C(\theta) = K(X, X; \theta) + \sigma^2 I_n$: covariance matrix;
- $K(X, X; \theta)$: kernel matrix.

- **Gradient Decent (Benchmark Method)**

$$\theta^{r+1} = \theta^r - \mu \cdot \nabla_\theta g(\theta)|_{\theta=\theta^r},$$

where

- $\frac{\partial g(\boldsymbol{\theta})}{\partial \theta_j} = \mathrm{Tr}\left(\boldsymbol{C}^{-1}(\boldsymbol{\theta}) \frac{\partial \boldsymbol{C}(\boldsymbol{\theta})}{\partial \theta_j}\right) - \boldsymbol{y}^T \boldsymbol{C}^{-1}(\boldsymbol{\theta}) \frac{\partial \boldsymbol{C}(\boldsymbol{\theta})}{\partial \theta_j} \boldsymbol{C}^{-1}(\boldsymbol{\theta}) \boldsymbol{y};$
- $C^{-1}(\theta)$: $\mathcal{O}(n^3)$ computational complexity.

A Motivating Example

❶ Given a covariance matrix with four blocks:

$$C(\theta_1, \theta_2, \theta_3, \theta_4) = \left[\begin{array}{cc:cc} a_{11}^{\theta_1} & a_{12}^{\theta_1} & a_{13}^{\theta_2} & a_{14}^{\theta_2} \\ a_{21}^{\theta_1} & a_{22}^{\theta_1} & a_{23}^{\theta_2} & a_{24}^{\theta_2} \\ \hdashline a_{31}^{\theta_3} & a_{32}^{\theta_3} & a_{33}^{\theta_4} & a_{34}^{\theta_4} \\ a_{41}^{\theta_3} & a_{42}^{\theta_3} & a_{43}^{\theta_4} & a_{44}^{\theta_4} \end{array}\right].$$

❷ Computation in the first unit:

$$C(\boldsymbol{\theta}_1, z_{-1}^r) = \left[\begin{array}{cc:cc} a_{11}^{\theta_1} & a_{12}^{\theta_1} & a_{13}^{\theta_2^r} & a_{14}^{\theta_2^r} \\ a_{21}^{\theta_1} & a_{22}^{\theta_1} & a_{23}^{\theta_2^r} & a_{24}^{\theta_2^r} \\ \hdashline a_{31}^{\theta_3^r} & a_{32}^{\theta_3^r} & a_{33}^{\theta_4^r} & a_{34}^{\theta_4^r} \\ a_{41}^{\theta_3^r} & a_{42}^{\theta_3^r} & a_{43}^{\theta_4^r} & a_{44}^{\theta_4^r} \end{array}\right]$$

$$= \begin{bmatrix} \boldsymbol{C}_{11}(\boldsymbol{\theta}_1) & \boldsymbol{C}_{12}(\boldsymbol{\theta}_2^r) \\ \boldsymbol{C}_{21}(\boldsymbol{\theta}_3^r) & \boldsymbol{C}_{22}(\boldsymbol{\theta}_4^r) \end{bmatrix}.$$

❸ The block-wise matrix inverse satisfies:

$$\begin{bmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{12} \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_{\frac{n}{2}\times\frac{n}{2}} & \boldsymbol{0}_{\frac{n}{2}\times\frac{n}{2}} \\ \boldsymbol{0}_{\frac{n}{2}\times\frac{n}{2}} & \boldsymbol{I}_{\frac{n}{2}\times\frac{n}{2}} \end{bmatrix}.$$

❹ The matrix derivative satisfies:

$$\frac{\partial \boldsymbol{C}}{\partial \theta_{1j}} = \begin{bmatrix} \frac{\partial \boldsymbol{C}_{11}(\boldsymbol{\theta})}{\partial \theta_{1j}} & \boldsymbol{0}_{\frac{n}{2}\times\frac{n}{2}} \\ \boldsymbol{0}_{\frac{n}{2}\times\frac{n}{2}} & \boldsymbol{0}_{\frac{n}{2}\times\frac{n}{2}} \end{bmatrix}.$$

❺ Solution of the block-wise matrix inverse:

$$\boldsymbol{B}_{11} = (\boldsymbol{C}_{11} - \boldsymbol{C}_{12}\boldsymbol{C}_{22}^{-1}\boldsymbol{C}_{21})^{-1},$$
$$\boldsymbol{B}_{12} = \boldsymbol{B}_{21}^T = -\boldsymbol{C}_{22}^{-1}\boldsymbol{C}_{21}\boldsymbol{B}_{11},$$
$$\boldsymbol{B}_{22} = (\boldsymbol{C}_{22} - \boldsymbol{C}_{21}\boldsymbol{C}_{11}^{-1}\boldsymbol{C}_{12})^{-1}.$$

Scalable GP

- **Goal**

Break the problem $\mathcal{P}_0$ into smaller pieces that are easier to handle distributively without making any approximations.

- **ADMM-based GP Hyper-parameter Optimization**

$$\mathcal{P}_1: \quad \arg\min_{\{\boldsymbol{\theta}_i\}} g(\{\boldsymbol{\theta}_i\}),$$
$$\text{s.t.} \quad \boldsymbol{\theta}_i - \boldsymbol{z} = \boldsymbol{0}, \boldsymbol{\theta}_i \in \Theta, i \in \mathcal{K},$$

where:

- $g(\{\boldsymbol{\theta}_i\}) = \boldsymbol{y}^T \boldsymbol{C}^{-1}(\{\boldsymbol{\theta}_i\})\boldsymbol{y} + \log|\boldsymbol{y}^T \boldsymbol{C}^{-1}(\{\boldsymbol{\theta}_i\})|$;
- $\boldsymbol{C}^{-1}(\{\boldsymbol{\theta}_i\})$: i-th block determined by $\boldsymbol{\theta}_i$.

Remark: $\mathcal{P}_1$ is equivalent to $\mathcal{P}_0$.

- **Lagrangian Function**

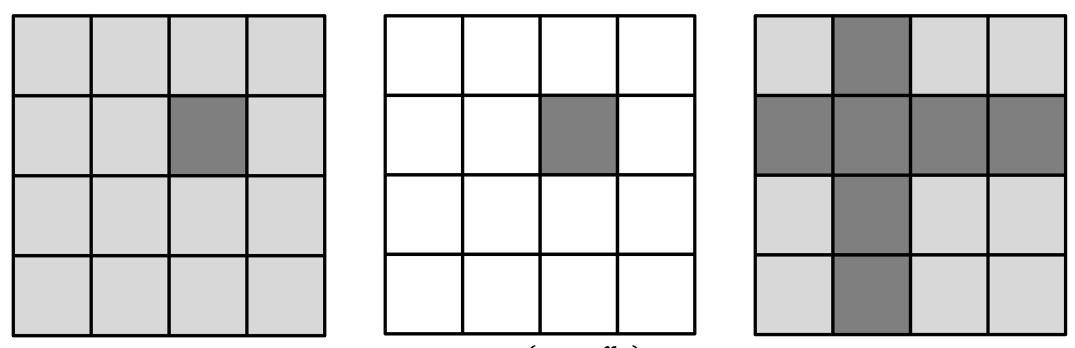
$$\mathcal{L}(\{\boldsymbol{\theta}_i, \boldsymbol{z}, \boldsymbol{\beta}\}) \triangleq g(\{\boldsymbol{\theta}_i\}) + \sum_{i=1}^{k} \boldsymbol{\beta}_i^T(\boldsymbol{\theta}_i - \boldsymbol{z}) + \sum_{i=1}^{k} \frac{\rho}{2} \|\boldsymbol{\theta}_i - \boldsymbol{z}\|_2^2.$$

- **ADMM Iteration**

$$\boldsymbol{\theta}_i^{r+1} = \arg\min_{\boldsymbol{\theta}_i} g(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r) + \boldsymbol{\beta}_i^{r,T}(\boldsymbol{\theta}_i - \boldsymbol{z}^r) + \frac{\rho}{2}\|\boldsymbol{\theta}_i - \boldsymbol{z}^r\|_2^2,$$
$$\boldsymbol{z}^{r+1} = \frac{1}{k}\sum_{i=1}^{k}\left(\boldsymbol{\theta}_i^{r+1} + \frac{1}{\rho}\boldsymbol{\beta}_i^r\right),$$
$$\boldsymbol{\beta}_i^{r+1} = \boldsymbol{\beta}_i^r + \rho(\boldsymbol{\theta}_i^{r+1} - \boldsymbol{z}^{r+1}).$$

where:

- $g(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r) = \boldsymbol{y}^T \boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)\boldsymbol{y} + \log|\boldsymbol{y}^T \boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)|$;
- $\boldsymbol{z}_{-i}^r = \{\boldsymbol{\theta}_1^r, \ldots, \boldsymbol{\theta}_{i-1}^r, \boldsymbol{\theta}_{i+1}^r, \ldots, \boldsymbol{\theta}_k^r\} \backslash \boldsymbol{\theta}_i$;
- $\boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)$: one block determined by $\boldsymbol{\theta}_i$, the others by $\boldsymbol{z}_{-i}^r$.



(a) $\boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)$ (b) $\frac{\partial \boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)}{\partial \theta_{ij}}$ (c) $\boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)$

Figure 3. When updating the local hyper-parameter, say $\boldsymbol{\theta}_i$, the i-th local computing unit requires only one block of the full covariance matrix, e.g., the dark block in (a); only one block of the partial derivative matrix is non-zero, e.g., the dark block in (b); only one vertical and one horizontal slice of the full matrix inverse is needed, e.g., the two dark slices in (c), for gradient update.

A Practical Implementation (Gauss-Seidel Method)

$$\left[\nabla_{\boldsymbol{\theta}_i}\mathcal{L}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r, \boldsymbol{z}^r, \boldsymbol{\beta}^r)|_{\boldsymbol{\theta}_i=\boldsymbol{\theta}_i^r}\right]_j = \frac{\partial}{\partial \theta_{ij}}\left(g(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r) + (\boldsymbol{\beta}_i^r)^T(\boldsymbol{\theta}_i - \boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\theta}_i - \boldsymbol{z}^r\|_2^2\right)\Bigg|_{\boldsymbol{\theta}_i=\boldsymbol{\theta}_i^r}$$

$$= \mathrm{Tr}\left(\boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)\frac{\partial \boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)}{\partial \theta_{ij}}\right) - \boldsymbol{y}^T \boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)\frac{\partial \boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)}{\partial \theta_{ij}}\boldsymbol{C}^{-1}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)\boldsymbol{y} + \beta_{ij}^r + \rho(\theta_{ij} - z_j^r)\Bigg|_{\boldsymbol{\theta}_i=\boldsymbol{\theta}_i^r}$$

$$\approx \mathrm{Tr}\left(\boldsymbol{C}^{-1}(\boldsymbol{z}^r)\frac{\partial \boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)}{\partial \theta_{ij}}\right) - \boldsymbol{y}^T \boldsymbol{C}^{-1}(\boldsymbol{z}^r)\frac{\partial \boldsymbol{C}(\boldsymbol{\theta}_i, \boldsymbol{z}_{-i}^r)}{\partial \theta_{ij}}\boldsymbol{C}^{-1}(\boldsymbol{z}^r)\boldsymbol{y} + \beta_{ij}^r + \rho(\theta_{ij} - z_j^r).$$

Related Works

- Y. Xu, F. Yin, W. Xu, J. Lin and S. Cui, "Wireless Traffic Prediction with Scalable Gaussian Process: Framework, Algorithms, and Verification," in IEEE Journal on Selected Areas in Communications (JSAC), March 2019, to appear.
- Y. Xu, F. Yin, W. Xu, J. Lin and S. Cui, " Distributed Gaussian Process: New Paradigm and Application to Wireless Traffic Prediction," in *IEEE International Conference on Communications (ICC)*, Shanghai, China, May 2019, to appear. in *IEEE International Conference on Communications (ICC)*, 2019.
- Y. Xu, F. Yin, W. Xu, J. Lin and S. Cui, " High-Accuracy Wireless Traffic Prediction: A GP-Based Machine Learning Approach," in *IEEE Global Communications Conference (GLOBECOM)*, Singapore, December 2017, pp. 1–6.
- W. Xu, Y. Xu, Y. Xu, C. Lee, Z. Feng, P. Zhang and J. Lin, " Data-Cognition-Empowered Intelligent Wireless Networks: Data, Utilities, Cognition Brain, and Architecture," in *IEEE Wireless Communications*, vol. 25, no. 1, pp. 56– 63, February 2018.

Contact:
- Yue XU (xuy@bupt.edu.cn)
- Yin FENG (yinfeng@cuhk.edu.cn)