Gaussian Process Regression with Grid Spectral Mixture Kernel: Distributed Learning for Multidimensional Data

Presenter: Richard Cornelius Suwandi

The Chinese University of Hong Kong, Shenzhen

July 7, 2022

FUSION 2022

Contents

Introduction

Motivation

- Kernel design for Gaussian processes (GPs) along with the associated hyper-parameter optimization is a challenging problem
- The computational complexity for training the model hyper-parameters can be very demanding and even prohibitive for large data sets
- Large amount of labeled training data are usually aggregated from a large number of local agents or mobile devices, which may cause severe data privacy issues

Contributions

- We propose a novel grid spectral mixture (GSM) kernel [1, 2] design for GPs that can automatically fit multidimensional data
- Two efficient distributed learning algorithms are proposed to alleviate the computational complexity owing to the curse of dimensionality in the kernel hyper-parameter optimization
- The proposed algorithms can help with preserving data privacy during the learning process

Background

Gaussian Process (GP) Regression

- We consider the following GP regression model

$$y = f(\boldsymbol{x}) + e, \quad e \sim \mathcal{N}(0, \sigma_e^2) \tag{1}$$

where $f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}_h))$ is a real-valued, scalar Gaussian process with mean function $\mu(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}_h)$

- The set of unknown hyper-parameters that needs to be tuned is denoted by $\boldsymbol{\theta} \triangleq [\boldsymbol{\theta}_h^T, \sigma_e^2]^T$

Grid Spectral Mixture (GSM) Kernel

- The basic idea behind the GSM kernel is to undertake an approximation of the underlying stationary kernel using the fact that any stationary kernel and its spectral density are Fourier duals [3]

Theorem

If the spectral density exists, then the stationary kernel function, $k(\boldsymbol{\tau})$, and its spectral density, $S(\boldsymbol{\omega})$, are Fourier duals of each other

$$k(\boldsymbol{\tau}) = \int_{\mathbb{R}^{d_x}} S(\boldsymbol{\omega}) \exp\left[j2\pi\boldsymbol{\tau}^\top\boldsymbol{\omega}\right] d\boldsymbol{\omega}, \tag{2a}$$

$$S(\boldsymbol{\omega}) = \int_{\mathbb{R}^{d_x}} k(\boldsymbol{\tau}) \exp\left[-j2\pi\boldsymbol{\tau}^\top\boldsymbol{\omega}\right] d\boldsymbol{\tau}. \tag{2b}$$

Grid Spectral Mixture (GSM) Kernel

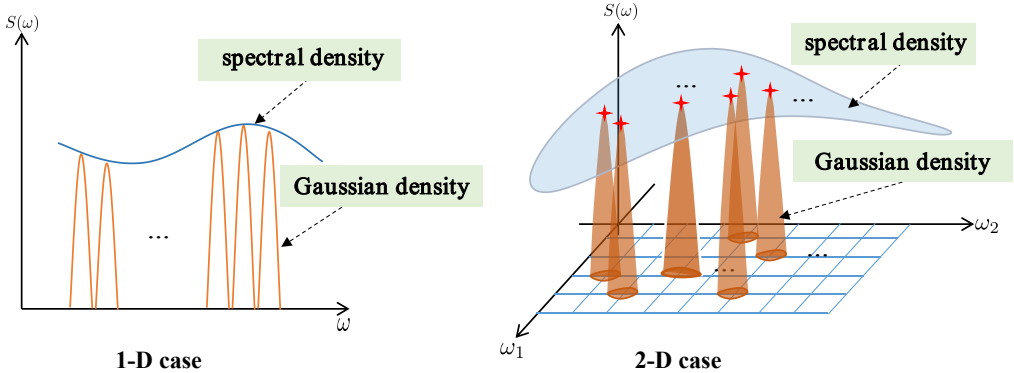

Figure: Basis kernel spectral density space under different input dimensions.

GSM Kernel: One-Dimensional Input

- The GSM kernel approximates the spectral density of the underlying kernel function in the frequency domain by a Gaussian mixture,

$$S(\omega) = \frac{1}{2}\sum_{q=1}^{Q}\theta_q\left[\mathcal{N}\left(\omega \mid \mu_q, v_q\right) + \mathcal{N}\left(\omega \mid -\mu_q, v_q\right)\right], \tag{3}$$

where $\{\mu_q\}_{q=1}^{Q}$ and $\{v_q\}_{q=1}^{Q}$ are fixed to preselected grid points and $\{\theta_q\}_{q=1}^{Q}$ are weights to be optimized

- Taking the inverse Fourier transform of $S(\omega)$, yields the original GSM kernel as

$$k(\tau) = \sum_{q=1}^{Q}\theta_q \underbrace{\cos(2\pi\tau\mu_q)\exp\left[-2\pi^2\tau^2 v_q\right]}_{k_q(\tau)}, \tag{4}$$

where $\tau = |x - x'| \in \mathbb{R}$

Hyper-parameter Optimization

- Learning the model hyper-parameters $\boldsymbol{\theta}$ in GPR model typically resorts to the type-II maximum likelihood,

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \underbrace{\boldsymbol{y}^\top \left[\boldsymbol{C}(\boldsymbol{\theta})\right]^{-1} \boldsymbol{y} + \log\det\left[\boldsymbol{C}(\boldsymbol{\theta})\right]}_{\triangleq l(\boldsymbol{\theta})} + \text{constant}, \tag{5}$$

where $\boldsymbol{C}(\boldsymbol{\theta}) \triangleq \boldsymbol{K}_{XX} + \sigma_e^2 \boldsymbol{I}_n$

- The optimization problem in Eq. (5) is a well-known difference-of-convex programming (DCP) problem [4], where $g(\boldsymbol{\theta}) \triangleq \boldsymbol{y}^\top \left[\boldsymbol{C}(\boldsymbol{\theta})\right]^{-1} \boldsymbol{y}$ and $h(\boldsymbol{\theta}) \triangleq -\log\det\left[\boldsymbol{C}(\boldsymbol{\theta})\right]$ are convex functions w.r.t. $\boldsymbol{\theta}$
- This DCP problem can be efficiently solved using the successive convex approximation (SCA) algorithm [5]

Vanilla SCA

- The vanilla SCA algorithm generates a sequence of feasible points $\boldsymbol{\theta}^t, t \in \mathbb{N}$ by solving

$$\boldsymbol{\theta}^{t+1} = \arg\min_{\boldsymbol{\theta}} \tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) \tag{6}$$

where $\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) : \Theta \times \Theta \mapsto \mathbb{R}$ is called the surrogate function

Assumption

The surrogate function $\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) : \Theta \times \Theta \mapsto \mathbb{R}$ satisfies the following conditions:

1. $\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t)$ *is strongly convex on space* Θ*;*
2. $\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t)$ *is differentiable with* $\nabla_{\boldsymbol{\theta}} \tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = \nabla_{\boldsymbol{\theta}} l(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^t}$

Vanilla SCA

- By performing the first-order Taylor expansion, we can make the convex function $h(\boldsymbol{\theta})$ affine and construct $\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t)$:

$$\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = g(\boldsymbol{\theta}) - h(\boldsymbol{\theta}^t) - \nabla_{\boldsymbol{\theta}} h(\boldsymbol{\theta}^t)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^t) \tag{7}$$

- The problem in Eq. (6) becomes a convex optimization problem, and can be solved effectively by using the commercial solver MOSEK [6, 2]
- The computational complexity in each iteration scales as $\mathcal{O}(Qn^3)$, where n is the number of training samples and Q is the number of basis kernels

Problem Statement

GSM Kernel: Multidimensional Input

- We can similarly use a Gaussian mixture to approximate the spectral density of the underlying kernel function,

$$S(\boldsymbol{\omega}) = \sum_{q=1}^{Q} \theta_q [\mathcal{N}(\boldsymbol{\omega}; \boldsymbol{\mu}_q, \boldsymbol{V}_q) + \mathcal{N}(\boldsymbol{\omega}; -\boldsymbol{\mu}_q, \boldsymbol{V}_q)], \tag{8}$$

where $\boldsymbol{\mu}_q = \left[\mu_q^{(1)}, ..., \mu_q^{(d_x)}\right]^\top$ and $\boldsymbol{V}_q = \operatorname{diag}\left(v_q^{(1)}, ..., v_q^{(d_x)}\right)$

- Taking the inverse Fourier transform of $S(\boldsymbol{\omega})$ yields the GSM kernel with multidimensional input as

$$k(\boldsymbol{\tau}) = \sum_{q=1}^{Q} \theta_q \cos\left(2\pi \boldsymbol{\tau}^\top \boldsymbol{\mu}_q\right) \prod_{p=1}^{d_x} \exp\left\{-2\pi^2 \tau_p^2 v_q^{(p)}\right\} \tag{9}$$

Grid Generation Strategy

- By assuming that each input dimension is independent, we empirically sample Q' frequencies, either uniformly or randomly, from the frequency region $[0, \mu_u^{(p)})$ for the p-th dimension and obtain $[\mu_1^{(p)}, \ldots, \mu_{Q'}^{(p)}]$
- The highest frequency $\mu_u^{(p)}$ is set to be equal to $1/2$ over the minimum input spacing between two adjacent training data points in the dimension p
- Using the sampled frequencies, we can generate $Q = Q'^{d_x}$ grid points in the $\mathbb{R}^{d_x}$ space
- Finally, using the generated grid points, we can construct Q isotropic multivariate Gaussian densities to approximate the underlying spectral density in the frequency domain

Proposed Algorithms

Distributed SCA (DSCA)

- To alleviate the computational burden owing to the curse of dimensionality, we leverage a multicore computing environment to optimize $\boldsymbol{\theta}$ in parallel
- The feasible set Θ in the GSM kernel admits a Cartesian product structure, i.e., $\Theta = \Theta_1 \times \Theta_2 \times \ldots \times \Theta_s$ with $\Theta_i \subseteq \mathbb{R}^{Q/s}$
- We can partition the optimization variable into s blocks, $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_s]^\top$, and construct a surrogate function that is additively separable in the blocks:

$$\tilde{l}(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = \sum_{i=1}^{s} \tilde{l}_i(\boldsymbol{\theta}_i, \boldsymbol{\theta}^t). \tag{10}$$

where

$$\tilde{l}_i(\boldsymbol{\theta}_i, \boldsymbol{\theta}^t) = g(\boldsymbol{\theta}_i, \boldsymbol{\theta}^t_{-i}) - h(\boldsymbol{\theta}^t) - \nabla_{\boldsymbol{\theta}_i} h(\boldsymbol{\theta}^t)^\top (\boldsymbol{\theta}_i - \boldsymbol{\theta}^t_i), \tag{11}$$

Distributed SCA (DSCA)

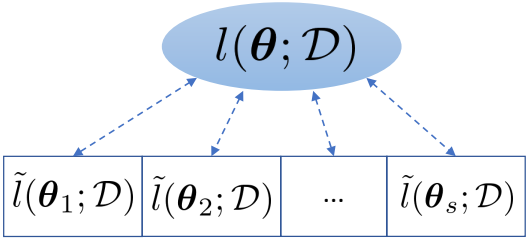

Figure: **Distributed SCA (DSCA)** for linear multiple kernel learning.

- The computational complexity of each computing core scales as $\mathcal{O}(\frac{Q}{s}n^3)$, where n is the number of training samples

Doubly Distributed SCA (D^2SCA)

- DSCA is impractical for big data and prone to data privacy issues
- We propose a doubly distributed algorithm based on the alternating direction method of multipliers (ADMM) [7, 8] which enables N multicore agents to collaboratively learn the global hyper-parameters while preserving the data privacy of the local agents
- Each agent optimizes the hyper-parameters using its local data and then exchanges the hyper-parameters with a central agent to reach a global consensus

Doubly Distributed SCA (D^2SCA)

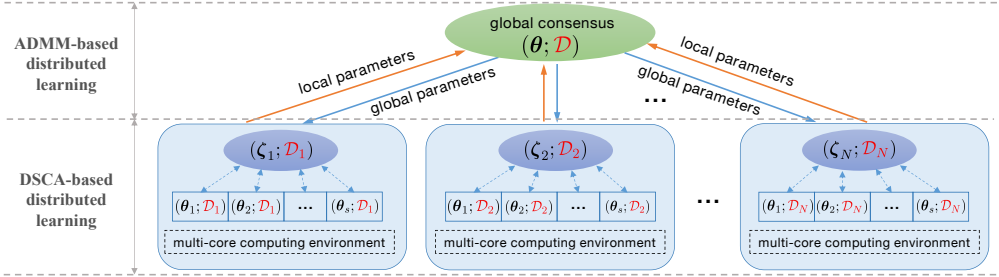

Figure: **Doubly Distributed SCA (D^2SCA)** for linear multiple kernel learning.

- The overall computational complexity of the D^2SCA algorithm scales as $\mathcal{O}(\frac{Qn^3}{sN^3})$

Results

Setup

- We investigate the training and prediction performance of the proposed learning algorithms, DSCA and D^2SCA, on various data sets
- We have selected 8 one-dimensional input data sets and 4 multidimensional input data sets as our benchmark
- We compare the proposed GSM kernel-based GP (GSMGP) with the SM kernel-based GP (SMGP) proposed by Wilson *et al.* in [9] and the squared-exponential kernel-based GP (SEGP) in terms of the prediction mean squared error (MSE)

1-D Case: Prediction Performance

Data Set	GSMGP DSCA	GSMGP D^2SCA	SMGP	SEGP	LSTM	ARIMA
ECG	**1.1E-02**	1.2E-02	1.9E-02	1.6E-01	1.6E-01	1.8E-01
CO2	**9.2E-01**	1.4E+00	1.1E+00	1.5E+03	2.9E+02	4.9E+00
Electricity	4.3E+03	**3.6E+03**	7.5E+03	8.3E+03	8.0E+03	1.2E+04
Employment	**5.4E+01**	7.0E+01	7.0E+02	8.4E+03	1.9E+03	3.9E+02
Hotel	**4.2E+02**	1.5E+03	2.8E+03	5.6E+04	5.0E+04	1.7E+04
Passenger	**6.9E+01**	1.1E+02	1.6E+02	8.8E+02	7.0E+02	4.5E+03
Clay	**8.5E+01**	2.4E+02	3.3E+02	1.5E+03	3.6E+02	3.3E+02
Unemployment	**2.0E+03**	3.1E+03	1.4E+04	5.6E+05	1.7E+05	1.5E+04

Table: Performance comparison between the proposed GSMGP and its competitors in terms of the **prediction MSE**.

1-D Case: Prediction Performance

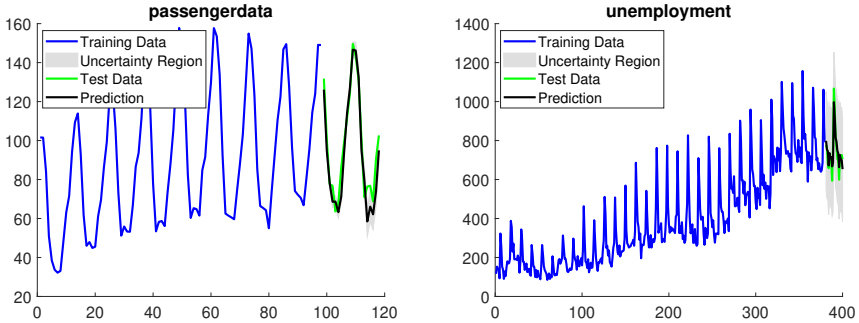

Figure: Training and prediction performance of the GSMGP with $\sigma = 0.001$ and $Q = 500$ uniformly generated grids. The optimal weights are solved via the distributed SCA (DSCA) algorithm.

1-D Case: Training Performance

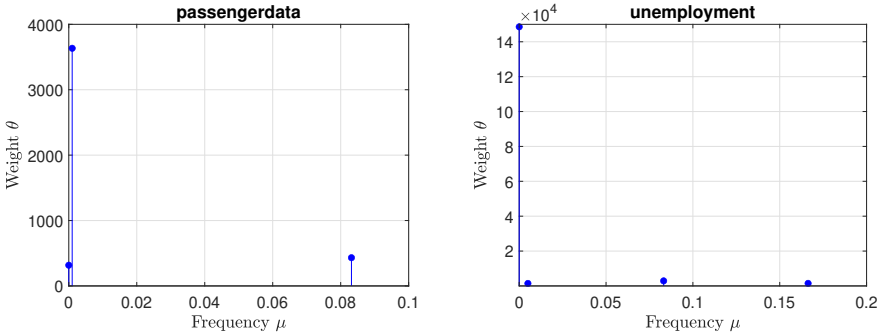

Figure: Estimated weights and frequencies generated by the distributed SCA (DSCA) algorithm for selected data sets.

M-D Case: Prediction Performance

Data Set	GSMGP DSCA	GSMGP D^2SCA	SMGP	SEGP	LSTM
ALE	2.4E-02	**2.3E-02**	3.8E-01	3.7E-02	3.4E-02
CCCP	1.9E+01	**1.6E+01**	2.1E+05	1.7E+01	2.8E+02
Airfoil	**1.7E+01**	5.2E+01	6.9E+01	7.7E+01	7.3E+01
Concrete	6.7E+01	**4.0E+01**	1.7E+03	1.3E+02	1.4E+02

Table: Performance comparison between the proposed GSMGP and its competitors, SMGP and SEGP, in terms of the **prediction MSE**.

M-D Case: Training Performance

Data Set	GSMGP-DSCA # of Iter.	GSMGP-D^{2}SCA # of Iter.	SMGP # of Iter.	SEGP # of Iter.
ALE	4	3	63	122
CCCP	5	7	82	141
Airfoil	7	5	500	137
Concrete	3	3	500	154

[1] The number of iterations in GSMGP-D^{2}SCA is the number of global iterations.

Table: Performance comparison between the proposed GSMGP and its competitors, SMGP and SEGP, in terms of the total number of iterations.

Conclusion

Conclusion

- We propose an extension of the grid spectral mixture (GSM) kernel for multidimensional input
- Two efficient distributed learning algorithms, namely DSCA and D^2SCA, are proposed to alleviate the computational complexity owing to the curse of dimensionality in the kernel hyper-parameter optimization
- The proposed algorithms can learn the global hyper-parameters with lower computational complexity and preserve data privacy during the learning process
- Experimental results verify that the proposed GSM kernel and the associated learning algorithms are superior in terms of training and prediction performance compared to their competitors

References I

- F. Yin, X. He, L. Pan, T. Chen, Z.-Q. Luo, and S. Theodoridis, "Sparse structure enabled grid spectral mixture kernel for temporal Gaussian process regression," in *Proc. Int. Conf. Inf. Fusion (FUSION)*, (Cambridge, UK), pp. 47–54, July 2018.
- F. Yin, L. Pan, T. Chen, S. Theodoridis, Z.-Q. Luo, and A. M. Zoubir, "Linear multiple low-rank kernel based stationary Gaussian processes regression for time series," *IEEE Trans. Signal Process.*, vol. 68, pp. 5260–5275, 2020.
- C. E. Rasmussen and C. I. K. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

References II

- G. Scutari and Y. Sun, "Parallel and distributed successive convex approximation methods for big-data optimization," in *Multi-agent Optimization*, pp. 141–308, Springer, 2018.
- MOSEK ApS, "MOSEK optimization toolbox for Matlab," *User's Guide and Reference Manual, Version*, vol. 4, 2019.
- S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, pp. 1–122, January 2011.
- Y. Xu, F. Yin, W. Xu, J. Lin, and S. Cui, "Wireless traffic prediction with scalable gaussian process: Framework, algorithms, and verification," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1291–1306, 2019.

References III

A. Wilson and R. P. Adams, "Gaussian process kernels for pattern discovery and extrapolation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, (Atlanta, USA), pp. 1067–1075, 2013.