NSF-Based Mixture of Gaussian Processes and Its Variational EM Algorithm

Xiangyang Guo, Daqing Wu, Tao Hong, and Jinwen Ma(✉)

Department of Information and Computational Sciences, School of Mathematical Sciences and LMAM, Peking University, Beijing 100871, China
{guoxy,wudq,paul.ht}@pku.edu.cn, jwma@math.pku.edu.cn

Abstract. Mixture of Gaussian processes (MGP) is a powerful model for dealing with data with multi-modality. However, input distributions of Gaussian process (GP) components of an MGP are designed to be Gaussians, which cannot model complex distributions that frequently appear in datasets. It has been proven that neural spline flow (NSF) models can transform a simple base distribution into any complex target distribution through the change of variables formula. In this paper, we propose an NSF-based mixture model of Gaussian processes (NMGP), which extends the conventional MGP by using distributions modeled by NSFs for the input variables instead of Gaussians. In addition, a variational EM algorithm is established to estimate the parameters of an NMGP. It is demonstrated by the experimental results that our proposed NMGP models outperform classic MGPs.

Keywords: Mixture of Gaussian processes · Neural spline flows · EM algorithm · Variational inference

1 Introduction

Gaussian process (GP) has proven to be an effective tool for a wide variety of applications in machine learning, such as modeling the inverse dynamics of a robot arm [1]. Unfortunately, GPs fail to handle multimodal datasets and, given N training samples, training a GP suffers from an expensive time complexity scaling as $O(N^3)$. To overcome these two limitations, Tresp [2] proposed the mixture of Gaussian processes (MGP), in which more than one GP is combined through a gating network. The gating network of an MGP is set to be a Gaussian mixture model (GMM) [2–5], in which each Gaussian determines the input distribution of a GP component. However, Gaussians cannot model complex distributions which frequently appear in datasets. Therefore, Yuan et al. [6] suggested replacing the Gaussians with GMMs. Although GMMs can alleviate the problem, we must determine the number of Gaussians in each GMM in advance. Inappropriate numbers may lead to bad predictive accuracy. In addition, Li et al.

T. Mantoro et al. (Eds.): ICONIP 2021, CCIS 1516, pp. 498–505, 2021.
https://doi.org/10.1007/978-3-030-92307-5_58

[7] proposed using Student's t-distributions as the input distributions to deal with multimodal data with overlaps between different modalities. Both GMMs and Student's t-distributions are parametric, which are not flexible enough to model a complex distribution.

One of the most important tasks in machine learning and statistics is to model an unknown probability distribution given samples drawn from that distribution, and normalizing flow models, proposed by Tabak et al. [8] and Tabak et al. [9], are widely applied to this problem [10]. Normalizing flow is an invertible differentiable function, which can transform a simple base distribution, e.g. $\mathcal{N}(\mathbf{0}, \mathbf{I})$, into any complex distribution through the change of variables formula.

In this paper, we propose a neural-spline-flow-based mixture model of Gaussian processes (NMGP), where the neural spline flows (NSF), to be introduced in Sect. 3, are a kind of normalizing flows. In an NMGP model, the input distributions are designed to be distributions generated via NSFs instead of Gaussians. A variational EM algorithm is developed to train our proposed NMGP models, and experimental results show that NMGPs not only produce smaller errors compared to the conventional MGPs but also explore the true input distributions.

The remainder of this paper is organized as follows. We briefly introduce GP and MGP models in Sect. 2. Section 3 describes the construction of NMGPs. The variational EM algorithm is described in Sect. 4. Then, we present the experimental results in Sect. 5 and Sect. 6 concludes this paper.

2 GP and MGP Models

First, a short description of GP models is given. A GP, expressed as $\{f(\mathbf{x})|\mathbf{x} \in \mathbb{X}\}$, is a collection of random variables, any finite number of which follow a multivariate Gaussian distribution [1], and it is completely specified by its mean function $m(\mathbf{x})$ and covariance function $c(\mathbf{x}, \mathbf{x}')$, where $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and $c(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$. In practice, we usually use a noisy GP, $\{y(\mathbf{x})|\mathbf{x} \in \mathbb{X}\}$, obtained by adding i.i.d. Gaussian noise to each $f(\mathbf{x})$, i.e. $y(\mathbf{x}) = f(\mathbf{x}) + \epsilon(\mathbf{x})$ with $\epsilon(\mathbf{x}) \sim \mathcal{N}(0, r^{-1})$, where r^{-1} denotes the noise level. We write a noisy GP as $y \sim \mathcal{GP}(m(\mathbf{x}), c(\mathbf{x}, \mathbf{x}') + r^{-1}\delta(\mathbf{x}, \mathbf{x}'))$, where $\delta(\mathbf{x}, \mathbf{x}')$ is the Kronecker delta function.

In this paper, we set $m(\mathbf{x})$ to be zero for simplicity and use the squared-exponential covariance function, defined as $c(\mathbf{x}, \mathbf{x}'|\boldsymbol{\sigma}) = \sigma_0^2 \exp\{-1/2 \sum_{i=1}^{d}(x_i - x_i')^2/\sigma_i^2\}$, where $\sigma_i > 0$ and d is the dimensionality of $\mathbf{x}$. In such a setting, given a training set $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ from a GP, we have

$$\mathbf{y}|\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{C} + r^{-1}\mathbf{I}_N), \tag{1}$$

where $\mathbf{C}(m, n) = c(\mathbf{x}_m, \mathbf{x}_n|\boldsymbol{\sigma})$.

Next, we briefly introduce MGP models. Here, suppose that $\mathcal{D}$ is drawn from an MGP composed of K GP components. We introduce an indicator variable τ_n for each sample $(\mathbf{x}_n, y_n)$. The generative process of $\mathcal{D}$ is given by:

$$p(\tau_n = k) = \pi_k, \quad \mathbf{x}_n|\tau_n = k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{2}$$

and

$$y_n|\tau_n = k \sim \mathcal{GP}(0, c(\mathbf{x}, \mathbf{x}'|\boldsymbol{\sigma}_k) + r_k^{-1}\delta(\mathbf{x}, \mathbf{x}')). \tag{3}$$

3 NSF-Based Mixture of Gaussian Processes

Let $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ be the base variable and $\mathbf{x}$ the target variable following the distribution we want to model. Suppose that $h(x|\boldsymbol{\theta})$ is a monotonic rational-quadratic spline [11] parameterized by $\boldsymbol{\theta}$, whose inverse and derivative are easily computed. Then a one-layer NSF[1] $\mathbf{z} = \mathbf{h}(\mathbf{x})$ is given by $z_1 = h(x_1|\boldsymbol{\theta}_1)$ and $z_i = h(x_i|\boldsymbol{\theta}_i(\mathbf{x}_{1:i-1})), i = 2, \ldots, d$, where $\boldsymbol{\theta}_i(\mathbf{x}_{1:i-1})), i = 2, \ldots, d$, are neural networks taking $\mathbf{x}_{1:i-1}$ as inputs [12]. The inverse, $\mathbf{x} = \mathbf{h}^{-1}(\mathbf{z})$, is $x_1 = h^{-1}(z_1|\boldsymbol{\theta}_1)$ and $x_i = h^{-1}(z_i|\boldsymbol{\theta}_i(\mathbf{x}_{1:i-1})), i = 2, \ldots, d$. Recursively, a J-layer NSF is defined as $\mathbf{z} = \boldsymbol{\varphi}(\mathbf{x}) = \mathbf{h}_J(\mathbf{h}_{J-1}(\ldots \mathbf{h}_1(\mathbf{x}) \ldots))$. It is obtained that $p_{\mathbf{x}}(\mathbf{x}) = p_{\mathbf{z}}(\boldsymbol{\varphi}(\mathbf{x})) \prod_{j=1}^{J} |\det(\partial \mathbf{x}_j / \partial \mathbf{x}_{j-1})|$, where $\mathbf{x}_j = \mathbf{h}_j(\mathbf{h}_{j-1}(\ldots \mathbf{h}_1(\mathbf{x}) \ldots))$, $j = 1, \ldots, J$, $\mathbf{x}_J = \mathbf{z}$, and $\mathbf{x}_0 = \mathbf{x}$. Computing the determinants is cheap because the Jacobians are lower triangular. Therefore, we can easily calculate the likelihood of $\mathbf{x}$.

In an NMGP, a base variable $\mathbf{z}_k$, subject to $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, and an NSF $\boldsymbol{\varphi}_k(\mathbf{x}|\boldsymbol{\omega}_k)$, where $\boldsymbol{\omega}_k$ denotes the parameters, are introduced for kth GP. Then, to build an NMGP, we only need to replace the $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ in Eq. (2) with the probability distribution of $\mathbf{x} = \boldsymbol{\varphi}_k^{-1}(\mathbf{z}_k|\boldsymbol{\omega}_k)$, i.e. $p(\mathbf{x}|\boldsymbol{\omega}_k) = p_{\mathbf{z}_k}(\boldsymbol{\varphi}_k(\mathbf{x}|\boldsymbol{\omega}_k))|\det(\partial \boldsymbol{\varphi}_k(\mathbf{x}|\boldsymbol{\omega}_k)/\partial \mathbf{x})|$.

4 Variational EM Algorithm

4.1 E-Step

Similar to the variational EM algorithm for MGP models, the linear GP model [6,13–15], an approximation of the standard GP, is used to eliminate the dependency between the outputs $y_n, n = 1, \ldots, N$. To construct a linear GP, we need to choose M ($\ll N$) samples from $\mathcal{D}$ to form a support set $\mathcal{D}'$. Let $\boldsymbol{\lambda}$ follow $\mathcal{N}(\mathbf{0}, \mathbf{C}_{MM}^{-1})$, where $\mathbf{C}_{MM}$ is the $M \times M$ covariance matrix consisting of covariance functions between samples in $\mathcal{D}'$. Then we assume that $y_n|\boldsymbol{\lambda} \sim \mathcal{N}(c(\mathbf{x}_n, \mathcal{D}')\boldsymbol{\lambda}, r^{-1})$. It follows that $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{NM}\mathbf{C}_{MM}^{-1}\mathbf{C}_{NM}^T + r^{-1}\mathbf{I}_N)$, where $\mathbf{C}_{NM}$ is composed of covariance functions between samples in $\mathcal{D}$ and $\mathcal{D}'$. That is identical to the likelihood of a sparse Gaussian process obtained via the subset of regressors (SoR) method [16], which shows the rationality of linear models. The most important advantage of linear models is that, given $\boldsymbol{\lambda}$, $y_n, n = 1, 2, \ldots, N$, are independent.

In order to develop a variational EM algorithm for NMGPs, we pick K support sets $\mathcal{D}_k, k = 1, 2, \ldots, K$, and introduce corresponding variables $\boldsymbol{\lambda}_k, k = 1, 2, \ldots, K$. Then, K linear GPs are built to take the place of K standard GPs. We give $\boldsymbol{\pi}$ a Dirichlet prior $\mathrm{Dir}(\boldsymbol{\alpha})$ and r_k a Gamma prior $\Gamma(a, b)$. The graphical model representation for the proposed model is shown in Fig. 1.

For simplicity, we denote all the latent variables as $\boldsymbol{\Gamma} = \{\pi_k, \boldsymbol{\lambda}_k, r_k, \tau_n | k = 1, 2, \ldots, K, n = 1, 2, \ldots, N\}$. In the framework of linear models, the complete data log-likelihood is given by

[1] In this paper, only autoregressive transforms are utilized.

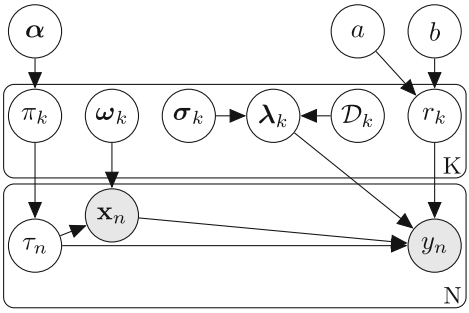

Fig. 1. The graphical model representation for the proposed model.

$$\begin{aligned}\ln p(\mathbf{\Gamma}, \mathbf{X}, \mathbf{y}) = \ln p(\boldsymbol{\pi}) + \sum\nolimits_{k=1}^{K} (\ln p(\boldsymbol{\lambda}_k) + \ln p(r_k)) + \sum\nolimits_{n=1}^{N} \ln p(\tau_n|\boldsymbol{\pi}) \\ + \sum\nolimits_{n=1}^{N} (\ln p(\mathbf{x}_n|\tau_n) + \ln p(y_n|\tau_n, \mathbf{x}_n, \boldsymbol{\lambda}_{\tau_n}, r_{\tau_n})).\end{aligned} \tag{4}$$

According to variational inference based on mean field theory, we choose a variational posterior distribution of the form

$$q(\mathbf{\Gamma}) = q(\boldsymbol{\pi}) \prod\nolimits_{k=1}^{K} \{q(\boldsymbol{\lambda}_k) q(r_k)\} \prod\nolimits_{n=1}^{N} q(\tau_n) \tag{5}$$

to approximate the true one in the E-step. Formulas used to calculate the above variational factors iteratively are standard [15] and thus omitted here.

4.2 M-Step

In an NMGP, the whole parameters, their current values, and the current variational posterior are denoted as $\mathbf{\Theta}$, $\mathbf{\Theta}_t$, and $q_t(\mathbf{\Gamma})$, respectively. The true Q-function, $Q(\mathbf{\Theta}|\mathbf{\Theta}_t) = \mathbb{E}_{p(\mathbf{\Gamma}|\mathbf{X},\mathbf{y},\mathbf{\Theta}_t)} \ln p(\mathbf{\Gamma}, \mathbf{X}, \mathbf{y}|\mathbf{\Theta})$, cannot be computed analytically. Therefore, we use $\hat{Q}(\mathbf{\Theta}|\mathbf{\Theta}_t) = \mathbb{E}_{q_t(\mathbf{\Gamma})} \ln p(\mathbf{\Gamma}, \mathbf{X}, \mathbf{y}|\mathbf{\Theta})$ as its approximation. We maximize $\hat{Q}(\mathbf{\Theta}|\mathbf{\Theta}_t)$ w.r.t. $\mathbf{\Theta}$ through gradient ascent methods. The deep learning framework PyTorch is utilized so there is no need to calculate the gradient manually.

4.3 Updating Support Sets

The first step of the variational EM algorithm is to determine K support sets randomly, and then a certain number, denoted as S, of E-steps and M-steps are performed. After that, to improve the result, the K support sets are updated according to certain criteria, and E-steps and M-steps are conducted again. We repeat the process T times determined in advance.

Fixing the parameters and variational posteriors of $\mathbf{\Gamma}\backslash\{\boldsymbol{\lambda}_k\}_{k=1}^K$, it is reasonable to assume that the best $\mathcal{D}_k$ maximizes the value of $q(\boldsymbol{\lambda}_k)$ at the mean [6,13–15]. That is equivalent to maximizing

$$|\mathbf{T}_k| = |\mathbf{C}_k + \mathbb{E}_q r_k \sum\nolimits_{n=1}^{N} q_{nk} c_k(\mathbf{x}_n) c_k(\mathbf{x}_n)^T|, \tag{6}$$

where $\mathbf{C}_k$ is the covariance matrix composed of covariance functions between data points in $\mathcal{D}_k$ and $c_k(\mathbf{x}_n)$ is the column vector consisting of covariance functions between $\mathbf{x}_n$ and training samples in $\mathcal{D}_k$. $|\mathbf{T}_k|$ can be thought of as a function of $\mathcal{D}_k$.

Finding the best $\mathcal{D}_k$ from $\binom{N}{M}$ candidates is an NP problem. Therefore, a greedy algorithm [6,13–15] is employed to find a suboptimal solution.

4.4 Predictive Distribution

Assume that $\mathbf{x}^*$ is a new input and y^* represents its predictive output. We need to calculate the distribution $p(y^*|\mathbf{x}^*, \mathcal{D}) = \int p(y^*|\mathbf{x}^*, \mathbf{\Gamma}) p(\mathbf{\Gamma}|\mathcal{D}) \mathrm{d}\mathbf{\Gamma}$. Similar to the calculation of Q-function, we can replace $p(\mathbf{\Gamma}|\mathcal{D})$ with the variational posterior $q(\mathbf{\Gamma})$ obtained in the last E-step. Thus, we have

$$\begin{aligned} p(y^*|\mathbf{x}^*, \mathcal{D}) &\approx \int p(y^*|\mathbf{x}^*, \mathbf{\Gamma}) q(\mathbf{\Gamma}) \mathrm{d}\mathbf{\Gamma} \approx p(y^*|\mathbf{x}^*, \mathbb{E}_q \mathbf{\Gamma}) \\ &= \sum\nolimits_k \frac{\mathbb{E}_q \pi_k p(\mathbf{x}^*|\boldsymbol{\omega}_k)}{\sum_i \mathbb{E}_q \pi_i p(\mathbf{x}^*|\boldsymbol{\omega}_i)} \mathcal{N}(c_k(\mathbf{x}^*)^T \mathbb{E}_q(\boldsymbol{\lambda}_k), (\mathbb{E}_q r_k)^{-1}), \end{aligned} \tag{7}$$

which is the weighted sum of K independent Gaussian distributions.

5 Experiments

In this section, experiments on three datasets are presented. In all experiments, $\boldsymbol{\alpha}$, a and b are set to be $(1, 1, \ldots, 1)^T$, 0.01 and 0.0001, respectively [6]. In addition, we set $T = 10$ and $S = 10$. We use the root mean squared error (RMSE) to measure the prediction accuracy. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^I$ be a test set and $\{\hat{y}_i\}_{i=1}^I$ the set of predictive values, the RMSE is defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{I} \sum\nolimits_{i=1}^{I} (y_i - \hat{y}_i)^2}. \tag{8}$$

It is demonstrated by the results to be discussed that our proposed NMGP models outperform the conventional MGP models.

5.1 Synthetic Dataset

This synthetic dataset, shown in Fig. 2, is drawn from the following three functions:

$$\begin{aligned} f_0(x) &= -2.5 \sin(2\pi x/12.5),\ x \sim 0.5\mathcal{N}(-12.5, 1.2^2) + 0.5\mathcal{N}(-7.5, 1.5^2), \\ f_1(x) &= 6 \sin(2\pi x/10),\ x \sim 0.5\mathcal{N}(-2.5, 1.5^2) + 0.5\mathcal{N}(2.5, 1.5^2), \\ f_2(x) &= -4.5 \sin(2\pi x/15),\ x \sim 0.5\mathcal{N}(7.5, 1.5^2) + 0.5\mathcal{N}(12.5, 1.2^2). \end{aligned} \tag{9}$$

The training set contains 300 points, in which points from three functions account for 30%, 40%, and 30%, respectively. The test set consists of 300 points evenly distributed in $[-15, 15]$. Gaussian noise, following $\mathcal{N}(0, 0.3^2)$, is added to all samples in this dataset.

We employ an NMGP consisting of three GP components to model the dataset and set $M = 30$ and $Q = 30$. The predictive curve, pointwise standard deviations, and three learned input distributions, similar to the true ones, are plotted in Fig. 2. The RMSEs of our NMGP and baseline models are shown in Table 1. The result demonstrates that our proposed NMGP not only caused smaller error but also produced more proper approximations of the true input distributions.

Table 1. RMSEs on three datasets

	Synthetic dataset	Coal gas dataset	Motorcycle dataset
MGP (MCMC EM) [5]	0.5411	0.6011	24.2000
TMGP (Hard-cut EM) [7]	0.5322	0.5939	23.8888
TMGP (Variational EM) [15]	0.5245	0.5854	21.6147
Our NMGP	**0.5139**	**0.5799**	**21.5936**

5.2 Coal Gas Dataset

The proposed NMGP model was also applied to a coal gas dataset [7,15], where the training set consists of 200 samples and the test set consists of 113 samples. Here, we set $K = 4$, $M = 20$ and $Q = 20$ following Guo et al. [15]. We display the dataset and the predictions in Fig. 2, and the RMSEs in Table 1.

5.3 Motorcycle Dataset

Finally, we employed the NMGP model to fit the motorcycle dataset [5,15] composed of 133 samples. The whole dataset is used to train the model and compute the RMSEs. Figure 2 shows the dataset and the result of our proposed model, and the RMSEs are presented in Table 1.

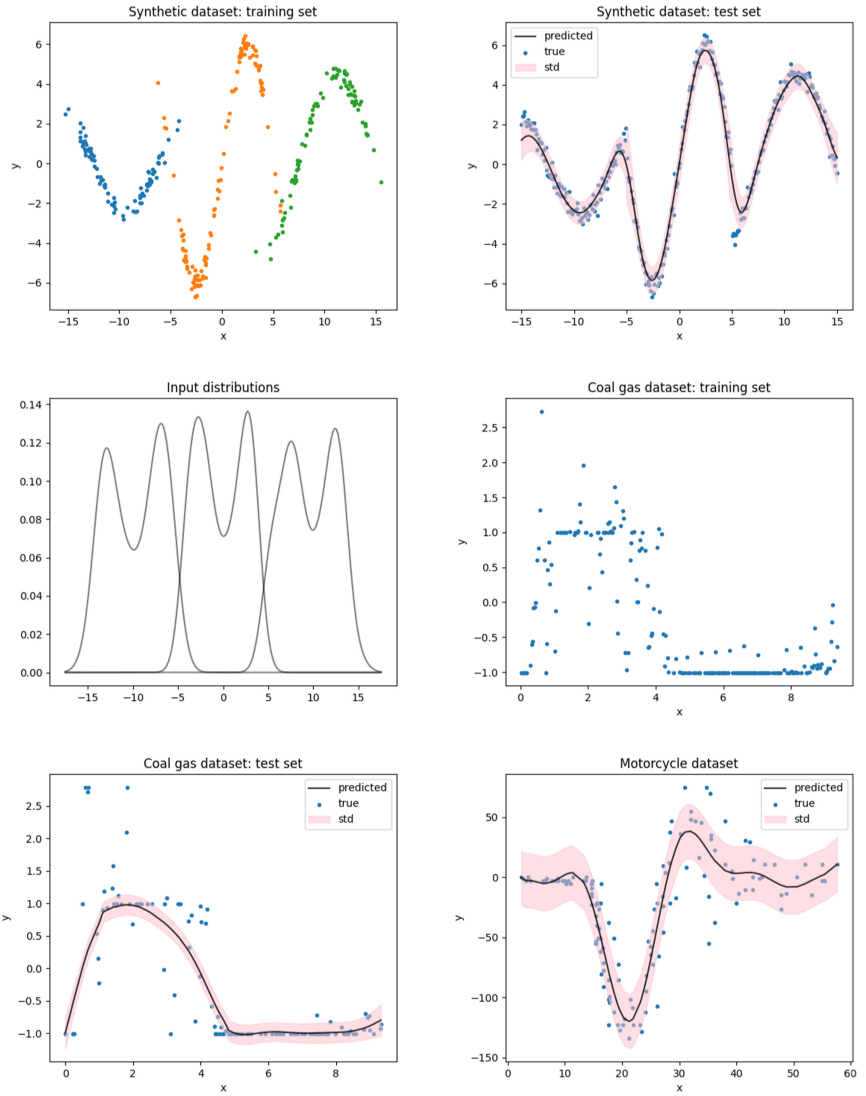

Fig. 2. Three datasets

6 Conclusion

We have proposed an NSF-based mixture model of Gaussian processes and developed a variational EM algorithm for training this model. NMGPs model input distributions using NSF models instead of Gaussians. It is demonstrated by the experimental results on three datasets that our proposed NMGP models outperform MGPs. However, the variational EM algorithm cannot be applied to large datasets since the calculation of the variational posterior involves the sum

of N terms. Thus, an algorithm with lower computational complexity should be developed in the future, e.g. hard-cut EM algorithm.

Acknowledgment. This work is supported by the National Key R & D Program of China (2018AAA0100205).

References

1. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
2. Tresp, V.: Mixtures of Gaussian processes. In: Advances in Neural Information Processing Systems, vol. 13, pp. 654–660 (2001)
3. Yang, Y., Ma, J.: An efficient EM approach to parameter learning of the mixture of Gaussian processes. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) ISNN 2011, Part II. LNCS, vol. 6676, pp. 165–174. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21090-7_20
4. Chen, Z., Ma, J., Zhou, Y.: A precise hard-cut EM algorithm for mixtures of Gaussian processes. In: Huang, D.-S., Jo, K.-H., Wang, L. (eds.) ICIC 2014. LNCS (LNAI), vol. 8589, pp. 68–75. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09339-0_7
5. Wu, D., Chen, Z., Ma, J.: An MCMC based EM algorithm for mixtures of Gaussian processes. In: Hu, X., Xia, Y., Zhang, Y., Zhao, D. (eds.) ISNN 2015. LNCS, vol. 9377, pp. 327–334. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25393-0_36
6. Yuan, C., Neubauer, C.: Variational mixture of Gaussian process experts. In: Advances in Neural Information Processing Systems, vol. 21, pp. 1897–1904 (2009)
7. Li, X., Li, T., Ma, J.: The unν-hardcut EM algorithm for non-central student-t mixtures of Gaussian processes. In: 15th IEEE International Conference on Signal Processing (ICSP), pp. 289–294 (2020)
8. Tabak, E.G., Vanden-Eijnden, E.: Density estimation by dual ascent of the log-likelihood. Commun. Math. Sci. **8**(1), 217–233 (2010)
9. Tabak, E.G., Turner, C.V.: A family of nonparametric density estimation algorithms. Commun. Pure Appl. Math. **66**(2), 145–164 (2013)
10. Kobyzev, I., Prince, S.J.D., Brubaker, M.A.: Normalizing flows: an introduction and review of current methods. IEEE Trans. Pattern Anal. Mach. Intell. **43**, 3964–3979 (2020)
11. Gregory, J., Delbourgo, R.: Piecewise rational quadratic interpolation to monotonic data. IMA J. Numer. Anal. **2**(2), 123–130 (1982)
12. Durkan, C., Bekasov, A., Murray, I., Papamakarios, G.: Neural spline flows. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
13. Sun, S., Xu, X.: Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction. IEEE Trans. Intell. Transp. Syst. **12**(2), 466–475 (2011)
14. Luo, C., Sun, S.: Variational mixtures of Gaussian processes for classification. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, vol. 26, pp. 4603–4609 (2017)
15. Guo, X., Li, X., Ma, J.: Variational EM algorithm for Student-t mixtures of Gaussian processes. In: International Conference on Intelligent Computing (2021)
16. Smola, A.J., Bartlett, P.: Sparse greedy Gaussian process regression. In: Advances in Neural Information Proceeding System, vol. 13, pp. 619–625 (2001)