


Article

An Improved Mixture Model of Gaussian Processes and Its Classification Expectation–Maximization Algorithm

Yurong Xie ^{1,†}, Di Wu ^{2,*,†}  and Zhe Qiang ³¹ School of Mathematics and Statistics, Shaanxi Normal University, Xi'an 710119, China² School of Computer Science, Shaanxi Normal University, Xi'an 710119, China³ School of Mathematics, Northwest University, Xi'an 710127, China

* Correspondence: wudi2@snnu.edu.cn

† These authors contributed equally to this work.

Abstract: The mixture of experts (ME) model is effective for multimodal data in statistics and machine learning. To treat non-stationary probabilistic regression, the mixture of Gaussian processes (MGP) model has been proposed, but it may not perform well in some cases due to the limited ability of each Gaussian process (GP) expert. Although the mixture of Gaussian processes (MGP) and warped Gaussian process (WGP) models are dominant and effective for non-stationary probabilistic regression, they may not be able to handle general non-stationary probabilistic regression in practice. In this paper, we first propose the mixture of warped Gaussian processes (MWGP) model as well as its classification expectation–maximization (CEM) algorithm to address this problem. To overcome the local optimum of the CEM algorithm, we then propose the split and merge CEM (SMCEM) algorithm for MWGP. Experiments were done on synthetic and real-world datasets, which show that our proposed MWGP is more effective than the models used for comparison, and the SMCEM algorithm can solve the local optimum for MWGP.

Keywords: mixture of experts; warped Gaussian process; classification expectation–maximization algorithm; local optimum; non-stationary probabilistic regression

MSC: 68T05

**Citation:** Xie, Y.; Wu, D.; Qiang, Z.

An Improved Mixture Model of Gaussian Processes and Its Classification Expectation–Maximization Algorithm.

Mathematics **2023**, *11*, 2251. <https://doi.org/10.3390/math11102251>

Academic Editor: Bo Wang

Received: 4 April 2023

Revised: 4 May 2023

Accepted: 6 May 2023

Published: 11 May 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The mixture of experts (ME) model is effective for multimodal data in statistics and machine learning [1]. In ME, the input space is softly divided into multiple regions by an input-dependent gating function, and each region is specified by an expert. Due to the diversity of experts, such as Gaussian distribution [2] and the support vector machine (SVM) [3], there are a variety of models based on the ME framework.

In the 2000s, Tresp constructed the mixture of Gaussian processes (MGP) model, a special ME where each expert is a stationary Gaussian process (GP) probabilistic regression model, by mixing GPs along the input space to treat non-stationary probabilistic regression [4–8]. Popular gating functions of the MGP include logistic distribution and Gaussian distribution. For learning MGP, the main algorithms are Markov Chain Monte Carlo (MCMC), variational Bayesian (VB), and expectation–maximization (EM). The MGP cannot work correctly on non-stationary probabilistic regression in some situations since the ability of each GP expert is limited.

In this paper, we propose the mixture of warped Gaussian processes (MWGP) model, which has more flexible and attractive properties than MGP to handle non-stationary probabilistic regression, by modeling each component of an MGP with a warped Gaussian process (WGP); these WGPs are combined in the input space by the Gaussian distribution. The WGP is capable of modeling non-stationary probabilistic regression by learning a nonlinear distortion (also called warping) of the GP outputs. The MWGP can be viewed

as a generalization of the MGP and WGP frameworks, and it allows for dealing with non-stationary data in each multimodal mode. In MWGP, the mixture parameter, warping function parameter, covariance function parameter, and indicator variable (regarded as the latent variable) are considered simultaneously. To handle this, we designed the classification expectation–maximization (CEM) algorithm for the training of MWGP. However, the CEM algorithm may easily converge to a local optimum for MWGP in some cases. We propose the split and merge CEM (SMCEM) algorithm of MWGP based on the SMCEM algorithm of the MGP and the CEM algorithm of MWGP to solve this problem. Experiments were conducted on synthetic and real-world datasets, and the results demonstrate the feasibility and superior accuracy of our proposed MWGP model trained using the CEM algorithm compared to other comparative models for probabilistic regression. Moreover, the SMCEM algorithm of MWGP can overcome local optima in some datasets at a negligible time cost.

The remainder of this paper is organized as follows. In Section 2, we present related works of GP, MGP, and WGP models. We describe GP, WGP, and our proposed MWGP in Section 3. In Section 4, we present our proposed SMCEM algorithm of MWGP, the CEM algorithm of MWGP, and the partial CEM algorithm of MWGP. The experimental results are presented in Section 5, and the conclusions are drawn in Section 6.

2. Related Works

Related works of the GP. The GP is a versatile tool for probabilistic regression and it has been successfully applied to practical fields such as time series prediction [9] and signal processing [10]. The non-stationary probabilistic regression problem exists widely, but it cannot be modeled effectively by the conventional GP model [11,12]. To solve this, the non-stationary GP model was proposed by introducing a robust and flexible covariance structure [13–16]. However, a single GP cannot handle the non-stationary probabilistic regression well due to its inherent simplicity.

Related works of the MGP model. The structure of the MGP is shown in Figure 1. As seen in this figure, the MGP is a more effective non-stationary model than the GP. However, the parameter estimation of the MGP is a challenge due to the unknown indicator variable (regarded as the latent variable) and the highly correlated sample [17–20]. The Markov Chain Monte Carlo (MCMC) method employing Gibbs sampling and hybrid Monte Carlo approximates the intractable integration and summation by the simulated sample of the indicator variable and parameter [6,7,21–23], and it is commonly used in a system of partial differential equations [24,25]. The MCMC generally obtains precise results, but it takes a long time to generate the simulated sample. To improve the efficiency, the variational Bayesian (VB) inference and the expectation–maximization (EM) algorithm were proposed. Ross and Dy constructed the VB inference on the basis of the mean-field approximation, where the indicator variable and the stochastic parameter are forced to be independent of the probability distribution [26]. Yuan and Neubauer established a variational EM algorithm by a similar mean-field method as the VB inference [8]. Then, the leave-one-out cross-validation (LOOCV) EM algorithm was proposed based on the LOOCV approximation method, where the probability density of the GP is approximated by the production of LOOCV probability densities [27]. To improve the accuracy, Chen et al. constructed the CEM algorithm by replacing the expectation step (E step) of the conventional EM algorithm with a classification–expectation step (CE step) [28,29]. For the CEM algorithm, samples are classified into components by the maximum a posteriori (MAP) principle of indicator variables in the CE step and the parameters of components are learned independently in the maximization step (M step). Then, the MCMC EM algorithm was designed by approximating the Q-function of the EM algorithm with the Gibbs sampling; this algorithm is generally accurate but slow [30–32]. The SMCEM algorithm was constructed by combining the split and merge EM algorithm and the CEM algorithm to address the local optimum of the CEM algorithm for MGP [33–36]. Regarding the model selection problem, i.e., selecting the number of components, the Dirichlet process as the gating function was developed [6,17,21,22]; moreover, Zhao and Ma proposed a synchronous balancing criterion [37]. Regarding the

robustness problem, the robust MGP with the Laplace noise and the robust MGP with the student- t noise were proposed to overcome this difficulty [38].

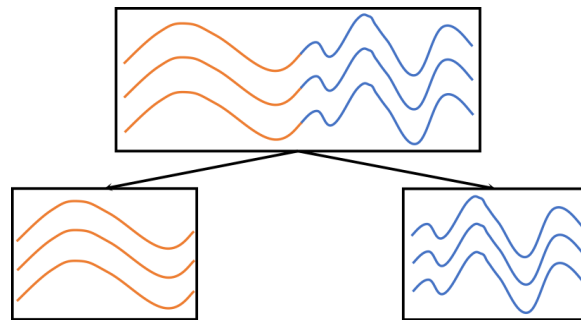


Figure 1. The diagram structure of the MGP model: the low-layer consists of two GPs and the high-layer structure consists of an MGP. The curves are divided into two curve segments along the input space marked in different colors, and curve segments of one color corresponding to a GP.

Related works of the WGP model. The WGP trained by the maximum likelihood estimation (MLE) method can handle non-stationary probabilistic regression effectively by transforming the GP output in a latent space to the real output in the observation space with a learnable nonlinear monotonic function [39], as seen in Figure 2. From this figure, it is clear that the WGP is much better than the GP, and the performance of the GP degenerates dramatically. In WGP, such a preprocessing transformation can be considered as an integral part of non-stationary probabilistic modeling. The improved WGP models involve a large number of parameters and hyperparameters, which limits the applicability of the WGP. The Hamiltonian MCMC method [40] and the VB inference [41] were proposed to improve the training of the WGP in some situations. To make the WGP structure flexible, Rios et al. constructed the WGP with a deep compositional architecture warping function [42], and the multi-task WGP was proposed [43]. For the optimization of the WGP, a spatial branching strategy was designed [44]. The WGP (being a useful generalization of the GP) has also been widely used in practical applications [45–48].

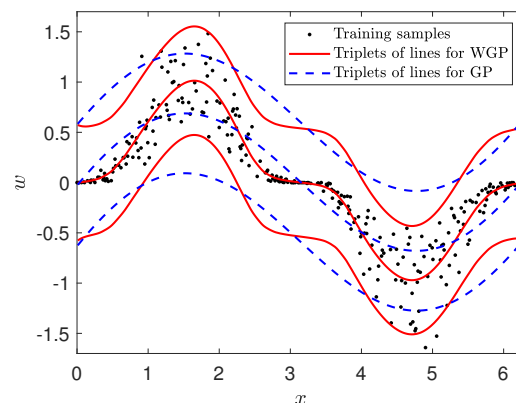


Figure 2. An example of a non-stationary data regression task. The one-dimensional data are generated by adding Gaussian noise to a sine function. The dataset contains 300 training samples and 600 test samples. These samples are then warped by the function $w = y^3$. The mean and two standard deviation (SD) bounds are represented by triplets of lines.

3. Model Construction

In this section, we first introduce the GP and WGP models and then describe our proposed MWGP model.

3.1. The GP

The GP is a non-parametric statistical model, which is described briefly as follows. For a dataset $\{\mathbf{x}_n \in \mathbb{R}^E, y_n \in \mathbb{R}\}_{n=1}^N$, the standard Gaussian process (GP) model for probabilistic regression is defined by

$$y_n = f(\mathbf{x}_n) + \varepsilon_n \quad \text{and} \quad \varepsilon_n \sim N(0, \sigma^2), \tag{1}$$

where $f(\bullet)$, y_n , \mathbf{x}_n , ε_n , and σ are the random latent function described, the n -th output, the n -th $E \times 1$ input, the n -th Gaussian noise, and the SD of the Gaussian noise, respectively. The random latent function $f(\bullet)$ on $\{\mathbf{x}_n\}_{n=1}^N$ is subject to a Gaussian distribution

$$p(\mathbf{f}|\mathbf{X}) = N(\boldsymbol{\mu}, \mathbf{K}),$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$ is the $E \times N$ input matrix, $\mathbf{f} = [f_n]_{N \times 1}$ is the latent vector, $f_n = f(\mathbf{x}_n)$, $\boldsymbol{\mu} = [\mu_n]_{N \times 1}$ is the mean vector, $\mu_n = \mu(\mathbf{x}_n)$ is the mean function, $\mathbf{K} = [K_{n\bar{n}}]_{N \times N}$ is the covariance matrix, and $K_{n\bar{n}} = K(\mathbf{x}_n, \mathbf{x}_{\bar{n}}; \boldsymbol{\gamma})$ is the covariance function (In this paper, we use the squared exponential covariance function

$$K(\mathbf{x}_n, \mathbf{x}_{\bar{n}}; \boldsymbol{\gamma}) = (\gamma^{(1)})^2 \exp[-(\mathbf{x}_n - \mathbf{x}_{\bar{n}})^T \boldsymbol{\Lambda} (\mathbf{x}_n - \mathbf{x}_{\bar{n}})/2],$$

where $\boldsymbol{\Lambda} = \text{diag}(1/(\gamma^{(2)})^2, 1/(\gamma^{(3)})^2, \dots, 1/(\gamma^{(E+1)})^2)$ is the $E \times E$ diagonal matrix) parameterized by $\boldsymbol{\gamma} = [\gamma^{(1)}, \gamma^{(2)}, \dots, \gamma^{(E+1)}]$

The likelihood function of the GP is obtained by integrating $p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})$ with respect to \mathbf{f} , given by

$$p(\mathbf{y}|\mathbf{X}) = N(\boldsymbol{\mu}, \mathbf{K} + \sigma^2 \mathbf{I}_N), \tag{2}$$

where $p(\mathbf{y}|\mathbf{f})$ is the independent identically distributed Gaussian distribution obtained by Equation (1), $\mathbf{y} = [y_n]_{N \times 1}$ is the output vector, and \mathbf{I}_N is the $N \times N$ unit matrix.

3.2. The WGP Model

In the non-stationary probabilistic regression problem, the WGP model describes the real output in the observation space as a parametric nonlinear transformation of the GP. For the dataset $\mathcal{D} = \{\mathbf{x}_n \in \mathbb{R}^E, w_n \in \mathbb{R}\}_{n=1}^N$, the WGP is constructed by introducing a latent variable set $\{y_n \in \mathbb{R}\}_{n=1}^N$, where w_n and \mathbf{x}_n are the n -th output in the observation space and the n -th $E \times 1$ input, respectively. The latent vector $\mathbf{y} = [y_n]_{N \times 1}$ is subject to a GP with a zero-mean function (i.e., $\mu(\mathbf{x}_n) = 0$), defined by Equation (2)

$$p(\mathbf{y}|\mathbf{X}) = N(\mathbf{0}_{N \times 1}, \mathbf{K} + \sigma^2 \mathbf{I}_N).$$

The latent variable y_n is transformed to w_n by a monotonic warping function (In this paper, we assume the warping function to be a feedforward neural network $g(w_n; \boldsymbol{\Omega}) = w_n + \sum_{j=1}^J a_j \tanh(h_j(w_n + l_j))$, where a_j and h_j are non-negative for any j to ensure monotonicity, J is the number of neurons, and $\boldsymbol{\Omega} = [a_1, a_2, \dots, a_J, h_1, h_2, \dots, h_J, l_1, l_2, \dots, l_J]^T$). $g(\bullet; \boldsymbol{\Omega})$

$$y_n = g(w_n; \boldsymbol{\Omega}),$$

where $g(w_n; \boldsymbol{\Omega})$ maps w_n to the entire real line, and $\boldsymbol{\Omega}$ is the parameter vector composed of J neurons. As stated above, the established WGP is fully incorporated into the probabilistic framework of the GP.

For convenience, the WGP is denoted as

$$\mathbf{w} \sim \text{WGP}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\Omega}), \tag{3}$$

where $\mathbf{w} = [w_n]_{N \times 1}$ is the output vector and $\boldsymbol{\theta} = \{\sigma, \gamma^{(1)}, \gamma^{(2)}, \dots, \gamma^{(E+1)}\}$. For WGP, the information flow direction is $\mathbf{x}_n \rightarrow y_n \rightarrow w_n$, and the relationships between the main variables are shown in Figure 3. From this figure, outputs $\{w_n \in \mathbb{R}\}_{n=1}^N$ are conditionally

independent when the strongly correlated $\{f_n\}_{n=1}^N$ are given. The parameters θ and Ω are learned jointly using a conjugate gradient method for WGP.

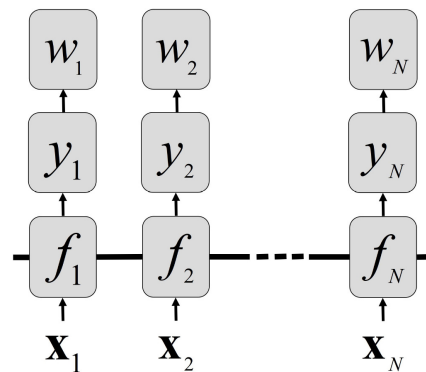


Figure 3. A diagram showing the relations among the main variables in the WGP model.

3.3. The MWGP Model

To process multimodal data while modeling the non-stationary nature of each mode, we describe our proposed MWGP model mathematically next, where C -different WGP components are mixed in the input region. The structure of MWGP is similar to that of the MGP, as illustrated in Figure 1. Compared to the MGP, the two-layer structure of MWGP can address non-stationary probabilistic regression in different ways.

A subscript c is inserted in the preceding notation for the number of components. The n -th sample (\mathbf{x}_n, w_n) is allocated to the c -th WGP component by an indicator variable z_{nc} (regarded as a latent variable), where $c = 1, 2, \dots, C$. If (\mathbf{x}_n, w_n) is in the c -th component, then $z_{nc} = 1$; otherwise, $z_{nc} = 0$. The distribution of the indicator variable vector $\mathbf{z}_n = [z_{n1}, z_{n2}, \dots, z_{nC}]^T$ is given by

$$P(\mathbf{z}_n = \mathbf{e}_c) = \eta_c, \tag{4}$$

where \mathbf{e}_c is the c -th column of the $C \times C$ unit matrix \mathbf{I}_C and $\sum_{c=1}^C \eta_c = 1$.

The distribution of the input vector \mathbf{x}_n is given by

$$p(\mathbf{x}_n | \mathbf{z}_n = \mathbf{e}_c) = N(\boldsymbol{\alpha}_c, \boldsymbol{\Sigma}_c), \tag{5}$$

where $\boldsymbol{\alpha}_c$ and $\boldsymbol{\Sigma}_c$ are the $E \times 1$ mean vector and the $E \times E$ covariance matrix of the Gaussian distribution, respectively. Equation (5) is commonly used in most generative mixture models.

After the distributions of Equations (4) and (5) are given, the distribution of the output vector \mathbf{w}_c is given based on Equation (3) by

$$\mathbf{w}_c \sim \text{WGP}(\mathbf{X}_c; \boldsymbol{\theta}_c, \boldsymbol{\Omega}_c), \tag{6}$$

where \mathbf{X}_c is the $E \times N_c$ input matrix composed of $\{\mathbf{x}_n | \mathbf{z}_n = \mathbf{e}_c; n = 1, 2, \dots, N\}$ in which $N_c = \sum_{n=1}^N z_{nc}$ is the sample number, \mathbf{w}_c is the $N_c \times 1$ output vector composed of $\{w_n | \mathbf{z}_n = \mathbf{e}_c; n = 1, 2, \dots, N\}$, $\boldsymbol{\Omega}_c$ parameterizes the warping function of the c -th component, and $\boldsymbol{\theta}_c = \{\sigma_c, \gamma_c^{(1)}, \gamma_c^{(2)}, \dots, \gamma_c^{(E+1)}\}$. For MWGP, C WGP components are independent, and each component is defined by Equation (6). MWGP is generally more flexible than the MGP and WGP; its information flow direction is $\mathbf{z}_n \rightarrow \mathbf{x}_n \rightarrow y_n \rightarrow w_n$, as shown in Figure 4. If $C = 1$, then the MWGP degenerates to WGP; if $g(w_n; \boldsymbol{\Omega}_c) = w_n$, then the MWGP degenerates to MGP.

With the above analysis, the mixture structure, the covariance function, and the warping function are incorporated simultaneously in the same probabilistic model framework. The computational cost of MWGP is similar to MGP since the time complexity of the inverse

covariance matrix operation in MWGP is the same as that in MGP, i.e., $O(N^3/C^2)$. For MWGP, there is an overfitting problem when too many extra parameters are added.

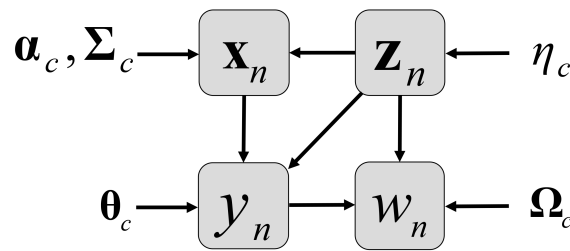


Figure 4. The probabilistic graphical model of the MWGP model: the elements inside the boxes are main variables and the others are parameters.

4. Algorithm Design

To handle the large time complexity of calculating the Q-function in the conventional EM algorithm of MWGP, we designed the CEM and partial CEM algorithms. In the CEM algorithm of MWGP, a local optimum is generally generated when there is a large separation between two parts of the MWGP component in practice. This separation is created by having many components of MWGP in one region and few in another. To escape from this separation, simultaneous split and merge operations were performed repeatedly by merging two similar components in a region with many components and splitting one component in another region with few components. As the CEM algorithm of MWGP can sometimes get trapped in a local optimum, we developed the SMCEM algorithm of MWGP to address this issue. The CEM algorithm of MWGP and the partial CEM algorithm of MWGP are sub-algorithms of the SMCEM algorithm for MWGP.

4.1. Procedures of the Proposed Algorithms

Denote the $C \times N$ indicator variable matrix $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N]$ and the whole parameter set $\Phi = \{\Phi_c\}_{c=1}^C$, where $\Phi_c = \{\eta_c, \alpha_c, \Sigma_c, \theta_c, \Omega_c\}$. For MWGP, procedures of the SMCEM algorithm, the CEM algorithm, and the partial CEM algorithm are presented in Algorithms 1–3.

The k-means clustering method is adopted in the first step of Algorithm 1 since it is possible for some samples to belong to the same component if they are close in distance. In the second and third steps of Algorithm 1, the split candidate set $\{t\}$ and the merge candidate set $\{u, s\}$ are sorted by split and merge criteria, respectively. By renumbering the merge and split candidate sets, we obtain the candidate set $\{u, s, t\}$. In the fifth step of Algorithm 1, we perform the partial CEM algorithm to retrain the parameters of new components and ensure that all other components are not affected by this retraining. The CEM algorithm is performed as a full training procedure for all components in the sixth step of Algorithm 1. In the seventh step of Algorithm 1, it is obvious that the accepted split or merge operation attempts to increase the value of the approximated Q-function in each iteration. We set hyperparameters C, J , and D according to the best RMSE (the root mean square error (RMSE) is used to characterize the accuracy of the model, which is defined mathematically by $\sqrt{\sum_{n=1}^N (y_n - \hat{y}_n)^2 / N}$, where \hat{y}_n is the estimation of y_n). In Algorithm 1, components of MWGP with poor aggregation are divided in the split operation, and those with high similarity are combined in the merge operation. Simultaneous split and merge operations can perform a global search by crossing over low-likelihood positions.

Algorithm 1 The SMCEM algorithm for MWGP

Input: \mathcal{D}, C, J .

Output: Φ_{best}, L_{best} .

- 1: **Initialization:** Initialize the indicator variable matrix $\mathbf{Z}^{(0)}$ by the k-means clustering on the input set $\{\mathbf{x}_n\}_{n=1}^N$, and obtain the indicator variable matrix $\mathbf{Z}^{(1)}$ and the parameter set $\Phi^{(0)}$ by performing the CEM algorithm described in Algorithm 2. Set the number of current iterations as $r = 1$, and $L_{best} = -\infty$.
 - 2: **The merge operation:** The component numbers of the merge operation u and s are obtained by the merge criterion described in Appendix C, where $u, s \in \{1, 2, \dots, C\}$ and $u \neq s$. The new component after merging the old u -th component and the old s -th component is called the u -th component.
 - 3: **The split operation:** The component number of the split operation t is obtained by the split criterion described in Appendix C, where $t \in \{1, 2, \dots, C\}$. New components after splitting the old t -th component are called the s -th and t -th components.
 - 4: For $n \in \{n | \mathbf{z}_n^{(3r-2)} = \mathbf{e}_s\}$, set $\mathbf{z}_n^{(3r-1)} = \mathbf{e}_u$; $\{n | \mathbf{z}_n^{(3r-2)} = \mathbf{e}_t; n = 1, 2, \dots, N\}$ is clustered into two clusters by the k-means clustering, and set $\mathbf{z}_n^{(3r-1)} = \mathbf{e}_s$ for samples of the first cluster; otherwise, set $\mathbf{z}_n^{(3r-1)} = \mathbf{z}_n^{(3r-2)}$.
 - 5: Obtain $\mathbf{Z}^{(3r)}$ by performing the partial CEM algorithm described in Algorithm 3.
 - 6: Obtain $\Phi^{(3r)}$ and $\mathbf{Z}^{(3r+1)}$ by performing the the CEM algorithm described in Algorithm 2.
 - 7: **Convergence criterion:** If the value of the approximated Q-function $L(\Phi^{(3r)}, \mathbf{Z}^{(3r+1)}) > L_{best}$, then set $L_{best} = L(\Phi^{(3r)}, \mathbf{Z}^{(3r+1)})$, $\Phi_{best} = \Phi^{(3r)}$, $r = r + 1$ and return to the second step; otherwise, stop.
-

Algorithm 2 The CEM algorithm for MWGP

Input: \mathcal{D}, C, J .

Output: $\Phi^{(r')}, \mathbf{Z}^{(r')}$.

- 1: **Initialization:** Set the initialized indicator variable matrix $\mathbf{Z}^{(0)} = \mathbf{Z}^{(3r)}$ (or $\mathbf{Z}^{(0)}$) and $r' = 1$.
- 2: **M step:** Update $\Phi^{(r')}$ by maximizing $L(\Phi, \mathbf{Z}^{(r'-1)})$ described in Appendix A.2.
- 3: **CE step:** Update $\mathbf{Z}^{(r')}$ by the approximated MAP principle described in Appendix A.1.
- 4: **Convergence criterion:** If $r' > 9$ and

$$\left(\sum_{i=r'-4}^{r'} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) - \sum_{i=r'-9}^{r'-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right) / \left| \sum_{i=r'-9}^{r'-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right| < \epsilon,$$

or $r' \geq r_{max}$, stop; otherwise, $r' = r' + 1$ and return to the second step.

Algorithm 3 The partial CEM algorithm for MWGP.

Input: \mathcal{D}, u, s, t, J .

Output: $\mathbf{Z}^{(r')}$.

- 1: **Initialization:** Set the initialized indicator variable matrix $\mathbf{Z}^{(0)} = \mathbf{Z}^{(3r-1)}$ and $r'' = 1$.
- 2: **Partial M step:** Update $\Phi_{c=u,s,t}^{(r')}$ by partially maximizing $L(\Phi, \mathbf{Z}^{(r''-1)})$, as described in Appendix B.1.
- 3: **Partial CE step:** Obtain $\mathbf{z}_n^{(r')}$ for $n \in \{n | \mathbf{z}_n^{(r''-1)} = \mathbf{e}_{c=u,s,t}\}$ by the approximated MAP principle described in Appendix B.2; otherwise, set $\mathbf{z}_n^{(r')} = \mathbf{z}_n^{(r''-1)}$.
- 4: **Convergence criterion:** If $r'' > 9$ and

$$\left(\sum_{i=r''-4}^{r''} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) - \sum_{i=r''-9}^{r''-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right) / \left| \sum_{i=r''-9}^{r''-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right| < \epsilon,$$

or $r'' \geq r_{max}$, stop; otherwise, $r'' = r'' + 1$ and return to the second step.

In the second and third steps of Algorithm 2, Φ and \mathbf{Z} are updated alternately. Samples are classified into C components to overcome the time complexity of the conventional EM

algorithm in the third step of Algorithm 2. In the fourth step of Algorithm 2, we apply a relatively long-term convergence criterion.

$$\left(\sum_{i=r'-4}^{r'} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) - \sum_{i=r'-9}^{r'-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right) / \left| \sum_{i=r'-9}^{r'-5} L(\Phi^{(i)}, \mathbf{Z}^{(i)}) \right| < \epsilon$$

since $L(\Phi^{(r')}, \mathbf{Z}^{(r')})$ may fluctuate during iterations, we present the largest number of iterations, i.e., $r_{max} = 30$ and $\epsilon = 0.002$. Regarding the annealing mechanism, Algorithm 2 can be viewed as a deterministic annealing EM algorithm with the annealing parameter tending to positive infinity, while the conventional EM algorithm can be viewed as a deterministic annealing EM algorithm with the annealing parameter being one. Theoretically, Algorithm 2 is more likely to fall into a local optimum than the conventional EM algorithm. The details of the CEM algorithm are described in Appendix A.

Algorithm 3 was performed on new components generated by the simultaneous split and merge operations. In the second and third steps of Algorithm 3, $\Phi_{c=u,s,t}^{(r'')}$ and $\mathbf{z}_n^{(r'')}$ are updated alternately, where $n \in \{n | \mathbf{z}_n^{(r''-1)} = \mathbf{e}_{c=u,s,t}\}$. In the third step of Algorithm 3, $\mathbf{z}_n^{(r'')}$ for $n \in \{n | \mathbf{z}_n^{(r''-1)} = \mathbf{e}_{c=u,s,t}\}$ is obtained by using the initialized $\mathbf{Z}^{(3r-1)}$, while $\mathbf{z}_n^{(r'')} = \mathbf{z}_n^{(r''-1)}$ is set for the other components. The details of the partial CEM algorithm are described in Appendix B.

4.2. Prediction Strategy

For MWGP, the time complexity of the conventional prediction method is generally high. We adopted the classification approximation method for MWGP to improve the predictive efficiency. In this prediction, the mean predictive output is used since the RMSE is measurable. In this paper, we used a predictive strategy similar to the MGP (or ME), i.e., the weighted prediction. A test sample was put into each WGP expert to calculate the predictive distribution individually, and then these predictive distributions were weighted and averaged according to the posterior probability to obtain the overall predictive distribution.

For the test sample \mathbf{x}_{N+1} in the c -th component, the predictive distribution in the latent space of the WGP is a standard GP

$$p(y_{N+1,c} | \mathbf{x}_{N+1,c}, \mathcal{D}, \boldsymbol{\theta}_c) = N(\tilde{y}_{N+1,c}, \sigma_{N+1,c}^2). \tag{7}$$

By a nonlinear transformation in Equation (7), the predictive distribution in the observation space is calculated by

$$\begin{aligned} & p(w_{N+1,c} | \mathbf{x}_{N+1,c}, \mathcal{D}, \boldsymbol{\theta}_c, \boldsymbol{\Omega}_c) \\ &= (\partial g(w_{N+1,c}; \boldsymbol{\Omega}_c) / \partial (w_{N+1,c})) / \sqrt{2\pi\sigma_{N+1,c}^2} \\ & \bullet \exp \left[- (g(w_{N+1,c}; \boldsymbol{\Omega}_c) - \tilde{y}_{N+1,c})^2 / 2\sigma_{N+1,c}^2 \right]. \end{aligned} \tag{8}$$

Compared to the shape of the predictive distribution in Equation (7), the shape of the predictive distribution in Equation (8) is generally asymmetric and multimodal. By integrating $w_{N+1,c}$ in Equation (8), the mean predictive output in the latent space of the WGP is obtained by

$$\begin{aligned} E(w_{N+1,c}) &= \int w_{N+1,c} p(w_{N+1,c} | \mathbf{x}_{N+1,c}, \mathcal{D}, \boldsymbol{\theta}_c, \boldsymbol{\Omega}_c) dw_{N+1,c} \\ &= \int g^{-1}(y_{N+1,c}; \boldsymbol{\Omega}_c) N(\tilde{y}_{N+1,c}, \sigma_{N+1,c}^2) dy_{N+1,c}, \end{aligned} \tag{9}$$

where $g^{-1}(\bullet; \boldsymbol{\Omega}_c)$ is the inverse of $g(\bullet; \boldsymbol{\Omega}_c)$. The closed-form solution of $g^{-1}(\bullet; \boldsymbol{\Omega}_c)$ is generally difficult to obtain, so we used the Newton–Raphson method to calculate it. Since

Equation (9) is the one-dimensional integral of $w_{N+1,c}$ in the Gaussian density function, it can be solved accurately by the Gauss–Hermite quadrature method.

Finally, the overall mean predictive output for MWGP is given based on Equation (9):

$$\hat{w}_{N+1} = \sum_{c=1}^C \hat{z}_{N+1,c} E(w_{N+1,c}),$$

where $\hat{\mathbf{z}}_{N+1} = \arg \max_{\mathbf{z}_{N+1}} P(\mathbf{z}_{N+1} | \Phi) p(\mathbf{x}_{N+1} | \mathbf{z}_{N+1}, \Phi)$.

5. Experimental Results

In this section, we show the experimental results of MWGP on synthetic datasets and three types of real-world datasets. The experiments were conducted on a personal computer equipped with a 2.9GHz Intel Core i7 CPU and 16.00 GB of RAM, using Matlab R2019b.

5.1. Comparative Models

Models and related algorithms are described in Table 1, where GP, support vector machine (SVM), and feedforward neural network (FNN) are comparative models. The GPML toolbox in Matlab R2019b, the SVM toolbox in Matlab R2019b, and the FNN toolbox in Matlab R2019b were adopted in our experiments. For MWGP II, MWGP I, and WGP, we chose an optimal value of J to avoid overfitting while balancing accuracy and efficiency. RMSE and MAE (note that the mean absolute error (MAE) is used to describe the sensitivity of the model to outliers, which is defined mathematically by $\sum_{n=1}^N |y_n - \hat{y}_n| / N$, where \hat{y}_n is the estimation of y_n) are used to assess the performance of real-world datasets.

Table 1. The symbols represent the models and related algorithms; the bold font is used for our proposed models.

Symbol	Model	Algorithm
MWGP II	MWGP	SMCEM
MWGP I		CEM
MGP [28]		CEM
WGP [39]	WGP	MLE
GP [49]	GP	
FNN [50]	FNN	Levenberg–Marquardt
SVM [51]	SVM	Sequential minimal optimization

5.2. Synthetic Datasets of MWGP I

To test the consistency of MWGP I, we generated 10 typical synthetic datasets by the MGP model with the component number $C = 2$ and the input dimension number $E = 1$, denoted by $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{10}$, respectively. \mathcal{S}_1 is the original dataset, where there are 300 training samples and 600 test samples. In each dataset, samples are warped by a monotonic function $g(\bullet; \Omega_c)$. The number of neurons J is set as 2, and Ω_c is randomly generated from a Gaussian distribution. The main parameters of MWGP I on \mathcal{S}_1 are shown in Table 2. The other datasets that differ from \mathcal{S}_1 are listed as follows.

- \mathcal{S}_2 (a low noise dataset): $\sigma_1 = \sigma_2 = 0.0200$.
- \mathcal{S}_3 (a high noise dataset): $\sigma_1 = \sigma_2 = 0.5000$.
- \mathcal{S}_4 : $\gamma_1^{(1)} = 0.0707, \gamma_2^{(1)} = 0.5000$.
- \mathcal{S}_5 : $\gamma_1^{(1)} = 0.2828, \gamma_2^{(1)} = 2.0000$.
- \mathcal{S}_6 (a short length-scale dataset): $\gamma_1^{(2)} = 0.8165, \gamma_2^{(2)} = 6.3246$.
- \mathcal{S}_7 (a long length-scale dataset): $\gamma_1^{(2)} = 0.2041, \gamma_2^{(2)} = 1.5811$.
- \mathcal{S}_8 (a medium overlapping dataset): $\Sigma_1^{1/2} = 2.1213, \Sigma_2^{1/2} = 3.1820$.
- \mathcal{S}_9 (a large overlapping dataset): $\Sigma_1^{1/2} = 3.0000, \Sigma_2^{1/2} = 4.5000$.
- \mathcal{S}_{10} (an unbalanced dataset): $\eta_1 = 0.2500, \eta_2 = 0.7500$.

The real parameters (RPs), average estimated parameters (AEPs), and standard deviations of estimated parameters (SDEPs) obtained by MWGP I on \mathcal{S}_1 are listed in Table 2, where the AEPs obtained by MWGP I are similar to the related RPs and the related SDEPs are generally small. As a result, the parameter estimate of MWGP I is practically unbiased and effective.

Table 2. RPs and AEPs with SDEPs were obtained through 150 trials using MWGP I on \mathcal{S}_1 .

		η_c	α_c	$\Sigma_c^{1/2}$	σ_c	$\gamma_c^{(1)}$	$\gamma_c^{(2)}$
$c = 1$	RP	0.5000	3.0000	1.8974	0.1414	0.1414	0.2887
	AEP	0.4944	3.1380	1.9185	0.1449	0.1584	0.2708
	SDEP	0.0059	0.0347	0.0588	0.0143	0.1389	0.2761
$c = 2$	RP	0.5000	10.500	2.8460	0.1414	1.0000	2.2361
	AEP	0.5056	10.688	2.9122	0.1477	1.2247	2.0492
	SDEP	0.0059	0.0380	0.0545	0.0138	0.1427	0.2658

The predictive results of MWGP I and MGP on \mathcal{S}_1 are presented in Figure 5a. The figure suggests that MWGP I outperforms MGP in the flat zone, specifically in the intervals (10.8, 11.7). In Figure 5b, the predictive probability density of MWGP I is asymmetrical across the whole distribution, but the predictive probability density of the MGP is symmetrical even when it is calculated by using the warped samples. Warping functions learned by MWGP I on \mathcal{S}_1 are shown in Figure 6. The warping function learned for the first component in Figure 6a is linear-like, while the warping function learned for the second component in Figure 6b is power-like, with an order between 0 and 1. It can be seen that MWGP I is flexible enough to handle non-stationary among different regions on a multimodal dataset.

In Table 3, the average predicted RMSEs, SDs of predicted RMSEs, p -values [52] of predicted RMSEs, and average running times for MWGP I and the other models on $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{10}$ are illustrated, where p -values are obtained by MWGP I and the other comparative models, respectively. The prediction accuracies of MWGP I and MGP are better than the other models due to the mixture structure, and the prediction accuracy of MWGP I is the best of all. MWGP I is superior to MGP in accuracy because of the warping function. Although the SDEPs of $\gamma_c^{(1)}$ and $\gamma_c^{(2)}$ are larger than the other parameters, the predicted results of MWGP I are accurate in \mathcal{S}_1 . Thus, MWGP I is robust for the estimates of $\gamma_c^{(1)}$ and $\gamma_c^{(2)}$. The SDs of predicted RMSEs for MWGP I and the other models are small. From the p -values, the predicted RMSE of MWGP I is different from the other models, except for MGP on \mathcal{S}_3 and \mathcal{S}_5 . Thus, our proposed MWGP I is effective, and it can optimize all parameters jointly.

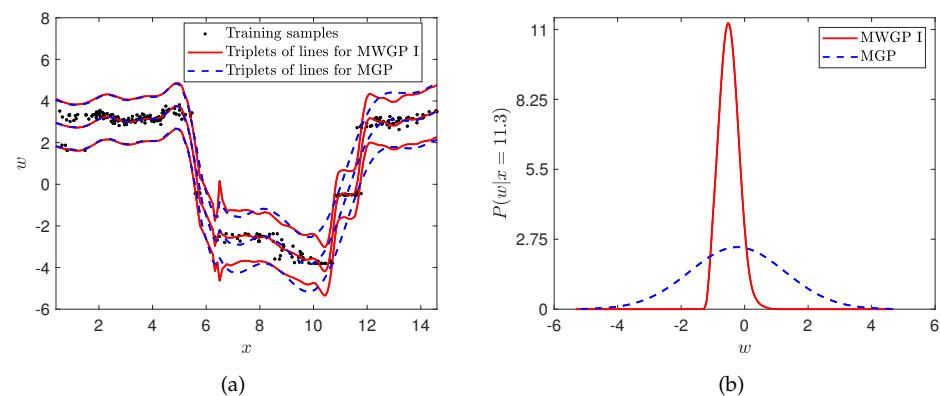


Figure 5. Fitting results of MWGP I and MGP on \mathcal{S}_1 : (a) Predictions of MWGP I and MGP. Line triplets represent the mean and two standard deviation (SD) bounds; (b) predictive probability densities of MWGP I and MGP at $x = 11.3$.

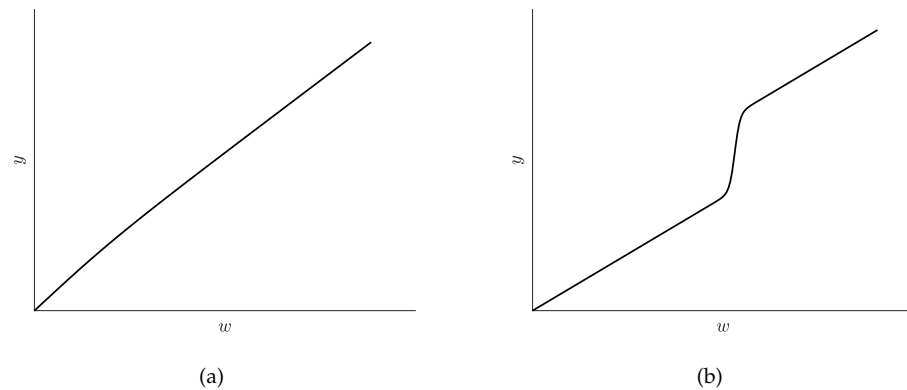


Figure 6. Warping functions obtained by MWGP I on \mathcal{S}_1 : (a) The warping function learned in the first component; (b) the warping function learned in the second component.

Table 3. The average predicted RMSEs, SDs of predicted RMSEs, p -values of predicted RMSEs, and average running times (seconds) for MWGP I and the other models from over one hundred trials on synthetic datasets; the bold font represents the best results.

Model	\mathcal{S}_1				\mathcal{S}_2				\mathcal{S}_3				\mathcal{S}_4				\mathcal{S}_5			
	RMSE			Time	RMSE			Time	RMSE			Time	RMSE			Time	RMSE			Time
	Average	SD	p -Value		Average	SD	p -Value		Average	SD	p -Value		Average	SD	p -Value		Average	SD	p -Value	
MWGP I	0.1024	0.0312	—	2.5916	0.0891	0.0641	—	2.6589	0.3751	0.0543	—	2.8216	0.0901	0.0394	—	2.5175	0.5077	0.0346	—	2.5261
MGP	0.1442	0.0217	0.0000	1.9620	0.1142	0.0638	0.0000	2.5429	0.3792	0.0473	0.1319	2.5187	0.1331	0.0418	0.0000	1.9213	0.5178	0.0415	0.0571	2.0896
WGP	0.2507	0.0242	0.0000	0.2250	0.1694	0.0505	0.0000	0.2019	0.4168	0.0096	0.0000	0.2133	0.1417	0.0437	0.0000	0.2041	0.5569	0.0310	0.0000	0.1940
GP	0.4083	0.0605	0.0000	0.1637	0.2867	0.0471	0.0000	0.1581	0.5434	0.0261	0.0000	0.1661	0.2146	0.0624	0.0000	0.1590	0.6676	0.0113	0.0000	0.1714
FNN	0.3715	0.1330	0.0000	1.9095	0.2695	0.0143	0.0000	1.6244	0.7014	0.1307	0.0000	1.7249	0.2928	0.0514	0.0000	2.4233	0.6749	0.1753	0.0000	2.1344
SVM	0.4605	0.1942	0.0000	45.236	0.3561	0.0139	0.0000	52.126	0.7901	0.1889	0.0000	39.451	0.3051	0.0539	0.0000	52.089	0.7295	0.2657	0.0000	58.141

Model	\mathcal{S}_6				\mathcal{S}_7				\mathcal{S}_8				\mathcal{S}_9				\mathcal{S}_{10}			
	RMSE			Time	RMSE			Time	RMSE			Time	RMSE			Time	RMSE			Time
	average	SD	p -value		average	SD	p -value		average	SD	p -value		average	SD	p -value		average	SD	p -value	
MWGP I	0.0807	0.0196	—	2.5135	0.2869	0.0318	—	2.5388	0.2582	0.0316	—	2.8099	0.4858	0.0296	—	2.7052	0.1816	0.0273	—	2.3483
MGP	0.1248	0.0197	0.0000	2.4189	0.3028	0.0528	0.0000	1.9746	0.2719	0.0372	0.0000	2.2128	0.5105	0.0343	0.0000	2.1638	0.2090	0.0337	0.0000	1.8251
WGP	0.1702	0.0527	0.0000	0.2609	0.3250	0.0492	0.0000	0.2269	0.3069	0.1058	0.0000	0.2342	0.5492	0.1279	0.0000	0.2958	0.2709	0.0581	0.0000	0.2382
GP	0.3456	0.0828	0.0000	0.1688	0.4787	0.0836	0.0000	0.1654	0.4352	0.1304	0.0000	0.1680	0.5596	0.1336	0.0000	0.2086	0.4416	0.0977	0.0000	0.1601
FNN	0.3230	0.0907	0.0000	2.0654	0.3720	0.1551	0.0000	1.8096	0.4527	0.1316	0.0000	2.2110	0.5496	0.1561	0.0000	2.1758	0.3768	0.1409	0.0000	1.6346
SVM	0.4915	0.1544	0.0000	51.590	0.5443	0.2024	0.0000	43.145	0.4958	0.1672	0.0000	55.062	0.5578	0.1736	0.0000	55.242	0.4675	0.1632	0.0000	48.347

5.3. Synthetic Datasets of MWGP II

For MWGP I, there is a local optimum in some cases. We propose MWGP II as a solution to this issue. To verify the consistency of MWGP II, we generated 6 typical synthetic datasets $\mathcal{S}_{11}, \mathcal{S}_{12}, \dots, \mathcal{S}_{16}$ by MGP with $C = 5$ and $E = 2$. In \mathcal{S}_{11} , there are 750 training samples and 1500 test samples. In each dataset, samples are warped by $g(\bullet; \Omega_c)$. We set $J = 3$ and generated Ω_c at random using a Gaussian distribution for these synthetic datasets. The main parameters of MWGP II on \mathcal{S}_{11} are shown in Table 4. The other datasets that differ from \mathcal{S}_{11} are listed as follows.

1. \mathcal{S}_{12} (a noise dataset): $\sigma_1 = \sigma_3 = \sigma_5 = 0.5000$, and $\sigma_2 = \sigma_4 = 0.1000$.
2. \mathcal{S}_{13} : $\gamma_1^{(1)} = 0.2828, \gamma_2^{(1)} = 2.0000, \gamma_3^{(1)} = 0.8944, \gamma_4^{(1)} = 0.7071$, and $\gamma_5^{(1)} = 0.5477$.
3. \mathcal{S}_{14} (a length-scale dataset): $\gamma_1^{(2)} = 0.2041, \gamma_2^{(2)} = 1.5811, \gamma_3^{(2)} = 1.2910, \gamma_4^{(2)} = 0.5000$, and $\gamma_5^{(2)} = 1.5811$.
4. \mathcal{S}_{15} (an overlapping dataset):

$$\Sigma_1^{1/2} = [3.0000, 2.4000; 2.4000, 3.0000],$$

$$\Sigma_2^{1/2} = [4.5000, -3.8730; -3.8730, 4.5000],$$

$$\Sigma_3^{1/2} = [3.0000, 0.0000; 0.0000, 3.0000],$$

$$\Sigma_4^{1/2} = [4.5000, -3.8730; -3.8730, 4.5000],$$

$$\Sigma_5^{1/2} = [3.0000, 2.4000; 2.4000, 3.0000].$$

5. \mathcal{S}_{16} (an unbalanced dataset): $\eta_1 = 1/9, \eta_2 = 3/9, \eta_3 = 1/9,$ and $\eta_4 = 3/9, \eta_5 = 1/9.$

Table 4. The main parameters of MWGP II on $\mathcal{S}_{11}.$

	η_c	$\alpha_c^{(1)}$	$\alpha_c^{(2)}$	$(\Sigma_c^{(11)})^{1/2}$	$(\Sigma_c^{(12)})^{1/2}$	$(\Sigma_c^{(21)})^{1/2}$	$(\Sigma_c^{(22)})^{1/2}$	σ_c	$\gamma_c^{(1)}$	$\gamma_c^{(2)}$	$\gamma_c^{(3)}$
$c = 1$	0.2000	3.0000	3.0000	1.8974	1.5000	1.5000	1.8974	0.1414	0.1414	0.2887	1.2910
$c = 2$	0.2000	10.500	10.500	2.8460	−2.1000	−2.1000	2.8460	0.0200	1.0000	2.2361	0.5000
$c = 3$	0.2000	18.000	18.000	1.8974	0.0000	0.0000	1.8974	0.1414	0.4472	1.8257	1.5811
$c = 4$	0.2000	25.500	25.500	2.8460	−2.1000	−2.1000	2.8460	0.0200	0.5000	0.7071	0.7071
$c = 5$	0.2000	33.000	33.000	1.8974	1.5000	1.5000	1.8974	0.1414	0.2739	2.2361	1.2910

The average ALLFs (approximated log-likelihood functions, i.e., values of the approximated Q-function after convergence of the SMCEM algorithm) and average running times of MWGP II and MWGP I on $\mathcal{S}_{11}, \mathcal{S}_{12}, \dots, \mathcal{S}_{16}$ are shown in Table 5. In these synthetic datasets, the average ALLF of MWGP II is larger than MWGP I, so MWGP II overcomes the local optimum of MWGP I. The average running time of MWGP II is longer than MWGP I since the partial CEM algorithm and the CEM algorithm are performed several times for the training of MWGP II. It can be concluded from the above discussion that our proposed MWGP II is effective.

Table 5. The average ALLFs and running times (seconds) of MWGP II and MWGP I from over one hundred trials on the synthetic datasets; the bold font represents the best results.

Model	\mathcal{S}_{11}		\mathcal{S}_{12}		\mathcal{S}_{13}	
	ALLF	Time	ALLF	Time	ALLF	Time
MWGP II	-1.0867×10^3	11.438	-1.6752×10^3	10.584	-1.7330×10^3	11.478
MWGP I	-1.1678×10^3	5.4443	-1.7388×10^3	3.9803	-1.8351×10^3	5.4065
Model	\mathcal{S}_{14}		\mathcal{S}_{15}		\mathcal{S}_{16}	
	ALLF	Time	ALLF	Time	ALLF	Time
MWGP II	-1.4240×10^3	10.315	-1.5961×10^3	11.527	-1.1597×10^3	12.130
MWGP I	-1.5726×10^3	3.9889	-1.6786×10^3	5.5424	-1.2283×10^3	5.7286

5.4. Toy and Motorcycle Datasets

Toy data [7,27] and motorcycle data [6,8,27] were used to test the performance of the MGP. We tested the consistency of our proposed MWGP II and MWGP I on the toy dataset \mathcal{S}_{17} and the motorcycle dataset $\mathcal{S}_{18}.$ \mathcal{S}_{17} consisted of four components generated by four continuous functions, i.e.,

$$\begin{aligned}
 y_1 &= 0.25x_1^2 - 40 + \sqrt{7}\epsilon, \\
 y_2 &= -0.0625(x_2 - 18)^2 + 0.5x_2 + 20 + \sqrt{7}\epsilon, \\
 y_3 &= 0.008(x_3 - 60)^2 - 70 + 2\epsilon, \\
 y_4 &= -\sin(x_4) - 6 + \sqrt{2}\epsilon,
 \end{aligned}$$

where $x_1 \in (0, 15), x_2 \in (35, 60), x_3 \in (45, 80), x_4 \in (80, 100)$ and $\epsilon \sim N(0, 1),$ as shown in Figure 7a. In each component, there are 50 training samples and 50 test samples. We set $J = 2$ and $C = 4$ for $\mathcal{S}_{17}.$

\mathcal{S}_{18} presents the accelerometer readings recorded at 133 moments during an experiment evaluating the effectiveness of crash helmets. In $\mathcal{S}_{18},$ samples belong to three components along the time axis (millisecond), i.e., (2.4, 11.4], (11.4, 40.4], and (40.4, 57.6], as shown in Figure 7b. We performed 7-fold cross-validation on this dataset, with the k -th fold consisting of the dataset $\{(x_n, y_n) : n = 7i + k, i = 0, 1, \dots, 18\}.$ We used 19 samples

as the test set and the remaining samples as the training set. For this dataset, we set $J = 2$ and $C = 3$ for \mathcal{S}_{18} .

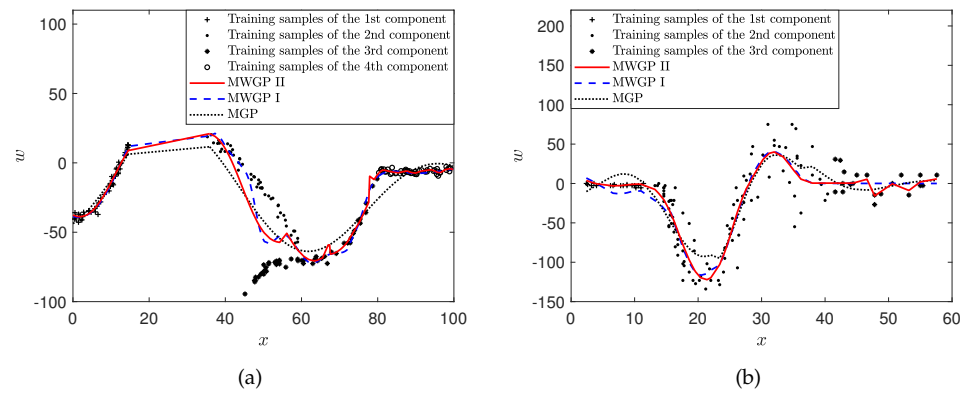


Figure 7. (a) Predictions of MWGP II, MWGP I, and MGP on \mathcal{S}_{17} ; (b) predictions of MWGP II, MWGP I, and MGP on \mathcal{S}_{18} .

We compare the MWGP II, MWGP I, MGP, WGP, GP, FNN, and SVM models on \mathcal{S}_{17} and \mathcal{S}_{18} , respectively. The average predicted RMSEs, average predicted MAEs, and average running times of these models are listed in Table 6. With these datasets, MWGP II and MWGP I are more accurate than the other models, and MWGP II is more accurate than MWGP I. The average predicted RMSE and average predicted MAE of the MGP are larger than those of a single GP and the WGP since the data in \mathcal{S}_{17} and \mathcal{S}_{18} are highly multimodal and non-stationary. The FNN and SVM can hardly fit \mathcal{S}_{17} and \mathcal{S}_{18} accurately. In Figure 7a, MWGP II and MWGP I are better than the MGP, for example on the interval (80, 100); in Figure 7b, MWGP II and MWGP I are better than the MGP, for example on the interval (2.4, 11.4]. Consequently, both MWGP II and MWGP I are effective for these tasks, and MWGP II can overcome the local optimum of MWGP I at the expense of only a little time on these datasets. In summary, the preprocessing transformation is critical for MGP in the toy and motorcycle datasets.

5.5. River-flow Datasets

We conducted experiments on ten river-flow datasets $\mathcal{S}_{19}, \mathcal{S}_{20}, \dots, \mathcal{S}_{28}$ to verify the consistency of our proposed MWGP II and MWGP I [53]. In each dataset, about 40 years (i.e., from 1920 to 1960) of monthly river flow for rivers in the USA (such as the Current River, the Mad River, the Madison River, and the Mackenzie River) were recorded. There are approximately 155 training samples and 313 test samples in each dataset. For river-flow datasets, there is minimal correlation between prediction accuracy and the value of C . We set $J = 2$ and $C = 4$ for these datasets.

For comparison, MWGP II, MWGP I, MGP, WGP, GP, FNN, and SVM are considered in $\mathcal{S}_{19}, \mathcal{S}_{20}, \dots, \mathcal{S}_{28}$, respectively. The average predicted RMSEs (cubic meters/second), average predicted MAEs, and average running times of these models are recorded in Table 6. From this table, MWGP II and MWGP I are smaller than the other models in terms of the average predicted RMSE and average predicted MAE, and the average predicted RMSE and average predicted MAE of MWGP II are smaller than those of MWGP I. Although MWGP I has the same accuracy as MWGP II in $\mathcal{S}_{25}, \mathcal{S}_{26}, \dots, \mathcal{S}_{28}$, it is more efficient than MWGP II. Based on the analysis above, our proposed MWGP II and MWGP I are effective for processing the river-flow datasets, and demonstrate that the nonlinear transformation is useful for this type of data. Additionally, MWGP II can overcome the local optimum of MWGP I at a minimal computational cost on some datasets.

Table 6. The average predicted RMSEs, average predicted MAEs, and average running times (seconds) of different models from over thirty trials on the toy dataset, the motorcycle dataset, and the river-flow datasets; the bold font represents the best results.

Model	S_{17}			S_{18}			S_{19}			S_{20}		
	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time
MWGP II	13.481	7.7561	6.9812	24.153	13.297	15.671	47.896	29.157	2.3341	10.425	5.5228	2.2285
MWGP I	14.312	8.1550	2.3733	25.987	14.309	6.8800	48.171	29.316	1.0935	10.668	5.7159	1.0824
MGP	14.714	8.4912	1.6331	26.370	14.351	4.4129	49.060	30.075	0.7358	11.071	6.0772	0.6139
WGP	14.772	8.4834	0.1231	29.277	16.579	0.4211	49.411	30.391	0.0824	11.654	6.5137	0.0806
GP	20.387	13.322	0.1070	26.700	14.725	0.3186	55.466	35.323	0.0703	14.174	8.7933	0.0645
FNN	18.004	11.974	1.6293	30.359	17.213	16.433	49.588	30.514	1.2033	11.669	6.5154	1.1592
SVM	17.267	11.445	46.223	29.782	16.885	182.67	54.627	34.917	25.333	12.780	7.4816	23.858
Model	S_{21}			S_{22}			S_{23}			S_{24}		
	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time
MWGP II	4.5938	3.7556	2.2943	14.266	8.6631	2.4118	16.617	10.885	2.2389	31.171	22.483	2.2581
MWGP I	4.6721	3.8321	1.0644	14.570	8.9534	1.1831	16.727	10.988	1.1016	31.536	22.814	1.1268
MGP	5.1759	4.3327	0.5734	14.924	9.2859	0.5911	16.814	11.067	0.7345	34.575	26.257	0.7646
WGP	4.7084	3.8626	0.0673	15.318	9.6578	0.0939	16.728	10.990	0.0886	32.428	23.877	0.0819
GP	5.6274	4.6852	0.0587	16.161	10.527	0.0718	17.043	11.302	0.0711	34.813	26.416	0.0755
FNN	4.7079	3.8629	1.1525	15.599	9.9261	1.3947	16.738	10.996	1.1650	32.666	24.093	1.1725
SVM	4.8446	3.9841	19.240	16.353	10.673	26.039	17.415	11.579	23.659	33.163	24.552	23.745
Model	S_{25}			S_{26}			S_{27}			S_{28}		
	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time	RMSE	MAE	Time
MWGP II	27.736	19.675	2.4640	30.696	22.095	2.4040	50.497	31.265	2.2823	33.789	24.613	2.2461
MWGP I	27.776	19.702	1.2057	30.708	22.121	1.1113	50.502	31.283	1.1090	33.792	24.624	1.0795
MGP	28.053	20.015	0.7480	32.061	23.216	0.7923	52.317	32.837	0.7065	34.529	25.277	0.7084
WGP	27.785	19.719	0.0991	30.712	22.128	0.0909	50.712	31.415	0.0857	34.004	24.858	0.0826
GP	28.310	20.263	0.0762	32.803	23.871	0.0756	52.994	33.478	0.0745	35.276	26.064	0.0738
FNN	27.921	19.998	1.1907	30.727	22.141	1.1980	50.603	31.357	1.1813	34.163	25.040	1.1099
SVM	28.098	20.068	24.062	31.812	22.844	27.023	52.920	33.432	26.387	35.072	25.821	20.228

6. Conclusions and Discussion

In this paper, we demonstrate that the MWGP model is a valuable generalization of the MGP and WGP models, and it is well suited for solving non-stationary probabilistic regression. From another point of view, the standard preprocessing transformation in MWGP can be learned adaptively and improved upon. We show that simultaneous split and merge operations are able to eliminate the component differences between the two regions to avoid the local optimum of the CEM algorithm for MWGP. Experimental results on synthetic and real-world datasets show that our proposed MWGP trained by the CEM algorithm as well as MWGP trained by the SMCEM algorithm are effective.

For MWGP, the actual number of WGP components is generally difficult to learn due to the correlation among outputs. In future work, we will focus on learning C for MWGP. Moreover, for probabilistic regression models, there are likely outliers in the observations that deviate significantly from the other samples. We will consider the robustness for MWGP based on the robustness of the MGP.

Author Contributions: Conceptualization, Y.X. and D.W.; methodology, Y.X., D.W., and Z.Q.; data simulation and experiment, Y.X. and D.W.; writing—original draft, Y. X.; writing—review and editing, D.W.; supervision, D.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (62006149), the Natural Science Foundation of Shaanxi Province (2020JQ-403), and the Foundation of Shaanxi Educational Committee (18JK0792).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The motorcycle data and the river-flow data that support the findings of this study are available at <https://doi.org/10.1111/j.2517-6161.1985.tb01327.x>; <https://doi.org/10.2307/1403750> (accessed on 9 June 2022). All other datasets are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare that they have no known competing financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Nomenclature

AEP	Average estimated parameter
ALLF	Approximated log-likelihood function
CEM	Classification expectation–maximization (also called hard-cut expectation–maximization or hard expectation–maximization)
EM	Expectation–maximization
FNN	Feedforward neural network
GP	Gaussian process
LOOCV	Leave-one-out cross-validation
MAE	Mean absolute error
MAP	Maximum a posteriori
MCMC	Markov chain Monte Carlo
ME	Mixture of experts
MGP	Mixture of Gaussian processes
MLE	Maximum likelihood estimation
MWGP	Mixture of warped Gaussian processes
RMSE	Root mean square error
RP	Real parameter
SDEP	Standard deviation of the estimated parameter
SD	Standard deviation
SMCEM	Split and merge classification expectation–maximization
SVM	Support vector machine
VB	Variational Bayesian
WGP	Warped Gaussian process

Appendix A. Details of the CEM Algorithm

Appendix A.1. The Derivation of the Q-Function and Details of the Approximated MAP Principle

Denote $g(\mathbf{w}_c; \mathbf{\Omega}_c) = [g(w_n; \mathbf{\Omega}_c) | \mathbf{z}_n = \mathbf{e}_c; n = 1, 2, \dots, N]$ as the $N_c \times 1$ function vector of the c -th component, $\mathbf{K}_c = [K(\mathbf{x}_n, \mathbf{x}_{\tilde{n}}; \gamma_c) | \mathbf{z}_n = \mathbf{z}_{\tilde{n}} = \mathbf{e}_c; n, \tilde{n} = 1, 2, \dots, N]$ as the $N_c \times N_c$ covariance matrix of the c -th component, and $\sum_{n=1}^{N_c} \ln \partial g(w_n; \mathbf{\Omega}_c) / \partial w_n$ as a Jacobian term. The total log-likelihood function of MWGP is given by

$$\begin{aligned}
 L(\Phi, \mathbf{Z}) &= \ln p(\mathcal{D}, \mathbf{Z} | \Phi) = \sum_{c=1}^C L_c(\Phi_c, \mathbf{Z}) \\
 &= \sum_{c=1}^C \left\{ \sum_{n=1}^N z_{nc} [\ln \eta_c + \ln p(\mathbf{x}_n | \mathbf{z}_n = \mathbf{e}_c)] + \ln p(\mathbf{w}_c | \mathbf{X}_c, \theta_c, \mathbf{\Omega}_c) \right\},
 \end{aligned}
 \tag{A1}$$

where $\ln p(\mathbf{w}_c | \mathbf{X}_c, \theta_c, \mathbf{\Omega}_c)$ is the log-likelihood function of the c -th WGP given by

$$\begin{aligned}
 \ln p(\mathbf{w}_c | \mathbf{X}_c, \theta_c, \mathbf{\Omega}_c) &= - \left[N_c \ln 2\pi + g(\mathbf{w}_c; \mathbf{\Omega}_c)^T (\mathbf{K}_c + \sigma_c^2 \mathbf{I}_{N_c})^{-1} g(\mathbf{w}_c; \mathbf{\Omega}_c) \right. \\
 &\quad \left. + \ln |\mathbf{K}_c + \sigma_c^2 \mathbf{I}_{N_c}| \right] / 2 + \sum_{n=1}^{N_c} \ln \partial g(w_n; \mathbf{\Omega}_c) / \partial w_n.
 \end{aligned}$$

The Q-function of the conventional EM algorithm of MWGP is obtained by the expectation of Equation (A1) with respect to \mathbf{Z} :

$$Q(\Phi | \Phi^{(r-1)}) = \mathbb{E}_{\mathbf{Z}} [L(\Phi, \mathbf{Z}) | \mathcal{D}, \Phi^{(r-1)}] = \sum_{\mathbf{Z}} P(\mathbf{Z} | \mathcal{D}, \Phi^{(r-1)}) L(\Phi, \mathbf{Z}).
 \tag{A2}$$

The posterior probability $P(\mathbf{z}_n|\mathbf{x}_n, w_n, \Phi^{(r)})$ is calculated by

$$\begin{aligned}
 P(\mathbf{z}_n|\mathbf{x}_n, w_n, \Phi^{(r)}) &\propto p(\mathbf{x}_n, w_n, \mathbf{z}_n|\Phi^{(r)}) \\
 &= P(\mathbf{z}_n|\Phi^{(r)})p(\mathbf{x}_n|\mathbf{z}_n, \Phi^{(r)})p(w_n|\mathbf{x}_n, \mathbf{z}_n, \Phi^{(r)}) \\
 &= P(\mathbf{z}_n|\Phi^{(r)})p(\mathbf{x}_n|\mathbf{z}_n, \Phi^{(r)})p(y_n|\mathbf{x}_n, \mathbf{z}_n, \Phi^{(r)}) \\
 &= P(\mathbf{z}_n|\Phi^{(r)})p(\mathbf{x}_n|\mathbf{z}_n, \Phi^{(r)}) \\
 &\quad \bullet p(g(w_n; \Omega^{(r)})|\mathbf{x}_n, \mathbf{z}_n, \Phi^{(r)})\partial g(w_n; \Omega^{(r)})/\partial w_n,
 \end{aligned}
 \tag{A3}$$

where $\partial g(w_n; \Omega^{(r)})/\partial w_n$ is the Jacobian term. Since the number of \mathbf{Z} is C^N , the time complexity of calculating Equation (A2) is $O(C^N)$. Thus, the classification approximation is adopted for calculating Equation (A2), and then we obtain the approximated Q-function for the CEM algorithm:

$$L(\Phi, \mathbf{Z}^{(r)}) = \sum_{c=1}^C L_c(\Phi_c, \mathbf{Z}^{(r)}), \tag{A4}$$

where $\mathbf{Z}^{(r)}$ is calculated by an approximation of the MAP method, i.e., $\mathbf{Z}^{(r)} = \arg \max_{\mathbf{Z}} P(\mathbf{Z}|\mathcal{D}, \Phi^{(r)})$:

$$\begin{aligned}
 \mathbf{z}_n^{(r)} &= \arg \max_{\mathbf{z}_n} P(\mathbf{z}_n|\mathbf{x}_n, w_n, \Phi^{(r)}) \\
 &= \arg \max_{\mathbf{z}_n} P(\mathbf{z}_n|\Phi^{(r)})p(\mathbf{x}_n|\mathbf{z}_n, \Phi^{(r)}) \\
 &\quad \bullet p(g(w_n; \Omega^{(r)})|\mathbf{x}_n, \mathbf{z}_n, \Phi^{(r)})\partial g(w_n; \Omega^{(r)})/\partial w_n.
 \end{aligned}
 \tag{A5}$$

In Equation (A5), $P(\mathbf{z}_n|\mathbf{x}_n, w_n, \Phi^{(r)})$ is derived by Equation (A3).

Appendix A.2. Details for Maximizing the Approximated Q-Function

Parameters θ_c and Ω_c are updated jointly by the conjugated gradient method inherited by training the WGP.

Parameters η_c , α_c , and Σ_c are solved analytically as follows. By adopting the Lagrange multiplier method when $\sum_{c=1}^C \eta_c = 1$, we have

$$\eta_c = N_c / \sum_{c=1}^C N_c.$$

Let $\partial L(\Phi, \mathbf{Z}^{(r-1)})/\partial \alpha_c = 0$ and $\partial L(\Phi, \mathbf{Z}^{(r-1)})/\partial \Sigma_c = 0$. Then, we have

$$\begin{aligned}
 \alpha_c &= \sum_{n=1}^N z_{nc}\mathbf{x}_n / N_c, \\
 \Sigma_c &= \sum_{n=1}^N z_{nc}(\mathbf{x}_n - \alpha_c)(\mathbf{x}_n - \alpha_c)^T / N_c.
 \end{aligned}$$

Appendix B. Details of the Partial CEM Algorithm

Appendix B.1. Details of Maximizing the Approximated Q-Function of the Partial CEM Algorithm

The approximated Q-function Equation (A4) is equal to

$$L(\Phi, \mathbf{Z}^{(r)}) = \sum_{c=u,s,t} L_c(\Phi_c, \mathbf{Z}^{(r)}) + \sum_{c \neq u,s,t} L_c(\Phi_c, \mathbf{Z}^{(r)}),$$

where the first three terms, i.e., $\sum_{c=u,s,t} L_c(\Phi_c, \mathbf{Z}^{(r)})$, are only maximized for the partial CEM algorithm. The details for maximizing $\sum_{c=u,s,t} L_c(\Phi_c, \mathbf{Z}^{(r)})$ are described in Appendix A.2.

Appendix B.2. Details of the Approximated MAP Principle of the Partial CEM Algorithm

When $\mathbf{z}_n^{(r-1)} = \mathbf{e}_{c=u,s,t}$, $\mathbf{z}_n^{(r)}$ is obtained by the approximated MAP method:

$$\begin{aligned} \mathbf{z}_n^{(r)} &= \arg \max_{\mathbf{z}_n \in \{\mathbf{e}_{c=u,s,t}\}} P(\mathbf{z}_n | \mathbf{x}_n, w_n, \Phi_c^{(r)}) \\ &= \arg \max_{\mathbf{z}_n \in \{\mathbf{e}_{c=u,s,t}\}} P(\mathbf{z}_n = \mathbf{e}_c | \Phi_c^{(r)}) p(\mathbf{x}_n | \mathbf{z}_n = \mathbf{e}_c, \Phi_c^{(r)}) \\ &\quad \bullet p(g(w_n; \Omega_c^{(r)}) | \mathbf{x}_n, \mathbf{z}_n = \mathbf{e}_c, \Phi_c^{(r)}) \partial g(w_n; \Omega_c^{(r)}) / \partial w_n, \end{aligned}$$

where $P(\mathbf{z}_n | \mathbf{x}_n, w_n, \Phi_c^{(r)})$ is derived by Equation (A3).

Appendix C. Split and Merge Criteria

Since there are too many candidate sets, it is necessary to propose specific reasonable criteria to speed up the SMCEM algorithm.

The **merge criterion** is defined by

$$F_{merge}(u, s) = \cos \langle \mathbf{p}_u, \mathbf{p}_s \rangle = \mathbf{p}_u^T \mathbf{p}_s / \|\mathbf{p}_u\| \|\mathbf{p}_s\|,$$

where $\|\cdot\|$ is the Euclidean norm, and \mathbf{p}_c represents the following $N \times 1$ vectors $[P(\mathbf{z}_1 = \mathbf{e}_c | \mathbf{x}_1, w_1, \Phi_c^{(r)}), P(\mathbf{z}_2 = \mathbf{e}_c | \mathbf{x}_2, w_2, \Phi_c^{(r)}), \dots, P(\mathbf{z}_N = \mathbf{e}_c | \mathbf{x}_N, w_N, \Phi_c^{(r)})]^T$, in which $P(\mathbf{z}_n = \mathbf{e}_c | \mathbf{x}_n, w_n, \Phi_c^{(r)})$ are derived by Equation (A3). Components with the largest $F_{merge}(u, s)$ are used for merging, where $u \neq s$.

The **split criterion** is defined by

$$F_{split}(t) = L_t(\Phi_t, \mathbf{Z}^{(r)}) / N_t.$$

The component with the smallest $F_{split}(t)$ is used for splitting.

References

1. Yuksel, S.E.; Wilson, J.N.; Gader, P.D. Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *23*, 1177–1193. [CrossRef]
2. Jordan, M.I.; Jacobs, R.A. Hierarchies mixtures of experts and the EM algorithm. *Neural Comput.* **1994**, *6*, 181–214. [CrossRef]
3. Lima, C.A.M.; Coelho, A.L.V.; Zuben, F.J.V. Hybridizing mixtures of experts with support vector machines: Investigation into nonlinear dynamic systems identification. *Inf. Sci.* **2007**, *177*, 2049–2074. [CrossRef]
4. Tresp, V. Mixtures of Gaussian processes. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Denver, CO, USA, 1 January 2000; Volume 13, pp. 654–660.
5. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B Methodol.* **1977**, *39*, 1–38.
6. Rasmussen, C.E.; Ghahramani, Z. Infinite mixture of Gaussian process experts. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 9–14 December 2002 ; Volume 2, pp. 881–888.
7. Meeds, E.; Osindero, S. An alternative infinite mixture of Gaussian process experts. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 4–7 December 2005; Volume 18, pp. 883–896.
8. Yuan, C.; Neubauer, C. Variational mixture of Gaussian process experts. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Vancouver, BC, Canada, 8–11 December 2008; Volume 21, pp. 1897–1904.
9. Brahim-Belhouari, S.; Bermak, A. Gaussian process for nonstationary time series prediction. *Comput. Stat. Data Anal.* **2004**, *47*, 705–712. [CrossRef]
10. Pérez-Cruz, F.; Vaerenbergh, S.V.; Murillo-Fuentes, J.J.; Lázaro-Gredilla, M.; Santamaría, I. Gaussian processes for nonlinear signal processing: An overview of recent advances. *IEEE Signal Process. Mag.* **2013**, *30*, 40–50. [CrossRef]
11. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Process for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006; Chapter 2.
12. MacKay, D.J.C. Introduction to Gaussian processes. *NATO ASI Ser. F Comput. Syst. Sci.* **1998**, *168*, 133–166.
13. Xu, Z.; Guo, Y.; Saleh, J.H. VisPro: A prognostic SqueezeNet and non-stationary Gaussian process approach for remaining useful life prediction with uncertainty quantification. *Neural Comput. Appl.* **2022**, *34*, 14683–14698. [CrossRef]
14. Heinonen, M.; Mannerström, H.; Rousu, J.; Kaski, S.; Lähdesmäki, H. Non-stationary Gaussian process regression with hamiltonian monte carlo. In Proceedings of the Machine Learning Research, Cadiz, Spain, 9–11 May 2016; Volume 51, pp. 732–740.
15. Wang, Y.; Chaib-draa, B. Bayesian inference for time-varying applications: Particle-based Gaussian process approaches. *Neuro-computing* **2017**, *238*, 351–364. [CrossRef]

16. Rhode, S. Non-stationary Gaussian process regression applied in validation of vehicle dynamics models. *Eng. Appl. Artif. Intell.* **2020**, *93*, 103716. [[CrossRef](#)]
17. Sun, S.; Xu, X. Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction. *IEEE Trans. Intell. Transp. Syst.* **2010**, *12*, 466–475. [[CrossRef](#)]
18. Jeon, Y.; Hwang, G. Bayesian mixture of gaussian processes for data association problem. *Pattern Recognit.* **2022**, *127*, 108592. [[CrossRef](#)]
19. Li, T.; Ma, J. Attention mechanism based mixture of Gaussian processes. *Pattern Recognit. Lett.* **2022**, *161*, 130–136. [[CrossRef](#)]
20. Kim, S.; Kim, J. Efficient clustering for continuous occupancy mapping using a mixture of Gaussian processes. *Sensors* **2022**, *22*, 6832. [[CrossRef](#)]
21. Tayal, A.; Poupart, P.; Li, Y. Hierarchical double Dirichlet process mixture of Gaussian processes. In Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI), Toronto, ON, Canada, 22–26 July 2012; pp. 1126–1133.
22. Sun, S. Infinite mixtures of multivariate Gaussian processes. In Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC), Tianjin, China, 14–17 July 2013; pp. 1011–1016.
23. Kastner, M. Monte Carlo methods in statistical physics: Mathematical foundations and strategies. *Commun. Nonlinear Sci. Numer. Simul.* **2010**, *15*, 1589–1602. [[CrossRef](#)]
24. Khodadadian, A.; Parvizi, M.; Teshnehlab, M.; Heitzinger, C. Rational design of field-effect sensors using partial differential equations, Bayesian inversion, and artificial neural networks. *Sensors* **2022**, *22*, 4785. [[CrossRef](#)] [[PubMed](#)]
25. Noii, N.; Khodadadian, A.; Ulloa, J.; Aldakheel, F.; Wick, T.; François, S.; Wriggers, P. Bayesian inversion with open-source codes for various one-dimensional model problems in computational mechanics. *Arch. Comput. Methods Eng.* **2022**, *29*, 4285–4318. [[CrossRef](#)]
26. Ross, J.C.; Dy, J.G. Nonparametric mixture of Gaussian processes with constraints. In Proceedings of the 30th International Conference on Machine Learning (ICML), Atlanta, GA, USA, 17–19 June 2013; pp. 1346–1354.
27. Yang, Y.; Ma, J. An efficient EM approach to parameter learning of the mixture of Gaussian processes. In Proceedings of the Advances in International Symposium on Neural Networks (ISNN), Guilin, China, 29 May–1 June 2011; Volume 6676, pp. 165–174.
28. Chen, Z.; Ma, J.; Zhou, Y. A precise hard-cut EM algorithm for mixtures of Gaussian processes. In Proceedings of the 10th International Conference on Intelligent Computing (ICIC), Taiyuan, China, 3–6 August 2014; Volume 8589, pp. 68–75.
29. Celeux, G.; Govaert, G. A classification EM algorithm for clustering and two stochastic versions. *Comput. Stat. Data Anal.* **1992**, *14*, 315–332. [[CrossRef](#)]
30. Wu, D.; Chen, Z.; Ma, J. An MCMC based EM algorithm for mixtures of Gaussian processes. In Proceedings of the Advances in International Symposium on Neural Networks (ISNN), Jeju, Republic of Korea, 15–18 October 2015; Volume 9377, pp. 327–334.
31. Wu, D.; Ma, J. An effective EM algorithm for mixtures of Gaussian processes via the MCMC sampling and approximation. *Neurocomputing* **2019**, *331*, 366–374. [[CrossRef](#)]
32. Ma, J.; Xu, L.; Jordan, M.I. Asymptotic convergence rate of the EM algorithm for Gaussian mixtures. *Neural Comput.* **2000**, *12*, 2881–2907. [[CrossRef](#)]
33. Zhao, L.; Chen, Z.; Ma, J. An effective model selection criterion for mixtures of Gaussian processes. In Proceedings of the Advances in Neural Networks-ISNN, Jeju, Republic of Korea, 15–18 October 2015; Volume 9377, pp. 345–354.
34. Ueda, N.; Nakano, R.; Ghahramani, Z.; Hinton, G.E. SMEM algorithm for mixture models. *Adv. Neural Inf. Process. Syst.* **1998**, *11*, 599–605. [[CrossRef](#)] [[PubMed](#)]
35. Li, Y.; Li, L. A novel split and merge EM algorithm for Gaussian mixture model. In Proceedings of the International Conference on Natural Computation (ICNC), Tianjin, China, 14–16 August 2009; pp. 479–483.
36. Zhang, Z.; Chen, C.; Sun, J.; Chan, K.L. EM algorithms for Gaussian mixtures with split-and-merge operation. *Pattern Recognit.* **2003**, *36*, 1973–1983. [[CrossRef](#)]
37. Zhao, L.; Ma, J. A dynamic model selection algorithm for mixtures of Gaussian processes. In Proceedings of the IEEE 13th International Conference on Signal Processing (ICSP), Chengdu, China, 6–10 November 2016; pp. 1095–1099.
38. Li, T.; Wu, D.; Ma, J. Mixture of robust Gaussian processes and its hard-cut EM algorithm with variational bounding approximation. *Neurocomputing* **2021**, *452*, 224–238. [[CrossRef](#)]
39. Snelson, E.; Rasmussen, C.E.; Ghahramani, Z. Warped Gaussian processes. *Adv. Neural Inf. Process. Syst.* **2003**, *16*, 337–344.
40. Schmidt, M.N. Function factorization using warped Gaussian processes. In Proceedings of the 26th International Conference on Machine Learning (ICML), Montreal, QC, Canada, 14–18 June 2009; pp. 921–928.
41. Lázaro-Gredilla, M. Bayesian warped Gaussian processes. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 6995–7004.
42. Rios, G.; Tobar, F. Compositionally-warped Gaussian processes. *Neural Netw.* **2019**, *118*, 235–246. [[CrossRef](#)]
43. Zhang, Y.; Yeung, D.Y. Multi-task warped Gaussian process for personalized age estimation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA, 13–18 June 2010; pp. 2622–2629.
44. Wiebe, J.; Cecílio, I.; Dunlop, J.; Misener, R. A robust approach to warped Gaussian process-constrained optimization. *Math. Program.* **2022**, *196*, 805–839. [[CrossRef](#)]
45. Mateo-Sanchis, A.; Muñoz-Marí, J.; Pérez-Suay, A.; Camps-Valls, G. Warped Gaussian processes in remote sensing parameter estimation and causal inference. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 1647–1651. [[CrossRef](#)]

46. Jadidi, M.G.; Miró, J.V.; Dissanayake, G. Warped Gaussian processes occupancy mapping with uncertain inputs. *IEEE Robot. Autom. Lett.* **2017**, *2*, 680–687. [[CrossRef](#)]
47. Kou, P.; Liang, D.; Gao, F.; Gao, L. Probabilistic wind power forecasting with online model selection and warped Gaussian process. *Energy Convers. Manag.* **2014**, *84*, 649–663. [[CrossRef](#)]
48. Gonçalves, I.G.; Echer, E.; Frigo, E. Sunspot cycle prediction using warped Gaussian process regression. *Adv. Space Res.* **2020**, *65*, 677–683. [[CrossRef](#)]
49. Rasmussen, C.E.; Nickisch, H. Gaussian processes for machine learning (GPML) toolbox. *J. Mach. Learn. Res.* **2010**, *11*, 3011–3015.
50. Svozil, D.; Kvasnicka, V.; Pospichal, J. Introduction to multi-layer feedforward neural networks. *Chemom. Intell. Lab. Syst.* **1997**, *39*, 43–62. [[CrossRef](#)]
51. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27. [[CrossRef](#)]
52. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [[CrossRef](#)]
53. Mcleod, A.I. Parsimony, model adequacy and periodic correlation in forecasting time series. *Int. Stat. Rev.* **1993**, *61*, 387–393. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.