True Gaussian mixture regression and genetic algorithm-based optimization with constraints for direct inverse analysis

Hiromasa Kaneko
Department of Applied Chemistry, School of Science and Technology, Meiji University, Kanagawa, Japan

ABSTRACT
In molecular, material, and process design, Gaussian mixture regression (GMR) can directly predict values of explanatory variables x such as experimental and process conditions from target values of objective variables y such as properties and activities of materials. In GMR, a prediction result of a Gaussian mixture distributions is provided as a probability density distribution. The weighted average of the means of Gaussians has been used as the representative value of Gaussian mixture. In this study, true GMR has been proposed, where the predicted value of the GMR model is calculated by solving a continuous optimization problem to obtain the maximum value of the probability density function (PDF). As the continuous optimization problem cannot be solved when variables contain discrete values, a genetic algorithm is used to search for a solution that satisfies the constraints, while maximizing the value of the PDF. Numerical simulations using nonlinear functions between y and x, and the case studies using real material datasets confirmed that the proposed methods produced solutions with drastically higher values of the PDF than the traditional methods, especially when x is constrained in the inverse analysis. The Python codes of the proposed methods are freely available. Direct inverse analysis with GMR using the proposed methods is expected to improve the efficiency of molecular, material, and process design.

KEYWORDS
Gaussian mixture regression; genetic algorithm; constraint condition; material design

1. Introduction
The designs of molecules, materials, and processes using machine learning have become common in recent years. Desirable molecules, materials, and processes are designed based on model y = f(x), which is constructed between explanatory variables x and objective variables y using molecular, material, and process data, respectively. Prediction of y values from x values is known as forward analysis of the model, and conversely, prediction of x values from y values is known as inverse analysis of the model. Molecular, material, and process design correspond to inverse analysis.

In general inverse analysis, a large number of samples of x are generated, while y values are predicted by inputting them into the model, and further, an appropriate sample is selected based on the predicted y values. More precisely, it is pseudo-inverse analysis, in which forward analysis is conducted multiple times. Bayesian optimization [1,2] in adaptive design of experiments is also a pseudo-inverse analysis, except that predicted y values are changed to values of an acquisition function. Optimization algorithms, such as a genetic algorithm (GA) [3] can be used to efficiently search for x values that result in a desired prediction of a y value and a maximum value of acquisition function.

However, using generative models constructed using Gaussian mixture regression (GMR) [4] and generative topographic mapping regression (GTMR) [5] allow us to directly predict x values from y values, which would be a direct inverse analysis, and not a pseudo-inverse analysis. In models constructed using GMR and GTMR, all relationships between x and y, i.e. joint...
probability distribution, are represented as a mixture of Gaussian distributions. Based on the joint probability distribution, the posterior probability density distribution of x can be calculated, which is also represented as a mixture of Gaussians, for given y values using multiplication theorem on probability and Bayes’ theorem. The direct inverse analysis using GMR exhibited more efficiency in adaptive design of experiments, than Bayesian optimization [4]. GMR and GTMR are equipped to handle multiple y variables [6], and GTMR can visualize a dataset simultaneously, along with forward and inverse analyses of the model.

In GMR and GTMR, x and y are handled equally as variables, and thus, prediction of x values from y values, and prediction of y values from x values are performed in the same manner. Yamada et al. investigated signal deconvolution methods using a short-time Fourier transform and a non-negative tensor/matrix factorization as well as methods for predicting NMR signals and physical properties using GTMR [7]. Wang et al. predicted continuous tool wear process using GMR [8]. Yao and Ge used GMR for oxygen content prediction of a nonlinear multimode primary reformer [9]. Ye et al. proposed a closed-loop framework of robot learning through demonstration based on the bagging method of GMR to obtain a robust learner of robot learning, with a high-precision reproduction [10]. Zheng et al. used GMR for modeling and optimizing the cement calcination process for reducing NOx emissions [11]. Kaneko proposed extended GMR [12], and latent variables-based GMR [13] to improve the predictive ability of forward and inverse analyses of GMR models.

While predicting y values from x values and x values from y values, they are both represented with mixture of Gaussians as joint probability distributions of y and x, respectively. To provide predicted ‘values’ for x and y, we are essentially required to calculate x and y values that have the highest probability density of the Gaussian mixture, respectively. However, in actuality, we simply use the weighted average of the mean of each Gaussian using weights as the predicted values of x and y. These values do not necessarily indicate the highest value of the probability density function (PDF) or the maximum value of the conditional probability density function. Therefore, in this study, true GMR is proposed to search for the predicted values of x and y, such that the value of the PDF is maximized. The PDF obtained using the Gaussian mixture is used as a nonlinear function to solve the continuous nonlinear optimization problem. This enables the values to maximize the true probability density of the GMR model while predicting y values from x values in forward analysis as well as the x values with the highest probability of achieving target y values, to be predicted in inverse analysis.

In inverse analysis, there can exist constraints on x. For example, for mole fractions, the sum of all the mole fractions must equal to one. Additionally, upper and lower limits exist for temperature and pressure, depending upon durability and safety of an equipment. Based on experimental conditions, discrete variables can be included in x, for example, presence or absence of an additive. True GMR can be applied by solving the continuous nonlinear optimization problem described before, when the constraints are upper and lower limits, and can be represented using linear and nonlinear equations; however, especially when discrete variables exist in x, the continuous nonlinear optimization problem cannot be solved. In this study, a search for x values with high values of PDF through utilization of a metaheuristics algorithm, especially GA, has been proposed. In GA, by preparing chromosomes to satisfy constraints and calculating the fitness of PDF, x values can be searched with a high value of PDF, thus satisfying the constraints on x. The proposed method is known as GA-based optimization with constraints using GMR (GAOC-GMR).

In this study, the proposed methods have been validated using complex nonlinear functions that facilitate calculation of y from x. Following the construction of the GMR model, the effectiveness of the proposed methods was confirmed by comparing values of the PDF for x values obtained using the traditional methods, with those for x values obtained using the proposed methods.

2. Method

Following the explanation of GMR and the traditional methods of calculating values predicted using the GMR model, the proposed methods of true GMR and GAOC-GMR have been described.

2.1. Gaussian mixture regression

GMR is a regression analysis approach based on the use of Gaussian mixture model (GMM) [4,12–14]. It constructs a model that expresses a dataset by superimposing multiple Gaussian distributions.

For a sample \( x \in \mathbb{R}^m \) (where m is the number of variables in x), the probability distribution function \( p(x) \) of the GMM is given as follows:

\[
p(x) = \sum_{k=1}^{n} \pi_k N(x; \mu_k, \Sigma_k),
\]

where, n is the number of Gaussians, and \( \mu_k \) and \( \Sigma_k \) are the mean vector and variance–covariance matrix of the kth Gaussian, respectively. \( \pi_k \) is the weight of the kth Gaussian, provided as follows:
\[ 0 \leq \pi_k \leq 1, \quad \sum_{k=1}^{n} \pi_k = 1. \] (2)

The expectation-maximization algorithm is used to maximize log-likelihood, and \( \mu_k, \Sigma_k \) and \( \pi_k \) can be determined. The log-likelihood function is given as follows:

\[
\log \{ p(x | \mu, \Sigma, \pi) \} = \sum_{i=1}^{N} \log \left\{ \prod_{k=1}^{n} \pi_k N(x_i | \mu_k, \Sigma_k) \right\},
\] (3)

where, \( N \) is the number of samples.

In GMR, the joint probability distribution of \( x \) and \( y \) is determined by calculating the GMM using a dataset that combines sample \( y \in \mathbb{R}^p \) of \( y \) (\( p \) is the number of \( y \) variables) and sample \( x \) of \( x \). Based on the multiplication theorem of probability and Bayes’ theorem, the \( y \) values can be predicted by calculating the posterior probability distribution \( p(y|x) \) when \( x \) is given; conversely, the \( x \) values can be predicted by calculating the posterior probability distribution \( p(x|y) \) when \( y \) is given. In other words, direct inverse analysis of the model is possible.

By explicitly separating \( x \) and \( y \) in Equation (1) of the GMM, the joint probability distribution of \( x \) and \( y \) is given as follows:

\[
p(x, y) = \sum_{i=1}^{n} \pi_i p(x | y, \mu_{x,i}, \Sigma_{x,i}) p(y | \mu_{y,i}, \Sigma_{y,i})
\] (4)

where, \( \mu_{x,i} \) and \( \mu_{y,i} \) are the mean vectors, and \( \Sigma_{x,i} \) and \( \Sigma_{y,i} \) are the variance–covariance matrices of \( x \) and \( y \) for the \( i \)th Gaussian, respectively, while \( \Sigma_{xy,i} \) and \( \Sigma_{yx,i} \) are the covariance matrices of \( x \) and \( y \) for the \( i \)th normal distribution, respectively.

Estimating \( x \) from \( y \) corresponds to determining the posterior probability distribution \( p(x|y) \) of \( x \) for a given \( y \), and \( p(x,y) \) can be transformed as follows using the multiplication theorem of probability and Bayes’ theorem:

\[
p(x|y) = \sum_{i=1}^{n} p(x|y, \mu_{y,i}, \Sigma_{yy,i}) p(\mu_{y,i}, \Sigma_{yy,i}) p(y | \mu_{y,i}, \Sigma_{yy,i})
\]

\[= \sum_{i=1}^{n} p(x|y, \mu_{y,i}, \Sigma_{yy,i}) \frac{1}{\sqrt{(2\pi)^n | \Sigma_{yy,i} |}} \exp \left\{ -\frac{1}{2} (y - \mu_{y,i})^T \Sigma_{yy,i}^{-1} (y - \mu_{y,i}) \right\}
\]

\[= \sum_{i=1}^{n} w_{y,i} p(x|y, \mu_{y,i}, \Sigma_{yy,i})
\] (5)

where, \( p(x|y, \mu_{y,i}, \Sigma_{yy,i}) \) is the multivariate Gaussian distribution of the estimated \( x \) value for the \( i \)th Gaussian and \( w_{y,i} \) is the weight of this multivariate Gaussian distribution. For \( p(x|y, \mu_{y,i}, \Sigma_{yy,i}) \), the mean vector \( \mu(y) \) and the variance–covariance matrix \( \Sigma(y) \) are given as follows:

\[
\mu(y) = \mu_{y,i} + (y - \mu_{y,i}) \Sigma_{yy,i}^{-1} \Sigma_{yx,i}
\] (6)

\[
\Sigma(y) = \Sigma_{xx,i} - \Sigma_{xy,i} \Sigma_{yy,i}^{-1} \Sigma_{yx,i}
\] (7)

In the above explanation, swapping \( x \) and \( y \) allows \( y \) to be estimated from \( x \). Because GMR does not distinguish between \( x \)-variables and \( y \)-variables, GMR can handle both multiple \( x \)-variables and multiple \( y \)-variables for both regression and inverse analyses.

In this study, the GMM calculation applied the Gaussian Mixture [15] from the scikit-learn library; the GMR calculation used DCEKit [16]. The hyperparameters in GMR are the number of Gaussian, and the type of variance–covariance matrix. In addition, there exist two traditional methods of estimating \( y \) from \( x \) (or \( x \) from \( y \)) 1) weighted mean: calculate the weighted mean using \( w_{y,i} \) in Equations (6), and 2) max mean: use the mean of the Gaussian that maximizes \( w_{y,i} \) in Equation (6).

### 2.2. True GMR

The prediction result of GMR is given as a PDF, which is a mixture of Gaussians described in Equations (5)-(8). To obtain predicted ‘values’ from the PDF, weighted mean and max mean for mixture of Gaussians given in section 2.1 were used traditionally. However, both, weighted mean and max mean do not consider the variance-covariance matrix of Equation (8). Furthermore, weighted mean and max mean do not provide \( x \) and \( y \) values at which the PDF in Equation (5) attains its maximum value.

In the proposed true GMR, the PDF in Equation (5) is handled as a continuous nonlinear function, and \( x \) and \( y \) values that provide the maximum value of the PDF are solved as a continuous nonlinear optimization problem. Figure 1 shows the basic concept of true GMR. The red, blue, and green lines denote probability density of three Gaussians whose weights are 0.20, 0.41, and 0.39, respectively, while the black line denotes probability density of a Gaussians mixture, which implies the true probability density of GMR given as Equation (5). Although, weighted mean and max mean do not follow
the probability density, true GMR indicates the highest value of the probability density. Figure 1 confirms the performance of true GMR.

To solve a continuous nonlinear optimization problem, scipy.optimize.minimize [17] was used in Python. In this study, sequential least squares programming [18], an iterative solution method for nonlinear optimization, has been used as a continuous optimization method. Sequential least squares programming solves a sequence of optimization subproblems, each of which optimizes a quadratic model of the objective subject to a linearization of the constraints. Since the weighted mean and max mean exist as the initial values, the problem is solved using each, and the solution with the largest probability density between the two results is adopted.

The proposed method can be used to find a solution if x variables have continuous values and the constraints are upper and lower bounds, and can be represented by linear and nonlinear equations, if constraints on x exist. However, true GMR cannot be applied if variables include a discrete variable. For example, true GMR cannot be used in the presence or absence of an additive (0 or 1) in polymer polymerization, and to change the temperature by only 1 K or 5 K due to equipment limitations.

The Python code for true GMR is available at https://github.com/hkaneko1985/dcekikit.

2.3. Genetic algorithm-based optimization with constraints using GMR (GAOC-GMR)

It is proposed to acquire x and y values that provide the maximum value of the PDF in Equation (5) using GA, which is a metaheuristic inspired by the natural selection process. GA prepares several chromosomes representing candidates for x and y values as genes, and searches for solutions by selecting chromosomes with high fitness values preferentially, and repeating operations, such as crossover and mutation. In this study, the fitness in GA is defined as the logarithm of Equation (5). Various constraints on x in inverse analysis of GMR model can be considered by devising chromosomes in GA. The following constraints on x can be considered:

- Upper and lower limits of variables
- Variables have only 0 or 1
- Values of variables are totaled, and the result is \( m_3 (\sum x_i = m_3) \)
- Values of variables are totaled, and the result is between \( m_4 \) and \( n_4 (m_4 \leq \sum x_i \leq n_4) \)
- Among variables, only \( m_5 \) variables have \( n_5 \) (dummy variables when \( m_5 = 1 \) and \( n_5 = 1 \))
- Among variables, only \( m_6 \) variables have non-zero values, and the rest have 0
- Variables have values down to the \( m_7 \)th decimal place or the \( n_7 \)th place

In (1), mole fractions possess values between 0 and 1, where upper and lower limits are necessary for synthesis conditions, such as temperature and pressure, depending upon the constraints of an equipment. Examples of variables that have only 0 or 1 in (2) are dummy variables, such as presence or absence of an additive, washing, and pretreatment of raw materials. All categorical variables can be represented using dummy variables. Variables with values other than 1 can be converted to 0 or 1 by scaling. In (3), when considering mole fractions of raw materials while mixing several raw materials, it is necessary to total them up to 1. Some sets of variables can have different total values. As given in (4), the total value of several variables are required to be in a certain range.

When a categorical variable with multiple categories is transformed into dummy variables, only one of the dummy variables is required to be 1 and the others must be 0, as in (5). When \( m_6 \) raw materials are selected for use, only \( m_6 \) variables would have nonzero values, while the others would be 0, as in (6). In (7), when preparing raw materials, it is necessary to round off the values, as the raw materials can only be adjusted to the \( m_7 \)th decimal place, and temperature and pressure can only be changed in certain units depending upon equipment. Therefore, the problem is categorized as discrete optimization, or combinatorial optimization.

In this study, the above constraints on x and y are considered while generating chromosomes in GA. Figure 2 shows the flow of setting up the chromosome in GAOC-GMR. The initial chromosomes are generated with a uniform random number between the lower and upper limits for each variable. In the initial chromosomes and the chromosomes generated in the GA operation, the values are first converted to 0 or 1 by rounding off in constraint 2). Next, in constraint 5), among the values of the chromosomes corresponding to each set of the assigned variables, up to \( m_5 \) variables in descending order, of the values are converted to \( m_5 \), while the rest of the variables in the set are converted to 0. Further, in constraint 6), among the values of the chromosomes corresponding to each set of the assigned variables, up to \( m_6 \) variables in descending order of the
values remain the same, while the rest of the variables in the set are converted to 0. Next, in constraint 3), the values of the chromosomes corresponding to each set of the assigned variables are totaled, and divided by the sum, and further multiplied by \( m_3 \). Finally, in constraint 7), the values are rounded off to the nearest digit. The fitness value of each chromosome that does not satisfy the constraints 1) and 4) is set to a very low value such as \(-10^{10}\).

DEAP [19] was used to calculate the GA. The Python codes for GAOC-GMR are available at https://github.com/hkaneko1985/dcekit.

3. Results and discussion

To test the performance of the proposed true GMR and GAOC-GMR, forward and inverse analyses were conducted assuming complex nonlinear functions between \( x \) and \( y \) to be a system, where \( y \) can be obtained as a result of experiments based on \( x \). The nonlinear functions are based on the Rastrigin function [20] and Griewank function [21]. These are represented as follows:

\[
y = 10d + \sum_{i=1}^{d} (x_i^2 - 10 \cos(2\pi x_i)) \tag{9}
\]

\[
y = \frac{1}{4000} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1 \tag{10}
\]

where, \( d \) is the number of variables in \( x \), and \( x_i \) is the \( i \)th \( x \).

In this study, the following six case studies were conducted:

- Case study 1: Using the Rastrigin function, \( y \) is one variable and \( x \) is 25 variables
- Case study 2: Using the Griewank function, \( y \) is one variable and \( x \) is 30 variables
- Case study 3: Using Rastrigin and Griewank functions, \( y \) is two variables and \( x \) is 30 variables
- Case study 4: Using the Rastrigin function, \( y \) is one variable, \( x \) is 25 variables, and there exist constraints on \( x \)
- Case study 5: Using the Griewank function, \( y \) is one variable, \( x \) is 30 variables, and there exist constraints on \( x \)
- Case study 6: Using Rastrigin and Griewank functions, \( y \) is two variables, \( x \) is 30 variables, and there exist constraints on \( x \)

Two variables of \( y \) in case studies 3 and 6 are represented as \( y_1 \) and \( y_2 \). The constraints on \( x \) for case study 4 are given as follows:

- Variables from \( x_1 \) to \( x_7 \) have only 0 or 1 values
- One of the variables from \( x_5 \), \( x_6 \), and \( x_7 \) has a value of 1 while the others have a value of 0
- Variables from \( x_8 \) to \( x_{11} \) have values between 0 and 1
- Values of variables from \( x_8 \) to \( x_{11} \) are totaled, and the result is 1
- Values of variables from \( x_8 \) to \( x_{11} \) are rounded off to the second decimal place
- Variables from \( x_{12} \) to \( x_{16} \) have values between 0 and 1
- Values of variables from \( x_{12} \) to \( x_{16} \) are totaled, and the result is 1
- Values of variables from \( x_{12} \) to \( x_{16} \) are rounded off to the third decimal place
• Variables from x1 to x10 have values between 0 and 10
• Values of variables from x17 to x20 are rounded off to the first decimal place
• Variables from x21 to x25 have values between 10 and 200
• Values of variables from x21 to x25 are rounded off to the tens place

The constraints on x for case studies 5 and 6 are given as follows:

• Variables from x1 to x10 have values between 0 and 1
• Among variables from x1 to x10, only five variables have nonzero values, while the rest have a value of 0
• Values of variables from x1 to x10 are totaled, and the result is 1
• Values of variables from x1 to x10 are rounded off to the second decimal place
• Variables from x11 to x15 have only 0 or 1 values
• Variables from x16 to x20 have values between 0 and 5
• Values of variables from x16 to x20 are totaled, and the result lies between 5 and 10
• Values of variables from x16 to x20 are rounded off to the first decimal place
• Variables from x21 to x30 have values between 10 and 30
• Values of variables from x21 to x30 are rounded off to the ones place

The number of Gaussians is 20, and the type of variance-covariance matrix is ‘full’ in scikit-learn’s sklearn.mixture.GaussianMixture [15] in GMR, and further, 1000 samples generated by random numbers were used to construct the GMR model for each case study.

There was no significant difference between the estimation results of weighted mean, max mean, and true GMR in forward analysis of the GMR models. In the supplementary material, estimation results using weighted mean, max mean, and true GMR in case studies 1, 2, and 3 are shown in Table S1, and the plots of actual y values and estimated y values in case studies 1, 2, and 3 are shown in Figures S1, S2, and S3, respectively. It was confirmed that true GMR can estimate y values with the same level of accuracy as the traditional methods, in forward analysis.

Then, inverse analysis of the GMR model was conducted to compare the performance of the traditional methods with that of the proposed methods. In case studies 1 and 4, the target y values were set to 200, 300, 400, 500, 600, and 700, where 200 and 700 indicated the extrapolation domain of y. In case studies 2 and 5, the target y values were set to 100, 300, 500, 700, 900, 1100, 1300, and 1500, where 100 and 1500 indicated the extrapolation domain of y. In case studies 3 and 6, the target values of (y1, y2) were set to (100, 100), (200, 300), (300, 700), (400, 500), (500, 800), (600, 1200), and (700, 1400), where (100, 100) and (700, 1400) indicated the extrapolation region of y.

Tables 1–3 show logarithm of probability density for each target y value in case studies 1, 2, and 3, respectively. True GMR obtained x values with a probability density higher than that of max mean. Furthermore, for certain target y values, true GMR obtained x values with a probability density higher than that of weighted mean. However, for the other multiple target y values, the probability density of the proposed method was the same as that of weighted mean and max mean. This can be attributed to the fact that when the weights are concentrated in a Gaussian distribution, the probability

| Table 1. Logarithm of probability density for each target value y in case study 1. |
|---|
| Target y | Weighted mean | Max mean | True GMR | GAOC-GMR |
| 200 | −16.5 | −31.0 | −16.5 | −16.7 |
| 300 | −16.0 | −22.3 | −16.0 | −15.9 |
| 400 | −15.0 | −23.4 | −15.0 | −15.3 |
| 500 | −19.3 | −28.5 | −19.3 | −17.5 |
| 600 | −18.4 | −31.4 | −18.4 | −17.2 |
| 700 | −17.2 | −37.6 | −17.2 | −17.7 |

| Table 2. Logarithm of probability density for each target value y in case study 2. |
|---|
| Target y | Weighted mean | Max mean | True GMR | GAOC-GMR |
| 100 | −18.3 | −26.7 | −18.3 | −18.8 |
| 300 | −18.7 | −27.3 | −18.7 | −18.4 |
| 500 | −21.1 | −28.6 | −18.8 | −19.0 |
| 700 | −21.0 | −31.5 | −20.2 | −19.6 |
| 900 | −21.2 | −35.0 | −19.6 | −20.3 |
| 1100 | −19.8 | −38.7 | −19.8 | −21.4 |
| 1300 | −18.8 | −40.5 | −18.9 | −20.4 |
| 1500 | −18.5 | −40.9 | −18.5 | −21.2 |

| Table 3. Logarithm of probability density for each target value y in case study 3. |
|---|
| Target y | Weighted mean | Max mean | True GMR | GAOC-GMR |
| 100 | −19.0 | −19.0 | −18.9 | −19.6 |
| 200 | −19.0 | −19.0 | −19.0 | −19.4 |
| 300 | −19.0 | −19.2 | −19.0 | −19.4 |
| 400 | −19.2 | −19.2 | −19.2 | −19.4 |
| 500 | −19.8 | −26.2 | −19.8 | −20.3 |
| 600 | −18.6 | −32.0 | −18.6 | −20.8 |
| 700 | −19.4 | −33.8 | −19.4 | −20.9 |

| Table 4. Logarithm of probability density for each target value y in case study 4. |
|---|
| Target y | Weighted mean | Max mean | GAOC-GMR |
| 200 | 11.1 | 5.0 | 11.5 |
| 300 | 7.5 | −1.1 × 10^6 | 11.4 |
| 400 | 3.5 | −1.1 × 10^6 | 9.6 |
| 500 | 3.5 | −1.1 × 10^6 | 10.5 |
| 600 | 9.0 | −1.1 × 10^6 | 9.1 |
| 700 | 8.1 | 8.3 | 9.3 |
density of max mean is the global maximum, which in itself is not an issue. However, continuous nonlinear optimization was more complicated in the case studies due to the large number of variables, such as 25 and 30 variables in true GMR.

GAOC-GMR improved the values of the probability density of weighted mean, max mean, and true GMR for certain target y values. In true GMR, reasonable x values of weighted mean and those of max mean are used as initial values, and thus, x values with a high probability density are obtained stably. However, the global optimal solution is not guaranteed. Even when true GMR attains a local optimal solution, GAOC-GMR might obtain x values with a higher probability density. However, as GAOC-GMR is a heuristic method, the probability density may be lower than that of true GMR, in which case it is better to use the result of true GMR. It was confirmed that both true GMR and GAOC-GMR can be used to perform inverse analysis of the GMR model by predicting x values that result in a stably high probability density value.

The results of inverse analysis of the GMR models for case studies 4, 5, and 6, where x is constrained, are shown in Tables 4–6, respectively. In the cases of weighted mean and max mean, following the prediction of x values, they were set to be the closest to the constraints on x. From Tables 4–6, weighted mean and max mean resulted in much lower values of probability density due to the constraints on x. However, using GAOC-GMR, the values of probability density were improved even when the constraints on x existed. Furthermore, even when the y values were in the extrapolation region, x values with high probability density values were obtained, and the inverse analysis was stable. It was confirmed that the proposed method could be successfully used to perform inverse analysis of the GMR models.

Further, to test the performance of the proposed method, inverse analysis was conducted for the real material datasets. The datasets of thermoelectric conversion (ZT) materials [22] and superconducting (Tc) materials [23, 24] were used. In Tc, the number of samples is 21263, x is the fraction of each metal element, and y is the critical temperature. In ZT, the number of samples is 1394, x is the fraction of each metal element, and y is the efficiency of thermoelectric conversion [25], which is calculated as:

$$ZT = \frac{S^2 \sigma}{\kappa} T$$ (11)

where, $\kappa$, $S$, and $\sigma$ are the thermal conductivity, the Seebeck coefficient, and the electrical conductivity, respectively, while $T$ is the average temperature of the material proportional to conversion efficiency. Fivefold cross-validation was performed to determine the hyper-parameters for GMR. The number of Gaussian was 28 and the type of variance–covariance matrix was ‘spherical’ in scikit-learn’s sklearn.mixture.GaussianMixture [15] in ZT, and the number of Gaussian was 29 and the type of variance–covariance matrix was ‘spherical’ in Tc.

The purpose of this study is to determine a large probability density in the probability distribution that is obtained through the prediction results of forward and inverse analyses in GMR, and not to improve the predictive ability of GMR. The predictive ability of GMR in ZT and Tc has already been discussed in the literature [6, 12, 13].

Figure 3 shows histograms of ZT and Tc. The target values for ZT and Tc were set to 1.5 and 150, respectively, and inverse analysis of the GMR model was conducted.

### Table 5. Logarithm of probability density for each target value y in case study 5.

| Target y | Weighted mean | Max mean | GAOC-GMR |
|----------|---------------|----------|-----------|
| 100      | −22.1         | −27.1    | −18.7     |
| 300      | −20.1         | −30.3    | −18.1     |
| 500      | −24.6         | −30.3    | −19.8     |
| 700      | −28.0         | −32.1    | −21.0     |
| 900      | −45.7         | −41.9    | −18.4     |
| 1100     | −39.6         | −33.5    | −18.5     |
| 1300     | −18.5         | −41.5    | −17.2     |
| 1500     | −53.6         | −42.8    | −23.1     |

### Table 6. Probability density for each target value y in case study 6.

| Target y1 | Target y2 | Weighted mean | Max mean | GAOC-GMR |
|-----------|-----------|---------------|----------|-----------|
| 100       | 100       | −23.6         | −20.2    | −18.6     |
| 200       | 300       | −25.0         | −20.9    | −18.6     |
| 300       | 700       | −23.9         | −33.7    | −16.4     |
| 400       | 500       | −24.4         | −39.2    | −14.2     |
| 500       | 800       | −21.9         | −31.5    | −16.7     |
| 600       | 1200      | −20.1         | −21.8    | −18.3     |
| 700       | 1400      | −20.8         | −24.5    | −19.3     |
The sum of the fractions of metal elements was constrained to be one, and the fraction of each metal element was constrained to have values rounded off to the third decimal place. To set constraints on x, the number of metals should be set first. In this study, the number of metal element types used was changed to 2, 3, 4, 5, 6, 7, 8, 9, and 10, and the fraction of the metals were designed with GAOC-GMR. Tables 7 and 8 show logarithm of probability density for the target y value in ZT and Tc, respectively. GAOC-GMR successfully improved the values of probability density for all numbers of metal elements in both, ZT and Tc, even when the y values were in the extrapolation region. It was confirmed that GAOC-GMR could be used to perform inverse analysis of GMR model by predicting x values that result in a stably high probability density value.

4. Conclusions

In this study, we developed the methods for accurate calculation of predicted ‘values’ for y and x in forward and inverse analyses of GMR models, respectively, while the prediction results were given by Gaussian mixtures. Although predicted values were conventionally approximated by weighted mean and max mean, we proposed true GMR, a method in which the point with the highest value of the PDF is solved using continuous nonlinear optimization and used as the prediction result. Furthermore, when discrete variables exist, GAOC-GMR can be used to search for the point that maximizes the PDF by appropriately combining chromosomes with other constraints, and using the value of the PDF as the fitness in GA. True GMR can be executed only when constraints on variables are upper and lower bounds and represented as linear and nonlinear equations; however, GAOC-GMR can be executed with any constraints, including discrete variables.

The effectiveness of the proposed methods was verified using complex nonlinear functions between x and y, where x was assumed to be the experimental conditions and y was assumed to be the experimental results, and real material datasets. It was confirmed that true GMR could provide solutions with higher values of the probability density, than traditional methods. Furthermore, GAOC-GMR possesses the ability to search for much higher values of the probability density even when various constraints are present, and target y values are in the extrapolation region. In true GMR, reasonable x values of weighted mean and those of max mean are used as initial values, and thus, x values with a high probability density are obtained stably. However, a global optimal solution is not guaranteed. Even when true GMR falls into a local optimal solution, GAOC-GMR would obtain x values with the higher probability density. However, since GAOC-GMR is a heuristic method, the probability density can be lower than that of true GMR, in which case it is better to use the result of true GMR. Therefore, both, true GMR and GAOC-GMR can be used to perform inverse analysis of the GMR model by predicting x values that result in a stably high probability density value.

Direct inverse analysis with GMR using the proposed methods is expected to improve the efficiency of molecular, material, and process design. The Python codes for true GMR and GAOC-GMR are available at https://github.com/hkaneko1985/dcekit.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by a Grant-in-Aid for Scientific Research (KAKENHI) [Grant Numbers 19K15352, 20H02553, and 20H04538] from the Japan Society for the Promotion of Science and the Center of Innovation (COI) program [Grant Number R1WD12] from the Japan Science and Technology Agency.

ORCID

Hiromasa Kaneko http://orcid.org/0000-0001-8367-6476

References

[1] Shahriari B, Swersky K, Wang ZY, et al. Taking the human out of the loop: a review of Bayesian optimization. Proc IEEE. 2016;104:148–175.
[2] Iwama R, Kaneko H. Design of ethylene oxide production process based on adaptive design of experiments and Bayesian optimization. J Adv Manuf Proc. 2021;3:e10085.

[3] Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. Multimted Tools Appl. 2021;80:8091–8126.

[4] Kaneko H. Adaptive design of experiments based on Gaussian mixture regression. Chemom Intell Lab Syst. 2021;208:104226.

[5] Kaneko H. Data visualization, regression, applicability domains and inverse analysis based on generative topographic mapping. Mol Inf. 2019;38:1800088.

[6] Shimizu N, Kaneko H. Direct inverse analysis based on Gaussian mixture regression for multiple objective variables in material design. Mater Design. 2020;196:109168.

[7] Yamada S, Chikayama E, Kikuchi J. Signal deconvolution and generative topographic mapping regression for solid-state NMR of multi-component materials. Int J Mol Sci. 2021;22:1086.

[8] Wang G, Qian L, Guo Z. Continuous tool wear prediction based on Gaussian mixture regression model. Int J Adv Manuf Technol. 2013;66:1921–1929.

[9] Yao L, Ge Z. Nonlinear Gaussian mixture regression for multimode quality prediction with partially labeled data. IEEE Trans Industr Inform. 2019;15:4044–4053.

[10] Ye C, Yang J, Ding H. Bagging for Gaussian mixture regression in robot learning from demonstration. J Intell Manuf. 2020. https://doi.org/10.1007/s10845-020-01686-8.

[11] Zheng J, Du W, Lang Z, et al. Modeling and optimization of the cement calcination process for reducing NOx emission using an improved just-in-time gaussian mixture regression. Ind Eng Chem Res. 2020;59:4987–4999.

[12] Kaneko H. Extended Gaussian mixture regression for forward and inverse analysis. Chemom Intell Lab Syst. 2021;213:104325.

[13] Kaneko H. Lifting the limitations of Gaussian mixture regression through coupling with principal component analysis and deep autoencoding. Chemom Intell Lab Syst. 2021;218:104437.

[14] Bishop CM. Pattern recognition and machine learning. New York (NY): Springer; 2006.

[15] [cited 2021 Aug 28]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html

[16] [cited 2021 Aug 28]. Available from: https://datachemeng.com/deckit/

[17] Available from: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html (accessed 2021 Aug 28)

[18] [cited 2021 Aug 28]. Available from: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize-slsqp.html

[19] [cited 2021 Aug 28]. Available from: https://deap.readthedocs.io/en/master/

[20] [cited 2021 Aug 28]. Available from: https://www.sfu.ca/~ssurjano/rastr.html

[21] Griewank AO. Generalized descent for global optimization. J Optim Theory Appl. 1981;34:11–39.

[22] Katsura K, Kumagai M, Kodani T, et al. Data-driven analysis of electron relaxation times in PbTe-type thermoelectric materials. Sci Technol Adv Mater. 2019;20:511–520.

[23] Hamidieh K. A data-driven statistical model for predicting the critical temperature of a superconductor. Comput Mater Sci. 2018;154:346–354.

[24] [cited 2021 Aug 28]. Available from: https://archive.ics.uci.edu/ml/datasets/Supercconductivity+Data#

[25] Bell LE. Cooling, heating, generating power, and recovering waste heat with thermoelectric systems. Science. 2008;321:1457–1461.