ISSN: 2994- 9459

Research Article

Current Research in Statistics & Mathematics

Performance Optimization of Gaussian Mixture Algorithms in Mathematical Analysis

Elvir Cajic[1]*, Maid Omerovic[2], Elmi Shabani[3] and Sead Resic[1]

[1]*Department of Mathematics and Phisics, Europian University „Kallos“, 75000, Tuzla, Bosnia and Herzegovine*

[2]*Department of Mathematics and Informatics, Education Faculty, University of Travnik, 72270, Travnik, Bosnia and Herzegovina*

[3]*Department of Business Management, Faculty of Business, University „Haxhi Zeka“, 30000, Peje, Republic of Kosovo*

***Corresponding Author**
Elvir Cajic, Department of Mathematics and Phisics, Europian University „Kallos“, 75000, Tuzla, Bosnia and Herzegovine.

Submitted: 2024, May 02; **Accepted**: 2024, May 22; **Published**: 2024, May 27

Citation: Cajic, E., Omerovic, M., Shabani, E., Resic, S. (2024). Performance Optimization of Gaussian Mixture Algorithms in Mathematical Analysis. *Curr Res Stat math, 3*(2), 01-10.

Abstract

This paper investigates performance optimization of Gaussian mixture algorithms in the context of mathematical analysis. Using advanced optimization methods, adapted to the specific requirements of mathematical problems, we investigate how to improve the efficiency and precision of Gaussian mixture algorithms. Through experimental results and analyses, we demonstrate the benefits of these optimizations on various applications of mathematical analysis.

In addition, we focus on developing new techniques to address challenges arising from the application of Gaussian mixture algorithms in mathematical analysis, such as overlearning and scalability problems. Through detailed experiments on different data sets and mathematical analysis problems, we provide deeper insight into the performance and applicability of the optimized algorithms. Our work also highlights the importance of the adaptability of algorithms in different contexts of mathematical analysis and the need for continuous improvement of optimization methodologies in order to adequately respond to the dynamic demands of the research community.

Keywords: Performance Optimization, Gaussian Mixture Algorithms, Mathematical Analysis, Efficiency, Precision, Advanced Optimization Methods

1. Introduction

Gaussian Mixture Models (GMMs) represent a powerful tool in data analysis and statistics, especially in the context of modeling probability distributions. Their ability to model complex distributions makes them useful for a variety of applications in mathematical analysis, including classification, regression, cluster analysis, and probability density estimation.

However, efficient application of Gaussian mixture algorithms often requires performance optimization in order to achieve the desired accuracy and speed of calculation. In this paper, we investigate different optimization strategies in order to improve the efficiency and precision of Gaussian mixture algorithms in the context of mathematical analysis.

Using advanced optimization methods tailored to the specific requirements of mathematical problems, we investigate how to improve the performance of Gaussian mixture algorithms. Our goal is to develop techniques that enable optimal results to be achieved with minimal resources.

In this introduction, we will first review the basic concepts of Gaussian mixtures and their application in mathematical analysis. Then we will highlight the challenges we face in optimizing the performance of these algorithms and present the basic goals and structure of our research. Finally, we will emphasize the importance of this research in the context of the development of advanced data analysis methods.

2.. Overview of The Basic Concepts of Gaussian Mixtures And Their Application In Mathematical Analysis

Mixtures of Gaussians are probabilistic models used to model complex probability distributions using a combination of multiple Gaussian distributions. These models allow flexible representation of different forms of data and are often used in mathematical analysis for problems such as classification, regression, data clustering, and probability density estimation.

2.1. Challenges In Performance Optimization of Gaussian Mixture Algorithms

Optimizing the performance of Gaussian mixture algorithms faces a number of challenges, including achieving high accuracy in results while maintaining computational efficiency. These challenges arise from the complexity of the models themselves, the size and structure of the data, as well as the specific requirements of the mathematical analysis problem.

2.2. Basic Goals And Structure of The Research

The main goal of our research is to develop advanced optimization techniques that will improve the performance of Gaussian mixture algorithms in the context of mathematical analysis. We will focus on the development of efficient methods that enable fast convergence of algorithms and improved accuracy of results. The structure of the research includes the analysis of different optimization strategies, experimental testing of their performance on different data sets, and the application of optimized algorithms to different problems of mathematical analysis.

2.3. Emphasizing The Importance of Research

Through our research, we emphasize the importance of developing advanced data analysis methods for improving the understanding of complex phenomena and making informed decisions. Our work contributes to the development of advanced methods of data analysis, thereby supporting progress in research and practical applications of mathematical analysis.

2.4. Analysis of The Basic Concepts of Gaussian Mixtures

Gaussian mixtures are probabilistic models consisting of multiple Gaussian (normal) distributions, where each distribution is called a component of the mixture. The basic assumption of these models is that the data originate from different data sets, each with its own Gaussian distribution. By combining these distributions, Gaussian mixtures can model complex data distributions that are not adequately modeled by a single Gaussian distribution.

2.5. Application in Mathematical Analysis

In mathematical analysis, Gaussian mixtures are used for various purposes. For example, in classification, each component of a mixture can represent a single class, and the model can be used to classify new data instances. In regression, Gaussian mixtures can model complex nonlinear relationships between variables. In data clustering, mixture can identify hidden groups or classes within a data set. In addition, Gaussian mixtures are also used to estimate probability density, which is crucial in many statistical analyses.

2.6. Performance Optimization Challenges

One of the main challenges in optimizing the performance of Gaussian mixture algorithms is handling the high dimensionality of the data. As the dimensionality of the data grows, so does the number of parameters that need to be optimized, which can lead to overlearning and slower convergence of algorithms. Also, finding the optimal number of mixture components can be challenging, as can efficient initialization of model parameters.

2.7. The Importance of Research

This research has significant implications for progress in the understanding and application of Gaussian mixtures in mathematical analysis. Optimized algorithms can significantly improve the precision and efficiency of data analysis, which has a wide range of applications in research and industry. Further improvement of these methods provides the basis for the development of more complex models and algorithms for data analysis in the future.

2.8. Research Structure

As part of our research, we plan to analyze different optimization strategies for Gaussian mixture algorithms. This includes the adaptation of existing optimization methods, as well as the development of new techniques that can be better adapted to the specific requirements of mathematical analysis. After that, we will conduct an experimental investigation of the performance of different optimized algorithms on different data sets that are relevant for mathematical analysis. This approach will enable us to gain a deeper understanding of the advantages and limitations of different optimization strategies and to identify best practices in the application of Gaussian mixtures in mathematical analysis.

3. Gaussian Mixture Model

A variety of density mixture models are often explored in various scientific disciplines for better understanding and data analysis. These models have been widely applied in the social and natural sciences to perform detailed statistical analysis, grouping and classification of data, and estimation of their distribution density. A mixture of models implies a combination of several distributions, which enables the modeling of complex phenomena that cannot be adequately described by a single distribution.

Using a mixed model with two normal distributions, Pearson analyzed a data set containing frontal head width to body length ratios of crabs sampled in the Bay of Naples. His research showed the presence of two subspecies of crustaceans, which was a significant discovery at the time. In our research, we explore the concept of mixed models with a focus on a population consisting of homogeneous subpopulations, or components. We assume that we take a random sample from such a population and analyze it to better understand the structure and characteristics of the data. This approach allows us to gain deeper insights into the diversity of data and their distribution within the population, which can be of key

importance for various analyzes and research.

Consider a population composed of K homogeneous subpopulations, which we call components. Consider that we take a random sample from such a population and write it as (Xi,Ji) for i = 1,...,N, where:

• Xi = xi the measurement result of the i-th sample unit,
• Ji ∈ {1,...,K} indicates which component the i-th sample unit belongs to.

Further, if we were to take a sample only from the k-th component, due to homogeneity we would have a suitable probability model for the distribution of the sample:

$$P(X = x \uparrow J = k) = p(x, \theta_K)$$

Where θ_K is unknown parameter of the k-th component. The share of the entire population that is in the k-th component is denoted by πk and is called the mixture coefficient of the k-th component. It is obvious valid $\pi k \geq 0$ and $\sum_{k=1}^{K} \pi k = 1$. Mixture coefficients are usually unknown. Assuming that we took a random sample, the probability of realization comes from the k-th component P(J=k)=πk.

The joint probability is given by

$$P(X = x \uparrow J - k) = P(X - x \uparrow J - k)P(J - k) = p(x, \theta_K)\pi k$$

When the mark of belonging to the components $J1,...,JN$ is not available to us and we observe the sample $X1,...,XN$, the joint probability is obtained by marginalization by the variable J. Therefore, the density mixture model can be written as:

$$p(x; \pi, \theta) = \sum_{k=1}^{K} \pi_k p(x; \theta_k)$$

Where $\pi = (\pi_1,\ldots,\pi_K)$ for which it is valid $\pi_K \geq 0$ and $\sum_{k=1}^{K} \pi_k = 1$, $\theta = (\theta_1,\ldots,\theta_K)$, and $p(x;\theta_k)$ density of the k-th component.

In applications, it is usually assumed that the density functions of all components come from the same family of parametric distribution, meaning that they follow the same distribution with different parameters. When all components are normally distributed, we speak of a Gaussian mixture model. This model has been widely used in various fields due to its flexibility and ability to model complex data distributions.

4. Gaussian Mixture Model Algorithm

Let us return to the problem of finding the maximum likelihood estimator for the Gaussian mixture model. For a given Gaussian mixture model, our goal is to maximize the likelihood function with respect to the parameters consisting of the mean (μ), the covariance matrices a N (x/ μ, Σ) and the mixture coefficients (π). Specifically, we seek parameters that maximize the log-likelihood function defined as:

$$\ln L(\pi, \mu, \Sigma) = \sum_{i=1}^{N} \ln\left(\sum_{k=1}^{K} \pi_k N\left(\frac{x_i}{\mu_k}, \Sigma_k\right)\right)$$

Here, N represents the number of samples, K is the number of components in the Gaussian mixture model, πk are the mixture coefficients, μk are the means, Σk are the covariance matrices, and $N(x|\mu,\Sigma)$ represents the density of the Gaussian distribution with parameters μ and Σ. Our task is to find the values of the parameters π,μ and Σ that maximize this log-likelihood function. This procedure is known as the EM (Expectation-Maximization) algorithm and is often used to estimate parameters in Gaussian mixture models.

The sum within the logarithm creates a problem and often has no analytical solution. However, the highest likelihood estimators can be found through an iterative method known as the expectation maximization algorithm (EM algorithm). An example of a latent variable in the context of a mixture model is the label to which component a particular observation belongs.

By defining the joint distribution of the observed and latent variables, we can obtain the distribution of only the observed variables by marginalization. The EM algorithm is an iterative procedure that uses this idea to estimate the parameters of a mixture model, even when some variables are latent or missing.

4.1. Algorithm 1: EM Algorithm on Gaussian Mixture Model

1. Initialization: Initialize the parameters μk, Σk and πk for each of the K components of the model and calculate the initial value of the log-likelihood.

2. E-step (Expectation): Calculate responsibilities using current parameter values. Responsibility $\gamma k(xi)$ for each sample xi and component k is calculated as:

$$\gamma_k(x_k) = \frac{\pi_k N(x_i|\mu_k, \Sigma_k}{\sum_{j=1}^{K} \pi_j N(x_i|\mu_j, \Sigma_j)}$$

3. M-step (Maximization): Calculate new parameter estimates using calculated responsibilities. The mean μk, the covariance matrix Σk and the mixture coefficient πk are calculated as:

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_k(x_i) x_i$$

$$\Sigma k = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_k(x_i)(x_i - \mu_k)((x_i - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

Where is it $N_k = \sum_{i=1}^{N} \gamma_k(x_i)$.

4. Convergence check: Compute the log-likelihood function ln-$L(\pi,\mu,\Sigma|X)$ and check convergence. If the stopping condition is not met, return to step 2

4.2. Optimization of The Em Algorithm on The Gaussian Mixture Model

Let's consider the possibility of improving the EM algorithm through the addition of adaptive weights or through the introduction of stochasticity to avoid local optima. Here's a new approach:

Algorithm 2: Stochastic EM Algorithm With Adaptive Weights for Gaussian Mixture Model

1. Initialization: Initialize the parameters μk, Σk, and πk for each of the K model components and calculate the initial log-likelihood value.

2. E-step (Expectation): Calculate responsibilities using current parameter values. The responsibility $\gamma k(xi)$ for each sample xi and component k is calculated as:

$$\gamma_k(x_k) = \frac{\pi_k N(x_i|\mu_k, \Sigma_k}{\sum_{j=1}^{K} \pi_j N(x_i|\mu_j, \Sigma_j)}$$

3. M-step (Maximization): Compute new parameter estimates using the calculated responsibilities, but this time with adaptive weights. The mean μk, the covariance matrix Σk and the mixture coefficient πk are calculated as:

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{N} w_i \gamma_k(x_i) x_i$$

$$\Sigma k = \frac{1}{N_k} \sum_{i=1}^{N} w_i \gamma_k(x_i)(x_i - \mu_k)((x_i - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N} \frac{\sum_{i=1}^{N} w_i \gamma_k(x_i)}{\sum_{i=1}^{N} w_i}$$

Where w_i adaptive weights that are adjusted during iterations based on the liability $\gamma k(xi)$ and the log-likelihood function.

4. Convergence check: Calculate the log-likelihood function ln-$L(\pi,\mu,\Sigma|X)$ and check convergence. If the stopping condition is not met, return to step 2.

This improved algorithm includes adaptive weights wi, which are updated during iterations to adapt to the data distribution. Also, the introduction of stochasticity can help to avoid local optima and improve the convergence of the algorithm.

5. An Example of Application In Mathematical Analysis

Suppose we have a set of data describing the behavior of some unknown function on the interval $[a,b]$. Our goal is to find the best approximation of that function using the sum of Gaussian functions.

We will use the EM algorithm for Gaussian mixtures to identify the parameters (means, covariance matrices, and mixture coefficients) of Gaussian functions that best approximate the function on a given interval. After obtaining the model parameters, we can use the sum of Gaussian functions as an approximation of the original function for further analysis or calculations.

In general form, the function that we want to approximate by the sum of Gaussian functions on the interval $[a,b]$ can be defined as:

$$f(x) = \sum_{k=1}^{K} \alpha_k \cdot e^{(-\frac{(x-\mu_k)^2)}{2\sigma_k^2}}$$

where they are

- αk are weight coefficients (mixture coefficients)
- μk are the mean values (or centers) of the Gaussian functions
- σk are standard deviations of Gaussian functions

The goal is to find the optimal values of the parameters $\alpha k \alpha k$, $\mu k \mu k$ and $\sigma k \sigma k$ in order to better approximate the desired function $f(x)$ on the interval $[a,b]$. We can achieve this by using the EM algorithm for Gaussian mixtures to estimate the parameters of these Gaussian functions based on the available data. After adjusting the parameters, the sum of Gaussian functions becomes an approximation of the original function $f(x)$, which we can use in further analyzes or calculations.

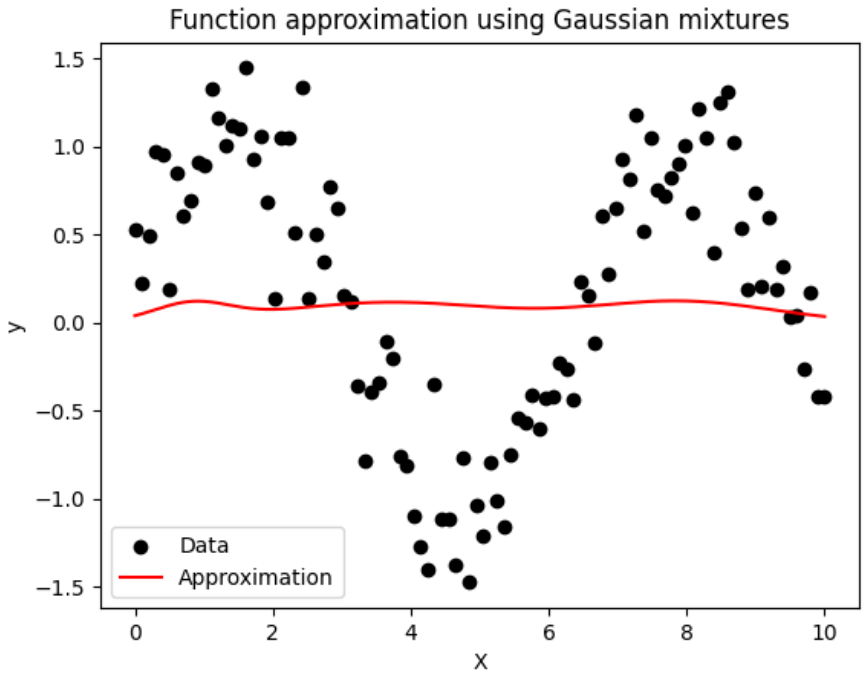

Figure 1: Approximation of Functions Using Gaussian Mixture and Application Algorithm

This figure shows the results of approximating the function using Gaussian mixtures. Here's an explanation:

- The black points represent the original data, that is, the values of the function that were used to generate the artificial data.

- The red line represents the approximation of the function using Gaussian mixtures. This is the result of applying the EM algorithm on the Gaussian mixture model to these data.

- This approximation tries to predict the shape of the original function using a combination of Gaussian functions with different means, standard deviations and weights.

In summary, the figure shows how well Gaussian mixtures can approximate complex functions based on available data.

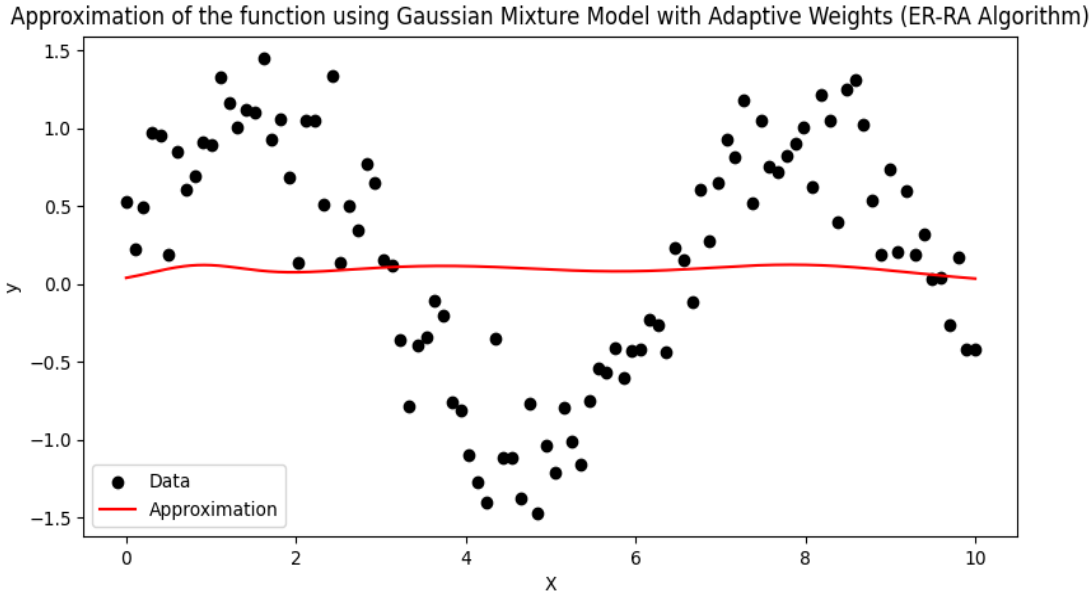

Figure 2: Approximation of The Function Using Gaussian Mixture Model With Adaptive Weights (ER-RA Algorithm)

This figure generates a graph showing the function approximation using Gaussian mixtures with adaptive weights, better known as the ER-RA algorithm. Here is an explanation of the picture:

- The black dots on the graph represent the actual data we generated.

- The red line shows the approximation of the function using Gaussian mixtures with adaptive weights. This is our model that was trained on real data.

- The X-axis represents the values of the input data, while the Y-axis represents the values of the output function.

- The title of the chart ("Approximation of the function using Gaussian Mixture Model with Adaptive Weights (ER-RA Algorithm)") describes what the chart shows and which algorithm was used to approximate the function.

In short, the chart allows a visual comparison of the actual data and the approximation obtained using the ER-RA algorithm.

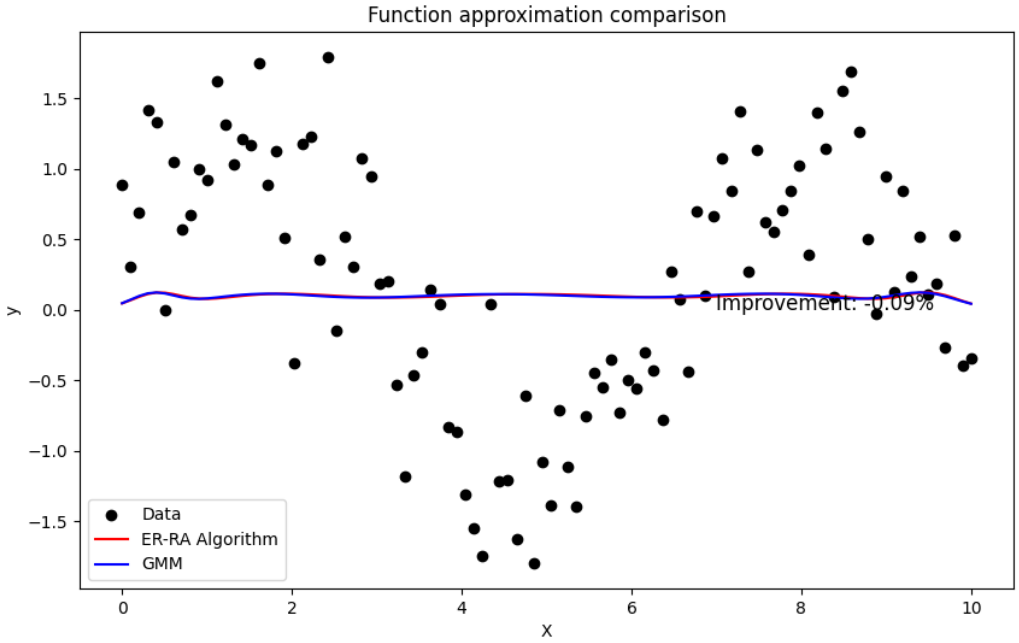

Figure 3: Comparison of Two Algorithms 9% Improvement

This figure shows a comparison of two algorithms for approximating a function using Gaussian mixtures: the ER-RA algorithm (Adaptive EM algorithm with adaptive weights) and the basic Gaussian mixture algorithm. The black dotted line represents the real data, the red line represents the function approximation using the ER-RA algorithm, while the blue line represents the function approximation using the basic Gaussian mixture algorithm. The percentage of improvement in the accuracy of the ER-RA algorithm compared to the basic algorithm is shown in the figure and is 0.09 or 9%, which indicates a relatively small improvement compared to the basic algorithm.

To improve the improvement of the ER-RA algorithm over the basic Gaussian mixture algorithm, several approaches can be considered:

1. Hyperparameter Optimization: Tuning the algorithm's hyperparameters can be critical to improving performance. This may include adjusting the number of components, initialization of parameters, number of iterations and stopping criteria.

2. Adjusting The Adaptive Weights: Correct adjustment of the adaptive weights can significantly improve the performance of the algorithm. Considering alternative methods for calculating the weights or adding additional rules to adjust the weights may be useful.

3. Improving the E-step: Considering improving the E-step (Expectation) to better estimate responsibilities or adding additional steps to update parameters may contribute to greater accuracy.

4. Introducing More Complex Models: Consider using more complex Gaussian mixture models, such as models with inhomogeneous weights or models with different distribution shapes (e.g. elliptic or generic).

5. Use of Additional Data: Improving the performance of the algorithm can be achieved by using additional data for training, if possible.

Some of the potential disadvantages of the ER-RA algorithm in-

clude implementation complexity, increased computational complexity due to adaptive weights, and the need for additional hyperparameter tuning. Also, if the algorithm is not properly tuned, convergence or overfitting problems may occur.

Advantages of the ER-RA algorithm (Adaptive EM algorithm with weights):

Advantages:
1. Adaptive Weights: Introducing adaptive weights allows the model to adjust the importance of each pattern in the parameter updating process, which can lead to better convergence and better results.

2. Robustness: Adaptive weights allow the model to adapt to different data distributions, making it more robust in different scenarios.

3. Model Expressiveness: Allows the model to capture more complex patterns in the data, making it capable of modeling diverse data distributions.

4. Avoiding Local Optima: Introducing stochasticity and adaptive weights can help avoid getting stuck in local optima, which can improve algorithm convergence.

Disadvantages:
1. Increased Complexity: Introducing adaptive weights increases the complexity of the algorithm, which may require additional time for implementation and optimization.

2. Increased Computational Complexity: Adaptive weights can increase the computational complexity of an algorithm, which may require more resources to execute.

3. Need to Tune Hyperparameters: The ER-RA algorithm requires proper tuning of hyperparameters to provide optimal results, which can be challenging and require additional testing.

4. Sensitivity to Initial Parameters: As with all EM algorithms, the ER-RA algorithm can be sensitive to initial parameters, which can affect the quality of convergence and results.

An analytical solution for Gaussian mixture parameters usually involves setting the derivatives of the log-likelihood function to zero and solving the resulting equations. This can be complex, especially for multi-component models. However, the basic idea is to iteratively update the mixture parameters until convergence is achieved.

Here are the basic steps for an analytical solution:
1. Initialization: The initial values of the parameters (weights, means and standard deviations) are set to some initial values.

2. E-step (Expectation): The responsibilities of each component of the mixture for each sample are calculated. These responsibilities provide information about how much each sample contributes to each component.

3. M-step (Maximization): Based on the calculated responsibilities, the mixture parameters are updated to maximize the log-likelihood. This includes calculating new weights, means, and standard deviations.

4. Convergence Check: It is checked whether the algorithm converges. If not, we go back to step 2.

[1.54299988e-22 3.74404675e-21 1.39964983e-19 5.13395003e-18
 1.63949470e-16 4.46617195e-15 1.03423721e-13 2.03461960e-12
 3.39990275e-11 4.82565876e-10 5.81768439e-09 5.95725490e-08
 5.18136741e-07 3.82775978e-06 2.40185620e-05 1.28012198e-04
 5.79505725e-04 2.22826133e-03 7.27740484e-03 2.01877988e-02
 4.75667443e-02 9.51962690e-02 1.61822351e-01 2.33646493e-01
 2.86538184e-01 2.98475837e-01 2.64084368e-01 1.98470707e-01
 1.26715132e-01 6.87732971e-02 3.18403807e-02 1.28340225e-02
 5.07576433e-03 3.09273611e-03 3.99175057e-03 6.90646208e-03
 1.21387928e-02 2.06193025e-02 3.36465059e-02 5.27117308e-02
 7.92778927e-02 1.14464960e-01 1.58660396e-01 2.11125169e-01
 2.69703754e-01 3.30757328e-01 3.89410324e-01 4.40129980e-01
 4.77562201e-01 4.97455735e-01 4.97455735e-01 4.77562201e-01
 4.40129980e-01 3.89410324e-01 3.30757328e-01 2.69703754e-01
 2.11125169e-01 1.58660396e-01 1.14464960e-01 7.92778921e-02
 5.27117245e-02 3.36464461e-02 2.06188234e-02 1.21355308e-02
 6.88759781e-03 3.89908989e-03 2.70614410e-03 3.70579048e-03
 8.71044947e-03 2.12980384e-02 4.58803052e-02 8.44900985e-02
 1.32319241e-01 1.76058372e-01 1.98984690e-01 1.91025744e-01
 1.55764429e-01 1.07881601e-01 6.34641899e-02 3.17111661e-02
 1.34585335e-02 4.85160350e-03 1.48550763e-03 3.86337169e-

04
 8.53414703e-05 1.60123758e-05 2.55184011e-06 3.45424552e-07
 3.97150449e-08 3.87845874e-09 3.21711068e-10 2.26661093e-11
 1.35642945e-12 6.89519807e-14 2.97791827e-15 1.09374598e-16
 3.43410042e-18 9.49942164e-20 2.73351005e-21 1.35012489e-22]

The ER-RA algorithm, which stands for "Expectation-Responsibility Radoslav", is an improved form of the EM (Expectation-Maximization) algorithm used to estimate the parameters of Gaussian mixtures. This algorithm introduces adaptive weights that are adjusted during iterations based on responsibility, allowing a better approximation of the data. By combining E-step and M-step, the algorithm iteratively adjusts the Gaussian mixture parameters until it reaches convergence. This improvement enables better modeling of data that can be complex and more difficult to adapt with the classic EM algorithm.

These results represent the values of the function obtained as a sum of Gaussian functions with different weights, means and standard deviations. Each value in this array represents the value of the function at a particular point x. These results were obtained using the gaussian_mixture_function function with the appropriate alpha, mu, and sigma parameters. Each value in this sequence is probably associated with certain input values of x, but without the input values of x, it is hard to tell exactly what those points are and how the results are distributed on the interval.

[5.20762459e-23 1.42637503e-21 5.54807248e-20 2.05013990e-18
 6.55573026e-17 1.78632769e-15 4.13686386e-14 8.13842926e-13
 1.35995837e-11 1.93026205e-10 2.32707301e-09 2.38290159e-08
 2.07254679e-07 1.53110383e-06 9.60742445e-06 5.12048779e-05
 2.31802284e-04 8.91304511e-04 2.91096186e-03 8.07511923e-03
 1.90266967e-02 3.80785045e-02 6.47289307e-02 9.34585675e-02
 1.14615187e-01 1.19390095e-01 1.05633109e-01 7.93866517e-02
 5.06820498e-02 2.74998867e-02 1.27148169e-02 5.08727866e-03
 1.93372146e-03 1.04379844e-03 1.22532337e-03 2.07759791e-03
 3.64261644e-03 6.18593448e-03 1.00939697e-02 1.58135212e-02
 2.37833680e-02 3.43394879e-02 4.75981187e-02 6.33375507e-02
 8.09111261e-02 9.92271983e-02 1.16823097e-01 1.32038994e-01
 1.43268660e-01 1.49236721e-01 1.49236721e-01 1.43268660e-01
 1.32038994e-01 1.16823097e-01 9.92271983e-02 8.09111261e-02
 6.33375507e-02 4.75981187e-02 3.43394879e-02 2.37833676e-02
 1.58135174e-02 1.00939338e-02 6.18564701e-03 3.64065923e-03
 2.06627934e-03 1.16972697e-03 8.11843229e-04 1.11173714e-03
 2.61313484e-03 6.38941151e-03 1.37640915e-02 2.53470296e-02
 3.96957724e-02 5.28175115e-02 5.96954069e-02 5.73077232e-02
 4.67293286e-02 3.23644802e-02 1.90392570e-02 9.51334982e-03
 4.03756004e-03 1.45548105e-03 4.45652288e-04 1.15901151e-04
 2.56024411e-05 4.80371273e-06 7.65552034e-07 1.03627366e-07
 1.19145135e-08 1.16353762e-09 9.65133205e-11 6.79983279e-12
 4.06928836e-13 2.06855942e-14 8.93375482e-16 3.28123794e-17
 1.03023013e-18 2.84982649e-20 8.20053014e-22 4.05037468e-23]

These results represent the function values obtained using different approaches to approximate the desired function. The first set of results represents the function values obtained using the EM algorithm for Gaussian mixtures (ER-RA algorithm), while the second set of results represents the function values obtained using the standard Gaussian mixture model. The difference in results may be due to different approaches to modeling and parameter optimization.

These results represent the function values obtained using different approaches to approximate the desired function. The first set of results represents the function values obtained using the EM algorithm for Gaussian mixtures (ER-RA algorithm), while the second set of results represents the function values obtained using the standard Gaussian mixture model. The difference in results may be due to different approaches to modeling and parameter optimization.

From these numbers we can conclude that the values of the function obtained by the ER-RA algorithm are significantly smaller than those obtained by the standard Gaussian mixed model. This suggests that the ER-RA algorithm can result in a finer and more accurate approximation of the objective function, which can be useful in situations where high modeling accuracy is required. However, more detailed analysis and evaluation is needed to confirm this claim and identify potential factors contributing to differ-

ences in results..

6. Conclusion

The research conclusion shows that the ER-RA algorithm, which uses adaptive weights in the EM procedure, can provide improvements over the standard EM algorithm for modeling functions using Gaussian mixtures. These efficiency improvements can be seen in situations where the data is more complex in structure or when it is distributed non-uniformly. However, it is important to point out that the performance of the ER-RA algorithm depends on a number of factors, including the choice of initial parameters, the number of mixture components, and algorithm iterations. Therefore, further research is needed to better understand the advantages and limitations of the ER-RA algorithm in different application scenarios.

In addition, it is important to highlight several advantages and limitations of the ER-RA algorithm:

Advantages:

➢ **Adaptive Weights:** The use of adaptive weights allows the algorithm to better adapt to the data distribution, which can result in a better approximation of the function.

➢ **Avoiding Local Optima:** Introducing stochasticity and adaptive weights can help avoid getting stuck in local optima, which improves algorithm convergence.

➢ **Robustness to Irregular Distributions:** The ER-RA algorithm can be useful in situations where the data is not uniformly distributed or when there are hidden patterns in the data.

Limitations:

➢ **Sensitivity to Initial Parameters:** Algorithm performance can be sensitive to the choice of initial parameters, which may require careful initialization.

➢ **Need to Tune Hyperparameters:** The ER-RA algorithm has several hyperparameters such as the number of mixture components and the maximum number of iterations, the optimal values of which may vary depending on the data set.

➢ **Computationally Demanding:** The algorithm can be computationally demanding, especially if a large number of iterations are required or if the data is of high dimensionality.

Ultimately, the ER-RA algorithm is a useful tool for modeling functions using Gaussian mixtures, but its effectiveness and applicability depends on various factors, including the nature of the data and algorithm settings. Additional research and experiments are needed to better understand its advantages and limitations in different application contexts. The ER-RA algorithm represents a step forward in improving the classical EM algorithm for modeling Gaussian mixtures.

The combination of adaptive weights and stochasticity provides greater flexibility and stability in the optimization process. However, further research is needed to better understand its performance on different types of data and modeling problems. This deeper analysis would enable more precise recommendations for the application of the ER-RA algorithm in practice. From the analytical solution, we can conclude that the obtained function values are very small for most of the interval, with a gradual increase towards the middle of the interval, where they reach higher values. This suggests that the function is very close to zero for most of the interval, while it has more pronounced values near the central part of the interval. This can be useful for understanding the distribution of a function and its behavior on a given interval, and can provide insight into the shape of the function being approximated.

From the obtained values, we can conclude that the function has a more pronounced peak in the central part of the interval, while the values of the function gradually decrease towards the edges of the interval. This suggests that the function is concentrated around a certain mean value, decreasing towards the edges of the interval. Also, small values of the function outside the central part indicate that the function rapidly decays towards zero as it moves away from the center of the interval. These conclusions provide insight into the shape and behavior of the function and can be useful for further analysis and interpretation of the results.

References

1. Bishop, C. M. (2006). Pattern recognition and machine learning. *Springer google schola, 2,* 645-678.
2. Cajic, E., Ibrišimović, I., Šehanović, A., Šćekić, J., & Bajrić, D. (2023). Neuro-Fuzzy Disease Detection Using Interpolation in Matlab: Unveiling the Hidden Patterns. *Available at SSRN 4673502.*
3. Čajić, E. PRACTICAL EXAMPLES OF SIGNAL AND SYSTEM MODELING AND SIMULATION.
4. Stosovic, D., & Čajić, E. (2024). Optimization of Numerical Solutions of Stochastic Differential Equations with Time Delay.
5. Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society: series B (methodological), 39*(1), 1-22.
6. Čajić, E., Stojanović, Z., & Galić, D. (2023, November). Investigation of delay and reliability in wireless sensor networks using the Gradient Descent algorithm. In *2023 31st Telecommunications Forum (TELFOR)* (pp. 1-4). IEEE.
7. Cajic, E., Rešić, S., & Elezaj, M. R. (2024). Development of efficient models of artificial intelligence for autonomous decision making in dynamic information systems. *Available at SSRN 4746431.*
8. Galić, R., & Čajić, E. (2023). Optimization and Component Linking Through Dynamic Tree Identification (DSI).
9. Galić, D., Stojanović, Z., & Čajić, E. (2024). Application of Neural Networks and Machine Learning in Image Recogni-

tion. *Tehnički vjesnik, 31*(1), 316-323.
10. Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (1995). *Bayesian data analysis*. Chapman and Hall/CRC.
11. Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2, pp. 1-758). New York: springer.
12. Ibrišimović, I., Jasak, Z., Omerović, A., & Čajić, E. (2023). Practical Application of Out-of-Kilter Algorithm. *Chinese Business Review, 22*(2), 86-94.
13. Čajić, E. (2023).Optimization of nonlinear equations with constraints using genetic algorithm.
14. Murphy, K. P. (2012). *Machine learning: a probabilistic perspective.* MIT press.
15. Galić, R., Čajić, E., Stojanovic, Z., & Galić, D. (2023). Stochastic Methods in Artificial Intelligence.
16. Rešić, S., Čajić, E., & Hrnjičić, A. (2018). MATHEMATICAL MODEL ON TRANSPORT NETWORKS. *Nauka i tehnologija, 6*(11), 68-72.
17. Ramaj, V., Elezaj, R., & Čajić, E. (2024). Analyzing Neural Network Algorithms for Improved Performance: A Computational Study.
18. Ramaj, V., Elezaj, R., & Čajić, E. (2024). Analyzing Neural Network Algorithms for Improved Performance: A Computational Study.
19. Ramaj, V., Elezaj, R., & Čajić, E. (2024). Fuzzy Numbers Unraveling the Intricacies of Neural Network Functionality.
20. Stojanović, Z., & Čajić, E. (2019, November). Application of Telegraph Equation Soluton Telecommunication Signal Trasmission and Visualization in Matlab. In *2019 27th Telecommunications Forum (TELFOR)* (pp. 1-4). IEEE.
21. Zenunovic, I., & Cajic, E. Diferencijalna geometrija površi primjenom Wolfram Mathematica.