Jewel Journal of Scientific Research (JJSR) 9(1&2): 107–118, 2024
©Federal University of Kashere-Nigeria (Print ISSN: 2384 – 6267, Online ISSN 2756-651X)
https://journals.fukashere.edu.ng/index.php/jjsr

Performance of some selected distance measures based on EM algorithm with split and merge

[1,*]Ahmed, H., [2]Baba I. A., [1]Muhammad S. and [1]Omar, A. H.
[1]Department of Statistics, Modibbo Adama University, Yola, Adamawa State, Nigeria.
[2]Department of Statistics, Abubakar Tafawa Balewa University, Bauchi, Bauchi State, Nigeria.
[*]**Corresponding Author**: hassanahmed.official@gmail.com, +2348032417537

Abstract
Finite mixture models have become increasingly prominent in statistical data analysis, reflected by a growing body of literature addressing their theoretical and practical aspects. This rise in interest is driven by the adoption of finite mixtures of distributions as computationally efficient tools for modeling complex data distributions from random phenomena. This paper aims to compare various statistical distances for the EM algorithm with split and merge, using both simulated and real data sets. The distances are: Kullback-Leibler Distance, Hellinger Distance and Total Variation Distance. Two types of data were used in this study: simulated data and real data. The simulated data was generated from a bivariate normal distribution, while the real data set consisted of information on diabetic patients. The results indicate that there is no significant difference in parameter estimates among the three distances tested. However, for both synthetic and real data sets, the Total variation distance proved to be the most efficient, as it reached the optimal solution quickest with minimal computational load.

Keywords: mixture-model, EM, algorithm, split and merge, GMM

Received: 12th May, 2024 *Accepted:* 14th June, 2024 *Published Online:* 26th Dec, 2024

Introduction

Finite mixture models have gained increasing prominence in statistical data analysis, as evidenced by a rising number of articles addressing their theoretical and practical aspects in scientific literature. This surge in interest is due to the widespread adoption of finite mixtures of distributions as computationally efficient tools for modeling complex data distributions arising from random phenomena. These models have been successfully applied across various fields, including agriculture, astronomy, bioinformatics, biology, economics, engineering, genetics, imaging, marketing, medicine, neuroscience, psychiatry, psychology, and many other disciplines within the biological, physical, and social sciences (McLachlan *et al*, 2019).

Finite mixture models and distributions are statistical tools used to understand data arising from multiple groups. Imagine a population with two height distributions, one for short and one for tall people. A finite mixture model would account for this by combining two simpler distributions (like normal distributions for each height group) and estimating how likely each data point comes from each group. This allows us to analyze the data while acknowledging the underlying hidden groups, even though we can't directly observe them.

Finite mixture models are often used to analyze data from populations believed to consist of homogeneous subpopulations. For

example, when a disease has a simple genetic cause, the population may be divided into two or three homogeneous groups. In the early stages of such studies, having a sensitive test to determine the number of subpopulations (denoted as k) is crucial. However, constructing such a test is often challenging because finite mixture models belong to a class of non-regular models, rendering many classical asymptotic results inapplicable.

Finite mixtures of distributions, especially normal distributions, have been extensively utilized as models in practical situations where data originate from two or more populations mixed in varying proportions (Kayal *et al*., 2023). Recent applications include a spatially constrained skew Student's-t mixture model for brain MR image segmentation and bias field correction (Cheng *et al*., 2022). Normal mixture models are also important in developing robust estimators to address challenges such as sensitivity to outliers in mixtures with light-tailed distributions like the normal distribution (Sugasawa *et al*., 2022).

Cluster analysis, particularly through the mixture likelihood approach, has seen increasing use of mixtures of distributions, normal or otherwise. This approach assumes that observations are from a mixture of several populations or groups in varying proportions. By adopting a parametric form for the density function in each group, a likelihood is formed, and unknown parameters are estimated through this likelihood. Probabilistic clustering is achieved based on estimated posterior probabilities of group membership, allowing entities to be assigned to groups with the highest estimated posterior probability (Ganesalingam and McLachlan, 1979).

While decomposing a finite mixture of distributions is a challenging problem, the advent of high-speed computers has shifted focus to the likelihood estimation of parameters in mixture distributions. The Expectation Maximization (EM) algorithm has become a widely used method for iterative computation of maximum likelihood estimates in incomplete data problems, where traditional methods may be more complicated (Dempster *et al.*, 1977). The EM algorithm, with its Expectation step (Estep) and Maximization step (M-step), has proven valuable in various problems involving incomplete data, offering an intuitive approach to estimation even before its formalization in the seminal DLR paper (Dempster *et al*., 1977).

The applications of the Expectation–Maximization (EM) algorithm range from theoretical studies on convergence, such as the analysis of EM and its variant DA-EM (Yu *et al*., 2018), to diverse modifications tailored for specific purposes, including image matching (Ma, Jiang, Jiang, & Gao, 2019), parameter estimation (Liu et al., 2019; Du & Gui, 2019), malaria diagnosis (Pag`es-Zamora *et al*., 2019), mixture simplification (Yu *et al*., 2018), and audio-visual scene analysis (Gebru, Alameda-Pineda, Forbes, & Horaud, 2016).

The popularity of the EM algorithm is largely due to its effectiveness in estimating parameters of mixture models (MM) (McLachlan, 2000; McLachlan & Krishnan, 2007). Mixture models are probabilistic models where the population consists of several sub-populations, each assumed to follow a simple parametric distribution. Gaussian Mixture Models (GMM) are a notable example where components follow a Gaussian distribution. Once estimated, MM parameters are applied in density estimation (Yu *et al*., 2018; B¨acklin *et al*., 2018) and clustering tasks (Pag`es-Zamora *et al*., 2019; Yang et al., 2012; Celeux & Govaert, 1995). In the context of MM parameter estimation, the EM algorithm serves as a clustering algorithm that maximizes the missing data log-likelihood function, acting as a maximum-likelihood estimator. EM's properties, such as monotonic convergence and probabilistic constraints, make it particularly appealing for MM parameter estimation.

The Expectation-Maximization (EM) algorithm is a powerful statistical technique for analyzing data containing hidden variables. These hidden variables influence the data we observe, but we can't directly

measure them. Imagine a box of fruits with unknown types (hidden variable) but labeled colors (observed data). EM tackles this by making educated guesses about the hidden variables (expectation step) and then using those guesses to improve our understanding of the overall data (maximization step). It iterates between these steps until it converges on a solution.

EM is particularly useful for problems with missing data or latent variables. For instance, it can be used to group customers into different segments based on their purchase history (even if some purchases are missing) or to identify hidden topics in a large collection of documents. It's also valuable for fitting complex models like Gaussian Mixture Models, which are used for data clustering, or Hidden Markov Models, used for modeling sequential data. Overall, EM is a versatile tool whenever you have hidden variables or missing data that cloud the true picture your data reveals.

In their paper (Zhang *et al.*, 2004), a novel Competitive EM (CEM) algorithm designed for finite mixture models was introduced to address the primary limitations of the EM algorithm, including susceptibility to local maxima and occasional convergence to the parameter space boundary. The CEM algorithm autonomously determines the number of clusters and efficiently executes "split" or "merge" operations using a new competitive mechanism. It demonstrates insensitivity to the initial configuration of the mixture component number and model parameters.

The aim of this paper is to compare various statistical distances for the EM algorithm with split and merge, based on simulated and real data sets.

Methods

For convenience, we provide some necessary and useful definitions and some mathematical derivations on finite mixture model, EM algorithm and the Competitive EM algorithm which we need throughout the research work.

Learning finite mixture models

It is said a d-dimensional random variable $x = [x_1, x_2, \ldots, x_d]^T$ follows a k-component finite mixture distribution, if its probability density function can be written as

$$p(x/\theta) = \sum_{m=1}^{\theta} \pi_m \, p(x/\theta_m) \quad (1)$$

where π_m is the prior probability of the mth component and satisfies

$$\pi_m \geq 0, \sum_{m=1}^{k} \pi_m = 1 \quad (2)$$

Where θ_m is the parameter of the mth density model and

$$\theta = \{(\pi_m, \theta_m), m = 1, \ldots, k\} \quad (3)$$

is the parameter of the mixture models. For our study we used a 4component 2-dimensional Gaussian mixture model.

Formulation and Derivations of the EM algorithm.

To estimate the parameters of the finite mixture model defined above, we employ the EM algorithm. Suppose we have a random variable *X* from a D-dimensional Gaussian mixture model, i.e.,

$\underline{X} \sim N\left(\underline{X}_j; \underline{\mu}_z, \underline{\Sigma}'_z\right)$

Then,

$p(Z, \underline{X}) = p(Z)p(\underline{X}|Z) = cat(Z: \prod 1)N\left(\underline{X}; \underline{\mu}_z, \underline{\Sigma}'_z\right)$

$= \prod_{d=0}^{D-1} \prod_d^{I(z=d)} \frac{1}{\sqrt{(2\Pi)^k det(\Sigma'_z)}} \exp\left\{\{-\frac{1}{2}\left(\underline{X} - \underline{\mu}_z\right)^{T\underline{\Sigma}'^{-1}_z} \left(\underline{X} - \underline{\mu}_z\right)\right\}$

E-Step

We apply Baye's rule since the posterior is yet unknown

$p(Z, \underline{X}) = \frac{p(Z)p(\underline{X}|Z)}{p(\underline{X})} \sim p(Z)p(\underline{X}|Z) = p(Z, \underline{X})$ (4)

The unnormalized responsibilities is given by,

$$\widetilde{\rho_d^{[i]}} = p\left(Z = d, \underline{X} = \underline{X}^{[i]}; \underline{\theta}^{[K]}\right) \quad (5)$$

$$= \prod_d \frac{1}{\sqrt{(2\Pi)^k det(\Sigma'_z)}} \exp\left\{-\frac{1}{2}\left(\underline{X}^{[i]} - \underline{\mu}_d\right)^T \underline{\Sigma}'^{-1}_z \left(\underline{X}^{[i]} - \underline{\mu}_d\right)^T\right\} \quad (6)$$

Therefore, normalised responsibility is given by,

$$\rho_d^{[i]} = \frac{\widetilde{\rho_d^{[i]}}}{\sum_{c=o}^{D-1} \widetilde{\rho_c^{[i]}}} \quad (7)$$

M-Step

$Q\left(\underline{\theta}; D; \underline{\theta}^{[k]}\right)$

$$= \sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \rho_c^{[i]} \log p\left(Z = d, \underline{X} = \underline{X}^{[i]}; \underline{\theta}\right) \quad (8)$$

$$= \sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \rho_c^{[i]} \left(\sum_{d=0}^{D-1} I(d = c) log \underline{\prod\prod}_d - \frac{K}{2} \log 2\Pi - \frac{1}{2} \log \det\left(\underline{\Sigma}'_d\right) - \frac{1}{2}\left(\underline{X}^{[i]} - \underline{\mu}_d\right)^T \underline{\Sigma}'^{-1}_z \left(\underline{X}^{[i]} - \underline{\mu}_d\right) \right)$$

To derive the unconstrained optimization, we build a Lagrangian, that is,

$\hat{Q}\left(\underline{\theta}, \lambda\right) = Q\left(\underline{\theta}\right) + \lambda(\lambda - \sum_{c=o}^{D-1} \pi_c)$ (12)

Maximising with respect to $\boldsymbol{\pi}$

$$\frac{\partial \widehat{\boldsymbol{Q}}}{\partial \widehat{\boldsymbol{\Pi}_e}} = \left(\sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \boldsymbol{\rho_c^{[i]}} \left(\frac{\partial}{\partial \widehat{\boldsymbol{\Pi}_e}} \sum_{d=0}^{D-1} \mathrm{I(d=c)} \log \boldsymbol{\Pi_d} \right) \right) - \boldsymbol{\lambda} \frac{\partial}{\partial \widehat{\boldsymbol{\Pi}_e}} \sum_{c=0}^{D-1} \boldsymbol{\Pi_c}$$

$$= \left(\sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \boldsymbol{\rho_c^{[i]}} \sum_{d=0}^{D-1} \mathrm{I(d=c)} \frac{\mathbf{1}}{\boldsymbol{\Pi_d}} \boldsymbol{\partial_{de}} \right) - \boldsymbol{\lambda} \sum_{c=0}^{D-1} \boldsymbol{\partial_{ce}}$$

$$= \left(\sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \boldsymbol{\rho_c^{[i]}} \boldsymbol{I(e = c)} \frac{\mathbf{1}}{\boldsymbol{\pi_e}} \right) - \boldsymbol{\lambda}$$

$$= \left(\sum_{i=0}^{N-1} \rho_e^{[i]} \frac{1}{\pi_e}\right) - \lambda$$
$$= \frac{1}{\pi_e} \sum_{i=0}^{N-1} \rho_e^{[i]} = \lambda$$
$$= \frac{1}{\pi_e} \gamma_e = \lambda$$
$$\therefore \widehat{\pi_e} = \frac{\gamma_e}{\lambda} \quad (9)$$

NOTE:

1. $\frac{\partial}{\partial \widehat{\pi_e}} \sum_{d=0}^{D-1} \mathrm{I(d=c)} \log \pi_d$
$= \frac{\partial}{\partial \widehat{\pi_e}} \mathrm{I(d=c)} \log \pi_d = \frac{\partial}{\partial \widehat{\pi_e}} \log \Pi_d$
$= \frac{\partial \log \pi_d}{\partial \pi_d} \frac{\partial \pi_d}{\partial \pi_e}$
$= \frac{1}{\pi_d} \partial_{de}$ (10)

2. $\frac{\partial}{\partial \widehat{\pi_e}} \pi_c = \partial_{ce}$ (11)

<u>Maximising with respect to λ</u>

$$\frac{\partial \hat{Q}}{\partial \lambda} = \lambda - \sum_{c=0}^{D-1} \pi_c = 0$$
$$\sum_{c=0}^{D-1} \frac{\gamma_c}{\lambda} = 1$$
$$\lambda = \sum_{c=0}^{D-1} \gamma_c = N \quad (16)$$

Maximising with respect to μ

$$\frac{\partial \hat{Q}}{\partial \underline{\mu}_e} = \sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \rho_c^{[i]} \left(\frac{\partial}{\partial \mu_e} \left(-\frac{1}{2} \left(\underline{X}^{[i]} - \underline{\mu}_c \right)^{T \underline{\Sigma}_c'} \left(\underline{X}^{[i]} - \underline{\mu}_c \right) \right) \right) = \underline{0}$$
$$= \sum_{i=0}^{N-1} \sum_{c=0}^{D-1} \rho_c^{[i]} \left(-\underline{\Sigma}_c' \left(\underline{X}^{[i]} - \underline{\mu}_c \right) \right) \partial_{ce} = \underline{0}$$
$$= \sum_{i=0}^{N-1} \rho_e^{[i] \underline{\Sigma}_e'} \left(\underline{X}^{[i]} - \underline{\mu}_e \right) = \underline{0}$$
$$= \sum_{i=0}^{N-1} \rho_e^{[i]} \left(\underline{X}^{[i]} - \underline{\mu}_e \right) = \underline{0}$$
$$= \sum_{i=0}^{N-1} \rho_e^{[i]} = \gamma_e \underline{\mu}_e$$
$$\therefore \underline{\mu}_e = \frac{\left(\sum_{i=0}^{N-1} \rho_e^{[i]} \underline{X}^{[i]} \right)}{\gamma_e} \quad (12)$$

Maximising with respect to $\underline{\Sigma}_e'$

$$\frac{\partial \hat{Q}}{\partial \underline{\Sigma}_e'}$$

$$= \sum_{i=0}^{N-1}\sum_{c=0}^{D-1} \rho_c^{[i]} \left(\frac{\partial}{\partial \underline{\Sigma}_e'} \left(\frac{-1}{2} \log \underline{\Sigma}_c \right) + \frac{\partial}{\partial \underline{\Sigma}_e'} \left(-\frac{1}{2} \left(\underline{X}^{[i]} - \underline{\mu}_c \right)^{\mathrm{T}} \underline{\Sigma}_c' \left(\underline{X}^{[i]} - \underline{\mu}_c \right) \right) \right)$$

$$= \sum_{i=0}^{N-1}\sum_{c=0}^{D-1} \rho_c^{[i]} \left(\frac{-1}{2} \underline{\Sigma}_c' \partial_{ce} + \left(\frac{-1}{2} \right) \underline{\Sigma}_c' \left(\underline{X}^{[i]} - \underline{\mu}_c \right) \left(\underline{X}^{[i]} - \underline{\mu}_c \right)^T \partial_{ce} \right) = \underline{0}$$

$$= \sum_{i=0}^{N-1} \rho_e^{[i]} \left(\underline{\Sigma}_e' \underline{\Sigma}_c' \left(\underline{X}^{[i]} - \underline{\mu}_e \right) \left(\underline{X}^{[i]} - \underline{\mu}_c \right)^T \underline{\Sigma}_e' \right)$$

$$= \left(\sum_{i=0}^{N-1} \rho_e^{[i]} \right) \underline{\Sigma}_e' - \sum_{i=0}^{N-1} \underline{\Sigma}_e' \rho_e^{[i]} \left(\underline{X}^{[i]} - \underline{\mu}_e \right) \left(\underline{X}^{[i]} - \underline{\mu}_e \right)^T \underline{\Sigma}_e' = \underline{0}$$

$$= \underline{\Sigma}_e' \gamma_e - \sum_{i=0}^{N-1} \rho_e^{[i]} \left(\underline{X}^{[i]} - \underline{\mu}_e \right) \left(\underline{X}^{[i]} - \underline{\mu}_e \right)^T = \underline{0}$$

$$\therefore \underline{\Sigma}_e' = \frac{1}{\gamma_e} \sum_{i=0}^{N-1} \rho_e^{[i]} \left(\underline{X}^{[i]} - \underline{\mu}_e \right) \left(\underline{X}^{[i]} - \underline{\mu}_e \right)^T \quad (13)$$

The E and M step is performed iteratively until convergence.

The Modified Competitive EM(CEM)

When EM encounters local maxima, the components usually overpopulate in some regions, i.e. the model overfits the data, but under-populate in other regions. The difficulty of passing through some low likelihood regions prevents them from getting to the expected positions. To overcome this problem, CEM do the split or merge operation when EM has converged to a local maximum, thus the components' distribution can self-adapt, and split will take place where the components are too few and merge will take place where the components are too many.

Statistical Distances

CEM use local Kullback divergence to measure the distance between the local data density $f_m(x)$ and model density $p_m(x)$ of the *mth* component. Define

$$j(m;\theta) = \int f_m(x) \frac{f_m(x)}{p_m(x)} \, dx \quad (14)$$

Split probability of the *mth* component is in direct proportion to $j(m;\theta)$ and the merge probability of the *mth* component is in inverse proportion $j(m';\theta')$, where m and θ denote, respectively, the new index of merged component and new parameters of mixture models if the *mth* and *lth* components merge. CEM will also use Hellinger distance to measure the distance as in Equation 43. The squared Hellinger is given by:

$$j(m;\theta) = 1 - \frac{det(\Sigma_1)^{1/4} det(\Sigma_2)^{1/4}}{det\left(\frac{\Sigma_1 + \Sigma_2}{2} \right)^{1/2}} \quad (15)$$

Also, CEM will also use total variation distance between the distributions and this is defined by:

$$j(m;\theta) = \sup_{f_m(x) \in I} |f_m(x) - p_m(x)| \quad (16)$$

Therefore,

$$p_{split}(m;\theta) = \frac{j(m;\theta)}{Z(\theta)} \quad (17)$$

$$p_{merge}(m,l;\theta) = \frac{\beta/j(m';\theta')}{Z(\theta)} \quad (18)$$

Where

$$Z(\theta) = \sum_{m=1}^{k} p_{split}(m;\theta) + \sum_{m=1}^{k}\sum_{l=m+1}^{k} p_{merge}(m,l;\theta) = 1 \quad (19)$$

Where β is a constant.

Operations of Split and Merge

The operation type and the candidates of split or merge components are sampled by $p_{split}(m;\theta)$ and $p_{merge}(m,l;\theta)$

Split Operation: Suppose that the *mth* component is chosen to split, CEM will generate two new components from the current samples in the *mth* component. We initialize the two new components parameters by random initialization method and optimize them by the basic EM algorithm.

Merge operation: Suppose that the *mth* and *lth* components are selected to merge, the parameters of the merged component can be calculated directly from the original *mth* and *lth* components parameters as follows:

$$\pi_m^{(M)} = \pi_m + \pi_l \quad (20)$$

$$\mu^{(M)} = (\pi_m\mu_m + \pi_l\mu_l)/\pi^{(M)} \quad (21)$$

$$\Sigma_m^{(M)} = \frac{\pi_m\left[\Sigma_m+\left(\mu_m-\mu_m^{(M)}\right)\left(\mu_m-\mu_m^{(M)}\right)^T\right]+\pi_l\left[\Sigma_l+\left(\mu_l-\mu_m^{(M)}\right)\left(\mu_l-\mu_m^{(M)}\right)^T\right]}{\pi_m^{(M)}} \quad (22)$$

Probability of Acceptance

After the operation type and operation candidates are chosen by sampling according to the split and merge probabilities, the acceptance probability is calculated to prevent poor operation. It is calculated by:

$$P_a = min\left(\exp\left(\frac{L(\theta(t+1),\boldsymbol{X})-L(\theta(t),\boldsymbol{X})}{\gamma}\right), 1\right) \quad (23)$$

where γ is a constant.

Results and Discussion

In this section, we would use synthetic and real data to run the modified CEM three (3) times. For each run, we would change the method of calculating the distances in split and merge. This is ideally equations 14, 15 and 16. Then results are recorded and shown for comparison.

Synthetic data

We use sample of size 1000 from 4-component GMM. In this GMM, the two components (kernels 1 and 2) share a common mean, but have different covariance matrices. The prior probability of kernel 4 is a little lower than the other kernels. The parameters of GMM are given as follows:

$$\pi_1 = \pi_2 = \pi_3 = 0.3,\ \pi_4 = 0.1, \quad (24)$$

$$\mu_1 = \mu_2 = [-4,-4]^T \quad (25)$$

$$\boldsymbol{\mu_3} = [\mathbf{2},\mathbf{2}]^{\boldsymbol{T}},\ \boldsymbol{\mu_4} = [-\mathbf{1},-\mathbf{6}]^{\boldsymbol{T}} \quad (26)$$

Software version and Implementation in R

The models are run in R using R-studio. The used versions for this work are R version 4.3.2 (2023-10-31), Platform: x86 64-apple-darwin23.0.0 (64bit), R-studio version: RStudio/2023.09.1+494 For MAC Operating system 14.2.1.

The package used are mixtools (Benaglia *et al*, 2009) version: 2.0.0, mclust (Scrucca *et al.*, 2023) version 6.0.1 and gratis (Kang *et al.*, 2020) version 1.0.5

Figure 1 clearly shows that in the simulated data there is considerable heterogeneity in both components. Note that these data are model-driven and merely used for illustration of the software functionality. Figure 2 shows a histogram of the generated data where we can see presence of mixture.

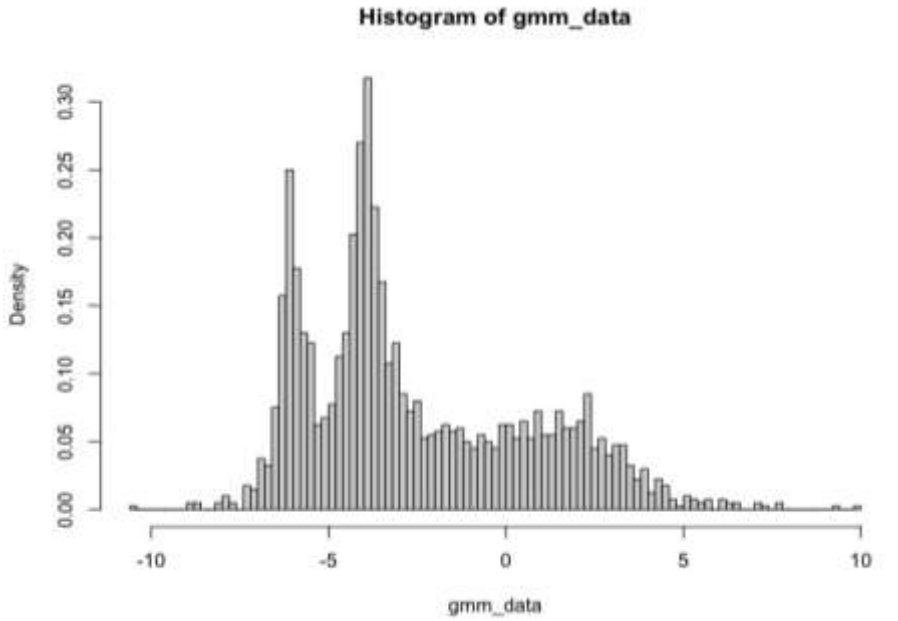

Figure 1: Bivariate two-Component Gaussian Mixture Model

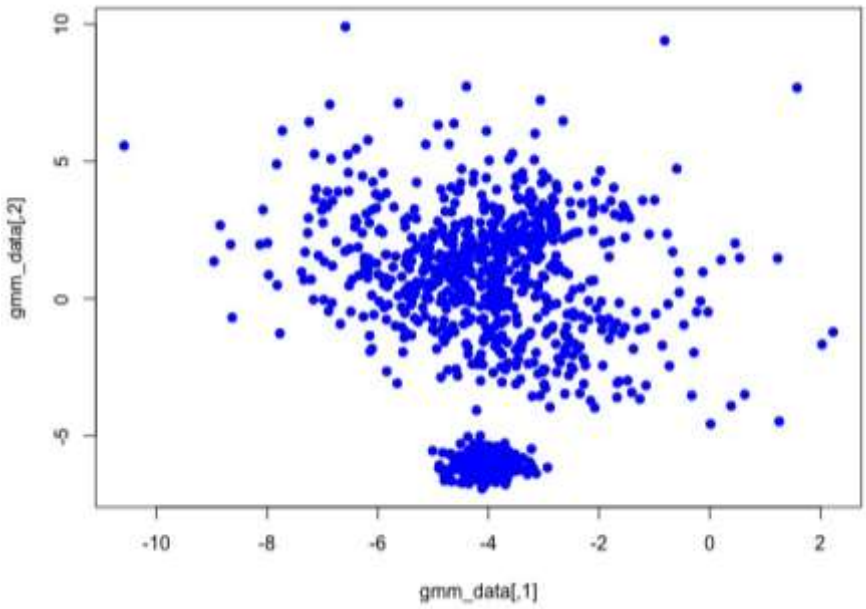

Figure 2: Plot of simulated data showing clustering

The table below shows the results and estimates of the fitted 4 - component mixture model Using Split and Merge.

Table 1: Estimates of selected 4 component Gaussian mixture model for simulated data

Distance		**Component 1**	**Component 2**	**component 3**	**component 4**
KL	Mixing Probability:	0.3027889	0.3343421	0.1917989	0.1710701
	Means:				
	Variable 1	-4.016259	-4.209846	-3.736834	-3.839645
	Variable 2	-6.012891	-0.332322	2.135295	2.320582
	Variances:				
		0.117010 -0.003298 -0.0032980 0.113375020	2.570516 -1.627323 -1.627323 3.056676	0.6657319 0.4300285 0.4300285 1.0806868	5.339230 -1.673955 -1.673955 6.592091
Hellinger	Mixing Probability:	0.298787	0.3567	0.234543	0.19878
	Means:				
	Variable 1	-4.89878	-4.454545	-3.65434	-3.76567
	Variable 2	-6.898767	-0.234344	3.122322	2.434444
	Variances:				
		0.12323 -0.002323 -0.0012323 0.2909898	2.6767 -1.98767 -1.50988 3.87877	0.778877 0.44333 0.411222 1.099887	5.222111 -1.98988 -1.68999 6.698877
Total Variation	Mixing Probability:	0.22099	0.112233	0.188999	0.1988778
	Means:				
	Variable 1	-4.122233	-4.30999	-3.88988	-3.78787
	Variable 2	-6.876567	-0.3243456	2.76548	3.0999
	Variances:				
		0.11233 -0.012343 -0.0098987 0.212321	2.793871 -1.512987 -1.76012 3.09812	0.223098 0.98120 0.12098 1.2312909	5.1123098 -1.5654098 -1.120902 6.3239890

We observe from Table 1 above that estimates of parameters from the various distances are not significantly different. It also worth noting that the number of iterations when running the algorithm using total variation is the minimum. This means that the computational load is lowest for Total variation distance.

Real Data

The data used is due to (Ahmed, Mohammed, & Baba, 2021). The data set consists of 553 cases of diabetes mellitus that were collected at Federal Medical Center, Yola. The variables measured: Age(years), Mass of a patient(kg/meters), glucose level (plasma glucose concentration, a 2-hour in an oral glucose tolerance test), pressure (Diastolic blood pressure mmHg), insulin (2-hour serum insulin mu U/ml) and class variable (0 or 1) treating 0 as false or negative and 1 treated as true or positive test for diabetes.

Figure 3 below show the histogram of the data.

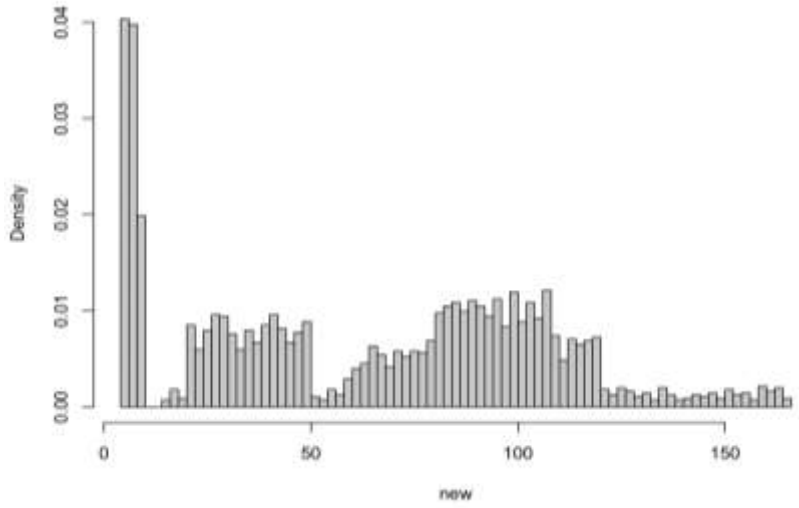

Figure 3: Real Data

The updated algorithm is put through real-data sets. The results obtained for estimates of parameters of the selected model is shown below on Table 5. The code shown above is slightly modified to have the results in this section.

Table 2: Estimates of selected 3 component Gaussian mixture model for real data.

Distance	**Component**	**1**	**2**	**3**	**4**
	π:	0.1234	0.2213	0.44355	0.443322
KL	Glucose	5.112	4.8876	5.8876	7.7766
	Pressure	201.33	66.87	32.144	5.3343
	Insulin	8.4434	50.8877	170.9988	320.778
	Age	45.3243	43.7788	29.3322	454.333
	Weight	88.7876	76.6655	90.4433	32.1223
	BIC :	-6192.06			
	Loglikelihood:	-3214.72			
Hellinger	π:	0.22321	0.21211	0.48912	0.3988
	μ: glucose				
		5.887112	4.33211	5.2234009	7.12229
	Pressure	202.442	66.11322	31.99897	6.000
	Insulin	9.112	51.223	171.9988	321.378
	Age	46.3213	44.7328	28.3244	455.221
	Weight	89.898	75.7665	91.6566	31.3432
	BIC :	-6192.06			
	Loglikelihood:	-21099.62			
Total Variation	π:	0.3432	0.2234	0.4322	0.56445
	μ: glucose				
		4.99899	4.1122	5.9987	7.7766
	Pressure	201.31	66.27	30.144	6.3343

Insulin	7.4234	48.343	172.7873	321.223
Age	46.8987	42.4434	28.7345	452.3343
Weight	89.6765	75.3421	91.4343	31.2133
BIC :	-6500.56			
Loglikelihood:	-23145.55			

It can be observed that the algorithm has chosen 4-component model with a loglikelihood of -3214.72 and BIC of -6192.06.

Table 3: Performance of the algorithm

Distance	**No of Splits**	**No. of Merge**	**Number of Iterations**	***Tavg***	P_s **(%)**
KL	75	22	80	94.6	95.1
Hellinger	70	29	85	96.6	94.1
Total Variation	50	18	56	90.6	98.1

The following are observed and recorded: The number of times the program performed split operations, number of times it performs merge operations, total number of iterations before convergence, average number of iterations carried out before the first split or merge, and percentage of optimal iterations.

With the real data too, from Table 2, it is worth noting that estimates of parameters from all the distances are not significantly different.

Table 3 shows performance of the algorithm for each of the distances. We can observe, just like in the case of the synthetic data, that Total variation distance has the minimum number of iterations and it reaches convergence faster with just 56 iterations on average. This further shows that total variation estimates the parameters quickly like the other distances but with minimum computational load.

From Table 1 we can observe that the KL distance has the smallest BIC. It can also be observed that the Total variation distance has the highest BIC. This indicates that for KL, there is a better alignment between the data and the resulting clusters. That means there is a better fit when KL divergence is used.

This results indicate that KL divergence fit the data better but with a heavier computational load. While Total variation has the lightest computational load but with a poorer fit.

Conclusion

The aim of this work is to compare 3 statistical distances namely: Kullback-Leibler Distance, Hellinger Distance and Total Variation Distance as applied in EM algorithm with split and merge. 2 types of data were used. This include simulated data and real data. The simulated data was generated from a bivariate normal distribution and the real data set is on diabetic patients. Results show that, in the estimate of parameters, there is no significant difference among the 3 distances. However, for both synthetic and real data sets, the Total variation distance shows that it is the most efficient since it reaches optimal solution quickest with minimal computational load.

References

Ahmed, H., Mohammed, M. B. and Baba, I. A. (2021). On Comparing Multi-Layer Perceptron and Logistic Regression for Classification of Diabetic Patients in Federal Medical Center, Yola, Adamawa State.

Ba¨cklin, C. L., Andersson, C., and Gustafsson, M. G. (2018). Self-tuning density estimation based on Bayesian averaging of adaptive kernel density estimations yields state-of-the-art performance. *Pattern Recognition*, 78: 133–143.

Benaglia, T., Chauveau, D., Hunter, D. R., and Young, D. (2009). Mixtools: An R Package for Analyzing Finite Mixture Models. *Journal of Statistical Software*, 32(6): 1–29. Retrieved from https://www.jstatsoft.org/v32/i06/

Celeux, G., and Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern recognition*, 28(5): 781–793.

Cheng, K. K., Lam, T. H., and Leung, C. C. (2022). Wearing face masks in the community during the COVID-19 pandemic: altruism and solidarity. *The Lancet*, 399(10336): e39–e40.

Dempster, A. P., Laird, N. M. and 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.

Du, Y. and Gui, W. (2019). Goodness of fit tests for the log-logistic distribution based on cumulative entropy under progressive type II censoring. *Mathematics*, 7(4): 361.

Ganesalingam, S., and McLachlan, G. J. (1979). Small sample results for a linear discriminant function estimated from a mixture of normal populations. *Journal of Statistical Computation and Simulation*, 9(2): 151–158.

Gebru, I. D., Alameda-Pineda, X., Forbes, F., and Horaud, R. (2016). EM algorithms for weighted-data clustering with application to audio-visual scene analysis. *IEEE transactions on pattern analysis and machine intelligence*, 38(12): 2402–2415.

Kang, Y., Hyndman, R. J. and Li, F. (2020). GRATIS: GeneRAting TIme Series with diverse and controllable characteristics. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 13(4): 354–376.

Kayal, S., Bhakta, R. and Balakrishnan, N. (2023). Some results on stochastic comparisons of two finite mixture models with general components. *Stochastic Models*, 39(2): 363–382.

Liu, C., Li, H.-C., Fu, K., Zhang, F., Datcu, M. and Emery, W. J. (2019). Bayesian estimation of generalized gamma mixture model based on variational em algorithm. *Pattern Recognition*, 87: 269– 284.

Ma, J., Jiang, X., Jiang, J. and Gao, Y. (2019). Feature-guided Gaussian mixture model for image matching. *Pattern Recognition*, 92: 231–245.

McLachlan, G. (2000). Peel. d. *Finite Mixture Models*.

McLachlan, G., Lee, S. X. and Rathnayake, S. I. (2019). Finite mixture models. *Annual review of statistics and its application*, 6: 355– 378.

McLachlan, G. J. and Krishnan, T. (2007). *The EM algorithm and extensions*. John Wiley & Sons.

Pag`es-Zamora, A., Cabrera-Bean, M. and Diaz-Vilor, C. (2019). Unsupervised online clustering and detection algorithms using crowdsourced data for malaria diagnosis. *Pattern Recognition*, 86: 209–223.

Scrucca, L., Fraley, C., Murphy, T. B. and Raftery, A. E. (2023). *Model-Based Clustering, Classification, and Density Estimation Using mclust in R*. Chapman and Hall/CRC. Retrieved from https://mclust-org.github.io/book/ doi: 10.1201/9781003277965

Sugasawa, S., Kim, J. K., and Morikawa, K. (2022). Semiparametric imputation using latent sparse conditional Gaussian mixtures for multivariate mixed outcomes. *arXiv preprint arXiv:2208.07535*.

Yang, M.-S., Lai, C.-Y. and Lin, C.-Y. (2012). A robust EM clustering algorithm for Gaussian mixture models. *Pattern Recognition*, 45(11): 3950–3961.

Yu, J., Chaomurilige, C., and Yang, M.-S. (2018). On convergence and parameter selection of the EM and DA-EM algorithms for Gaussian mixtures. *Pattern Recognition*, 77: 188–203.

Yu, L., Yang, T. and Chan, A. B. (2018). Density-preserving hierarchical EM algorithm: Simplifying Gaussian mixture models for approximate inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(6): 1323–1337.

Zhang, B., Zhang, C. and Yi, X. (2004). Competitive EM algorithm for finite mixture models. *Pattern recognition*, *37*(1): 131–144.