An Evolutionary Algorithm for Matrix-Variate Model-Based Clustering

Tommy Flynn Paul D. McNicholas

Department of Mathematics and Statistics, McMaster University, Ontario, Canada.

Abstract

An evolutionary algorithm (EA) is developed as an alternative to the expectation-maximization (EM) algorithm for parameter estimation in matrix-variate model-based clustering. Using the principles of biological evolution, EAs offer a distinct way to search the likelihood surface, helping to avoid local maxima. EAs have been developed for multivariate model-based clustering, however, there remains a paucity of research in the matrix-variate setting. In response to the growing interest in matrix-variate data applications, this work develops a crossover and mutation based EA for mixtures of matrix-variate normals. We find that its performance is competitive against the EM algorithm for clustering three-way data in both simulated and real-world datasets.

Keywords: clustering; mixture models; evolutionary algorithm; matrix-variate; three-way data; crossover; mutation.

1 Introduction

Model-based clustering is the use of finite mixture models to identify underlying group structures in data. Estimating parameters for mixture models is primarily accomplished using the expectation-maximization (EM) algorithm (**?**); however; it is susceptible to becoming trapped at local maxima (**?**). An alternative approach is the evolutionary algorithm (EA) which iteratively emulates natural selection on a population of candidate solutions. By leveraging a fitness function and genetic operators like crossover and mutation, EAs offer a distinct way to search the likelihood surface.

While EAs have been successful for multivariate model-based clustering (**??**), there remains a paucity of research in the matrix-variate setting. Matrix-variate or three-way data consists of random matrices organized in three dimensions. This structure commonly arises in multivariate longitudinal data, with multiple measurements taken at different time points, and greyscale image data. Model-based clustering has demonstrated its effectiveness in clustering three-way data by leveraging mixtures of matrix-variate distributions (**?????**).

In response to the growing interest in matrix-variate data applications, this work develops an EA for matrix-variate model-based clustering. The algorithm incorporates both crossover and mutation, and is applied in the matrix-variate normal setting. We find that its performance is competitive against the EM algorithm for clustering three-way data in both simulated and real-world datasets.

2 Background

2.1 Matrix-Variate Normal Distribution

Three-way data refers to datasets comprised of random matrices organized in three dimensions. These datasets are characterized by having n units (rows), p variables (columns), and N occasions (layers). Matrix-variate distributions provide an effective way to model three-way data, the most mathematically tractable being the matrix-variate normal distribution (**?**). An $n \times p$ random matrix $\mathscr{X}$ follows an $n \times p$ matrix-variate normal distribution, denoted $\mathcal{N}_{n\times p}(\mathbf{M}, \mathbf{\Sigma}, \mathbf{\Psi})$, if its density can be written as

$$\phi_{n\times p}(\mathbf{X}|\mathbf{M}, \mathbf{\Sigma}, \mathbf{\Psi}) = \frac{1}{(2\pi)^{\frac{np}{2}}|\mathbf{\Sigma}|^{\frac{p}{2}}|\mathbf{\Psi}|^{\frac{n}{2}}} \exp\left\{-\frac{1}{2}\mathrm{tr}\left(\mathbf{\Sigma}^{-1}(\mathbf{X}-\mathbf{M})\mathbf{\Psi}^{-1}(\mathbf{X}-\mathbf{M})'\right)\right\},$$

where $\mathbf{M}$ is the $n \times p$ location matrix, $\mathbf{\Sigma}$ is the $n \times n$ row covariance matrix, and $\mathbf{\Psi}$ is the $p \times p$ column covariance matrix. An equivalent formulation in the multivariate setting is given by

$$\mathscr{X} \sim \mathcal{N}_{n\times p}(\mathbf{M}, \mathbf{\Sigma}, \mathbf{\Psi}) \iff \mathrm{vec}(\mathscr{X}) \sim \mathcal{N}_{np}(\mathrm{vec}(\mathbf{M}), \mathbf{\Psi} \otimes \mathbf{\Sigma}),$$

where $\mathcal{N}_{np}(\cdot)$ is the np multivariate normal distribution, $\mathrm{vec}(\cdot)$ is the vectorization operator, and $\otimes$ is the Kronecker product.

2.2 Model-Based Clustering

Classification is a paradigm in which group membership labels are assigned to unlabelled data. Unsupervised classification, or clustering, assumes that all observations are unlabelled or are treated as such. One common method of clustering is model-based, which involves maximizing the likelihood of a G-component finite mixture model. A matrix-variate random variable $\mathscr{X}$ arises from a finite mixture model if its density can be written as

$$f(\mathbf{X}|\boldsymbol{\vartheta}) = \sum_{g=1}^{G} \pi_g f_g(\mathbf{X}|\boldsymbol{\theta}_g),$$

where $f_g(\cdot)$ is the gth component density, $\pi_g > 0$ is the gth mixing proportion with $\sum_{g=1}^{G} \pi_g = 1$, and $\boldsymbol{\vartheta} = (\boldsymbol{\pi}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_G)$ is the vector of parameters with $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_G)$. In most applications, the component density functions are taken to be identical and Gaussian, that is, $f_g(\mathbf{X}|\boldsymbol{\theta}_g) = \phi(\mathbf{X}|\boldsymbol{\theta}_g)$ for all g.

? traces the framing of a cluster in terms of a component of a mixture model back to **?**, and the earliest use of a mixture model for clustering to **?**, who uses a multivariate Gaussian mixture model. A recent review of model-based clustering can be found in **?**, while extensive details are available in **?**.

2.3 The EM Algorithm

Estimation of the mixture model parameters is typically performed using the EM algorithm (**?**). Consider a dataset comprising N unlabelled matrices $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ of size $n \times p$. Then the observed-data likelihood for a finite matrix-variate normal mixture is given by

$$\mathcal{L}(\boldsymbol{\vartheta}) = \prod_{i=1}^{N} \sum_{g=1}^{G} \pi_g \phi(\mathbf{X}_i|\mathbf{M}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Psi}_g).$$

To facilitate clustering, we introduce latent membership indicators z_{ig} where

$$z_{ig} = \begin{cases} 1 & \text{if observation } \mathbf{X}_i \text{ belongs to component } g, \\ 0 & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, N; g = 1, \ldots, G$. The indicators can be organized into row vectors $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N$ where $\mathbf{z}_i = (z_{i1}, z_{i2}, \ldots, z_{iG})$ denotes the membership label for data point i, and further arranged into an $N \times G$ solution matrix denoting a clustering of the data

$$\boldsymbol{\mathcal{Z}} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_N \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1G} \\ z_{21} & z_{22} & \cdots & z_{2G} \\ \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \cdots & z_{NG} \end{bmatrix}.$$

Using the data and membership labels, we obtain the complete-data log-likelihood as follows

$$l_c(\boldsymbol{\vartheta}) = \sum_{i=1}^{N}\sum_{g=1}^{G} z_{ig}\left[\log \pi_g + \log \phi(\mathbf{X}_i|\mathbf{M}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Psi}_g)\right].$$

For each iteration of the EM algorithm, the E-step replaces the z_{ig} indicators with their conditional expected values given the data and the current parameter estimates

$$\hat{z}_{ig} = \mathbb{E}[z_{ig}|\mathbf{X}_i, \hat{\boldsymbol{\vartheta}}] = \mathbb{P}[z_{ig} = 1|\mathbf{X}_i, \hat{\boldsymbol{\vartheta}}] = \frac{\hat{\pi}_g \phi(\hat{\mathbf{X}}_i|\hat{\mathbf{M}}_g, \hat{\boldsymbol{\Sigma}}_g, \hat{\boldsymbol{\Psi}}_g)}{\sum_{h=1}^{G} \hat{\pi}_h \phi(\hat{\mathbf{X}}_i|\hat{\mathbf{M}}_h, \hat{\boldsymbol{\Sigma}}_h, \hat{\boldsymbol{\Psi}}_h)}.$$

The predictions are reported as either soft or hard classifications. In soft classification, the values of $\hat{z}_{ig}$ remain as probabilities in $[0, 1]$, as computed during the E-step. Alternatively, hard classification reports the maximum a posteriori (MAP) classification i.e, $\text{MAP}\{\hat{z}_{ig}\}$, where

$$\text{MAP}\{\hat{z}_{ig}\} = \begin{cases} 1 & \text{if } g = \text{argmax}_h\{\hat{z}_{ig}\}, \\ 0 & \text{otherwise.} \end{cases}$$

In the M-step, the parameters are updated. **?** derives the matrix-variate normal maximum likelihood estimates as

$$\begin{aligned}
\hat{\pi}_g &= \frac{N_g}{N}, \quad N_g = \sum_{i=1}^{N} \hat{z}_{ig}, \\
\hat{\mathbf{M}}_g &= \frac{\sum_{i=1}^{N} \hat{z}_{ig}\mathbf{X}_i}{N_g}, \\
\hat{\boldsymbol{\Sigma}}_g &= \frac{\sum_{i=1}^{N} \hat{z}_{ig}(\mathbf{X}_i - \hat{\mathbf{M}}_g)\hat{\boldsymbol{\Psi}}_g^{-1}(\mathbf{X}_i - \hat{\mathbf{M}}_g)'}{pN_g}, \\
\hat{\boldsymbol{\Psi}}_g &= \frac{\sum_{i=1}^{N} \hat{z}_{ig}(\mathbf{X}_i - \hat{\mathbf{M}}_g)\hat{\boldsymbol{\Sigma}}_g^{-1}(\mathbf{X}_i - \hat{\mathbf{M}}_g)'}{nN_g},
\end{aligned}$$

for $g = 1, \ldots, G$. The E and M steps iterate until some convergence criterion is satisfied. We use a criterion based on the Aitken acceleration (**?**) defined by

$$a^{(t)} = \frac{l^{(t+1)} - l^{(t)}}{l^{(t)} - l^{(t-1)}},$$

where $l^{(t)}$ is the observed log-likelihood at iteration t. From this, **?** and **?** calculate the asymptotic estimate of the log-likelihood at iteration $t + 1$ as

$$l_{\infty}^{(t+1)} = l^{(t)} + \frac{1}{1 - a^{(t)}}(l^{(t+1)} - l^{(t)}).$$

In accordance with **?**, the EM algorithm terminates when

$$0 < l_{\infty}^{(t+1)} - l^{(t)} < \varepsilon,$$

for some pre-specified tolerance $\varepsilon > 0$.

2.4 Model Selection and Performance

In general, the number of underlying components or clusters is a priori unknown. To select the appropriate number, we use the Bayesian information criterion (BIC; **?**) defined by

$$\text{BIC} = 2l(\hat{\boldsymbol{\vartheta}}) - 2\rho \log N,$$

where $l(\hat{\boldsymbol{\vartheta}})$ is the maximized log-likelihood, N is the number of observations, and ρ is the number of free parameters. Arguments for the use of the BIC for mixture-models can be found in **?**, **?**, and **?**.

Model performance is based on the adjusted Rand index (ARI; **?**) which quantifies the pairwise agreements between two sets of class assignments after accounting for agreement by chance. Under random assignment, the expected ARI is 0, while a value of 1 is achieved for perfect assignment. Negative values indicate worse than random assignment.

2.5 Benefits Over Vectorization

When dealing with matrix-variate clustering, it is tempted to revert to the multivariate setting through vectorization, permitting the use of well-established techniques. However, matrix-variate analysis often offers a substantial reduction in the number of free scale parameters. **?** demonstrates a free scale parameter reduction from $(n^2p^2+np)/2$ to $(n^2+p^2+n+p)/2$, which simplifies and speeds up the procedure for almost all values of n and p. Therefore, it is much more profitable to remain in the matrix-variate setting when dealing with three-way data.

2.6 Evolutionary Algorithms

Evolutionary algorithms (EA) exploit the principles of biological evolution for global optimization. They explore the solution space without assuming any prior knowledge of its structure making them remarkably effective against challenging optimization problems. This treatment focuses on the perspective detailed in **?**. The goal of an EA is to maximize a fitness function by evolving a population of candidate solutions. In contrast to other optimization techniques that yield only a single solution, EAs offer the advantage of obtaining a population of optimal solutions.

To begin, the first generation of candidate solutions is initialized. For each generation, genetic operators such as crossover and mutation are applied to the population members giving rise to new individuals. Only the fittest individuals among the population survive ensuring the propagation of advantageous traits. This process continues until the population stagnates to a collection of maximally fit solutions.

The principal mechanism for reproduction is the crossover operator, which randomly exchanges genetic material between two or more parent solutions to create at least one offspring. This promotes exploration of the solution space by creating new and diverse candidate solutions. Over many generations, the offspring inherit favourable traits from their parents leading to fitter solutions. Alternatively, mutation is a genetic operator that acts on individual population members by randomly introducing slight genetic modifications. This prevents premature convergence to suboptimal populations that may result from crossover alone.

Early applications of EAs for model-based clustering can be found in **?** and **?**, which focus on evolving the parameter space. More recently, the works of **?** and **?** have evolved a population of latent cluster membership matrices. The main advantage of evolving membership labels instead of the parameters lies in the size of the respective spaces. The total number of possible classifications of N observations into G groups is finite, in contrast to the infinite parameter space (**?**). In this work, we aim to extend the crossover and mutation based multivariate Gaussian EA developed in **?** to the matrix-variate setting, allowing for broader applicability.

3 Methodology

3.1 Model and Fitness Function

The underlying model for our EA is a mixture of matrix-variate normal distributions. Representing individual clusterings of the data, matrices $\boldsymbol{\mathcal{Z}}_k$ consisting of indicator variables z_{ig} serve as population members. Unlike the EM algorithm which uses soft clustering, the EA estimates z_{ig} using hard clustering. For clarity, we denote $\hat{z}_{ig} \in [0,1]$ for the EM estimates and $\tilde{z}_{ig} \in \{0,1\}$ for the EA estimates. To compute the fitness of an individual solution, we first update the matrix-variate normal model parameters, replacing $\hat{z}_{ig}$ with $\tilde{z}_{ig}$, that is,

$$\begin{aligned}
\tilde{\pi}_g &= \frac{N_g}{N}, \quad N_g = \sum_{i=1}^{N} \tilde{z}_{ig}, \\
\tilde{\mathbf{M}}_g &= \frac{\sum_{i=1}^{N} \tilde{z}_{ig}\mathbf{X}_i}{N_g}, \\
\tilde{\mathbf{\Sigma}}_g &= \frac{\sum_{i=1}^{N} \tilde{z}_{ig}(\mathbf{X}_i - \tilde{\mathbf{M}}_g)\tilde{\mathbf{\Psi}}_g^{-1}(\mathbf{X}_i - \tilde{\mathbf{M}}_g)'}{pN_g}, \\
\tilde{\mathbf{\Psi}}_g &= \frac{\sum_{i=1}^{N} \tilde{z}_{ig}(\mathbf{X}_i - \tilde{\mathbf{M}}_g)\tilde{\mathbf{\Sigma}}_g^{-1}(\mathbf{X}_i - \tilde{\mathbf{M}}_g)'}{nN_g},
\end{aligned}$$

for $g = 1, \dots, G$. Then we calculate the observed log-likelihood at the current EA estimates

$$\text{fitness}(\tilde{\boldsymbol{\mathcal{Z}}}_k) = \sum_{i=1}^{N} \log \left(\sum_{g=1}^{G} \pi_g f(\mathbf{X}_i | \tilde{\mathbf{M}}_g, \tilde{\boldsymbol{\Sigma}}_g, \tilde{\boldsymbol{\Psi}}_g) \right).$$

3.2 Algorithm Design

The EA starts by initializing the first generation with a population of K parent solutions, denoted as $\tilde{\boldsymbol{\mathcal{Z}}}_1, \dots, \tilde{\boldsymbol{\mathcal{Z}}}_K$. In general, the parent solutions are randomly generated matrices representing random hard clusterings of the data. Alternatively, one can provide initial solutions, such as a hardened k-means or other handpicked solutions, to inform the evolutionary process.

Reproduction in the EA involves cloning the parents J times and performing crossover on each clone. The crossover operation randomly swaps two distinct rows of the parent solution to create clones with identical membership labels except for two observations. For example, suppose we randomly select labels $\tilde{\mathbf{z}}_2$ and $\tilde{\mathbf{z}}_N$ from the parent in Figure **??**. Since the rows are distinct, we swap them to create a clone $\tilde{\boldsymbol{\mathcal{Z}}}_{k_j}$ with the same genetic material except for those two labels.

$$\tilde{\boldsymbol{\mathcal{Z}}}_k = \begin{bmatrix} \tilde{z}_{11} & \tilde{z}_{12} & \dots & \tilde{z}_{1G} \\ 0 & 0 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \dots & 0 \end{bmatrix} \xrightarrow{\text{crossover}} \tilde{\boldsymbol{\mathcal{Z}}}_{k_j} = \begin{bmatrix} \tilde{z}_{11} & \tilde{z}_{12} & \dots & \tilde{z}_{1G} \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

Figure 1: Illustration of the crossover operator, randomly swapping distinct rows $\tilde{\mathbf{z}}_2$ and $\tilde{\mathbf{z}}_N$ from the parent solution to create a child with similar genetic material.

If the labels are identical, we continue randomly selecting rows until two distinct labels are found. After crossover, the population size grows to $K + KJ$, including the original parents and their offspring. The population is then organized into a list of descending fitness, from which we select the top K solutions to become the next generation of parents. This crossover step helps avoid stopping at local maxima of the fitness surface, i.e., the observed log-likelihood.

Because swapping alone does not guarantee better clustering results, we now introduce a greedy mutation step. For each of the surviving K individuals, we randomly swap distinct elements in a random row until their fitness improves. Suppose we randomly select label $\tilde{\mathbf{z}}_2$ from the parent in Figure **??**. Continuing, we may randomly select distinct elements $\tilde{z}_{22}, \tilde{z}_{2G} \in \tilde{\mathbf{z}}_2$ and swap.

$$\tilde{\mathcal{Z}}_k = \begin{bmatrix} \tilde{z}_{11} & \tilde{z}_{12} & \dots & \tilde{z}_{1G} \\ \tilde{z}_{21} & 0 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{z}_{N1} & \tilde{z}_{N2} & \dots & \tilde{z}_{NG} \end{bmatrix} \xrightarrow{\text{mutation}} \tilde{\mathcal{Z}}_k = \begin{bmatrix} \tilde{z}_{11} & \tilde{z}_{12} & \dots & \tilde{z}_{1G} \\ \tilde{z}_{21} & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{z}_{N1} & \tilde{z}_{N2} & \dots & \tilde{z}_{NG} \end{bmatrix}$$

Figure 2: Illustration of the mutation operator, swapping distinct elements $\tilde{z}_{22}$ and $\tilde{z}_{2G}$ in a random row of a surviving parent for a slightly modified solution.

If the mutation increases fitness, the swap is kept; otherwise, the swap is reversed. This process continues until either a profitable mutation is found or all rows have been exhausted leaving the parent unchanged. If a generation remains unchanged after applying both crossover and mutation, it is considered a stagnation. After a predetermined number of stagnations, the algorithm terminates, resulting in a population of K fit solution matrices.

3.3 Procedure

The subsequent data analysis adopts the clustering paradigm, treating all membership labels as unknown. For each dataset, two matrix-variate normal mixtures are applied: one using the EM algorithm for parameter estimation and the other using our EA. Both algorithms are initialized with a random start. Let $\mathcal{G}$ be the true number of classes, then the algorithms are run for $G = 2, \dots, \mathcal{G} + 1$. The value of G which yields the largest BIC is selected. In addition to selecting G, the EA must also determine the number of parents $K \in \{1, 2, 3\}$, and the number of clones $J \in \{4, 8, 12\}$. The number of parents is relatively low to ensure that mutations are applied sparingly, while the number of children is large to encourage crossover. For convergence, the EM algorithm uses the Aitken's based criterion with a tolerance of $\varepsilon = 10^{-6}$, while the EA uses $S = 3$ stagnations. Model performance is based on the final converged log-likelihood, AIC, total runtime, and EA to EM likelihood ratio. The EM algorithm is implemented using the `R` package **?** and the EA is available in `Julia` at **?** (**??**).

Algorithm 1 EA for matrix-variate model-based clustering.

```
    Input:
    $\boldsymbol{\mathcal{X}} = [\mathbf{X}_1, \dots, \mathbf{X}_N] \leftarrow n \times p \times N$ array of observations
    $\tilde{\mathfrak{Z}} = [\tilde{\boldsymbol{\mathcal{Z}}}_1, \dots, \tilde{\boldsymbol{\mathcal{Z}}}_K] \leftarrow N \times G \times K$ array of membership labels   $\triangleright$ random if not specified
    $G \leftarrow$ number of clusters
    $K \leftarrow$ number of parents
    $J \leftarrow$ number of clones
    $S \leftarrow$ max number of stagnations
 1: s = 0
 2: while $s < S$ do
 3:     for $k = 1$ to $K$ do                                      $\triangleright$ crossover step
 4:         for $j = 1$ to $J$ do
 5:             Crossover: randomly swap two distinct labels from parent $\tilde{\boldsymbol{\mathcal{Z}}}_k$ to get clone $\tilde{\boldsymbol{\mathcal{Z}}}_{k_j}$
 6:             Fitness: update model parameters and calculate log-likelihood of the clone
 7:         end for
 8:     end for
 9:     Survival: sort parents and clones by descending fitness and take top $K$ as new parents
10:     for $k = 1$ to $K$ do                                      $\triangleright$ mutation step
11:         for $r$ in random permutation of 1 to $N$ do
12:             Mutate: swap two distinct elements in row $r$
13:             if fitness increases then
14:                 break for
15:             else
16:                 swap back
17:             end if
18:         end for
19:     end for
20:     if parents are identical to last generation then
21:         $\mathsf{s} \leftarrow \mathsf{s} + 1$
22:     else
23:         $\mathsf{s} \leftarrow 0$
24:     end if
25: end while
    Return final population $\tilde{\mathfrak{Z}}$
```

4 Simulation

Two simulations are conducted by clustering a collection of datasets generated from mixtures of matrix-variate normals. In each simulation, a total of 25 datasets are created from the same set of arbitrarily selected model parameters. Simulation 1 involves 3×4 data consisting of $G = 2$ true classes and a total of $N = 300$ observations. Simulation 2 considers 4×3 data generated from $\mathcal{G} = 3$ true classes, totalling $N = 300$ observations. All datasets are balanced, with equal proportions in each class. The groups in Simulation 1 are well-separated, whereas the groups in Simulation 2 are more challenging to distinguish.

4.1 Simulation 1

In Simulation 1, the location parameters set to

$$\mathbf{M}_1 = \begin{bmatrix} 1 & 0 & 1 & -1 \\ -1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}, \quad \mathbf{M}_2 = \begin{bmatrix} 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 1 & 0 & 1 & -1 \end{bmatrix},$$

and the scale parameters are given by

$$\mathbf{\Sigma}_1 = \begin{bmatrix} 1 & 0.4 & 0.75 \\ 0.4 & 1 & 0 \\ 0.75 & 0 & 1 \end{bmatrix}, \qquad \mathbf{\Sigma}_2 = \begin{bmatrix} 1 & 0.6 & 0.25 \\ 0.6 & 1 & 0.1 \\ 0.25 & 0.1 & 1 \end{bmatrix},$$
$$\mathbf{\Psi}_1 = \begin{bmatrix} 1 & 0 & 0.35 & 0.15 \\ 0 & 1 & 0 & 0.85 \\ 0.35 & 0 & 1 & 0 \\ 0.15 & 0.85 & 0 & 1 \end{bmatrix}, \qquad \mathbf{\Psi}_2 = \begin{bmatrix} 1 & 0.2 & 0 & 0.6 \\ 0.2 & 1 & 0.55 & 0 \\ 0 & 0.55 & 1 & 0.3 \\ 0.6 & 0 & 0.3 & 1 \end{bmatrix}.$$

The optimal number of components for the EM algorithm was $G = 2$ and the optimal EA consisted of $G = 2$, $K = 1$, and $J = 12$. Table **??** presents the mean and standard deviation for ARI, runtime, and EA to EM likelihood ratio of the optimal models across the 25 runs. The results indicate that both the EM and EA exhibited nearly identical performance. While the EA found slightly superior maxima, as evidenced by the average likelihood ratio, the EM achieved slightly better ARI scores. We acknowledge that the EA is generally slower than the EM due to its more computationally intensive nature; however, the runtimes appear comparable in this case since the number of parents is quite low.

Table 1: Mean and standard deviation of ARI, runtime, and likelihood ratio for EA and EM associated with Simulation 1.

	ARI	**Runtime (sec)**	**Likelihood Ratio**
EA	0.992 (0.010)	1.56 (0.19)	1.001 (0.005)
EM	0.993 (0.008)	0.77 (0.06)	

4.2 Simulation 2

The location parameters for simulation 2 are given by

$$\mathbf{M}_1 = \begin{bmatrix} 0 & 0.5 & 1 \\ 0.5 & 1 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{M}_2 = \begin{bmatrix} 1 & 0.5 & 0 \\ 1.5 & 1 & 2 \\ 0 & 2 & 0.5 \\ 1.5 & 0.5 & 1 \end{bmatrix}, \quad \mathbf{M}_3 = \begin{bmatrix} 1.5 & 2.5 & 2 \\ 1 & 3 & 1.5 \\ 0.5 & 3 & 1.5 \\ 1.5 & 0.5 & 1 \end{bmatrix},$$

and the scale parameters configured to

$$\mathbf{\Sigma}_1 = \mathbf{\Sigma}_3 = \begin{bmatrix} 1 & 0.1 & 0.45 & 0.1 \\ 0.1 & 1 & 0.25 & 0.35 \\ 0.45 & 0.25 & 1 & 0.1 \\ 0.1 & 0.35 & 0.1 & 1 \end{bmatrix}, \qquad \mathbf{\Sigma}_2 = \begin{bmatrix} 1 & 0.2 & 0 & 0.6 \\ 0.2 & 1 & 0.55 & 0 \\ 0 & 0.55 & 1 & 0.3 \\ 0.6 & 0 & 0.3 & 1 \end{bmatrix},$$

$$\mathbf{\Psi}_1 = \begin{bmatrix} 1 & 0.4 & 0.75 \\ 0.4 & 1 & 0 \\ 0.75 & 0 & 1 \end{bmatrix}, \qquad \mathbf{\Psi}_2 = \mathbf{\Psi}_3 = \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0 \\ 0.5 & 0.5 & 1 \end{bmatrix}.$$

The number of components selected for the EM algorithm was $G = 3$, while the optimal EA was identified with $G = 3$, $K = 3$, and $J = 12$. Table **??** presents similar results to the previous simulation, with the EA exhibiting marginally superior performance in terms of likelihood, while the EM slightly outperformed in ARI scores. Notably, the larger number of parents resulted in significantly longer runtimes due to the greedy nature of the mutation step.

Table 2: Mean and standard deviation of ARI, runtime, and likelihood ratio for EA and EM associated with Simulation 2.

	ARI	**Runtime (sec)**	**Likelihood Ratio**
EA	0.930 (0.032)	14.72 (1.82)	1.041 (0.101)
EM	0.942 (0.026)	1.83 (0.42)	

5 Application

Two real-world datasets are used for clustering. First is the Landsat satellite dataset, which involves 3×3 digital images of the same regions taken in four different spectral bands. The pixels are arranged into matrices of size 4×9 and three classes are used. The second dataset contains 16×16 greyscale images of handwritten digits. We focus on clustering digits 1 and 7.

5.1 Landsat Satellite Dataset

This dataset is a collection of digital images captured by the Landsat program, which is a series of Earth-observing satellites managed by NASA and the United States Geological Survey (USGS). The original data was purchased from NASA by the Australian Centre for Remote Sensing and subsequently donated to the UCI machine learning repository in a preprocessed form (**?**).

Four digital images of size 3×3 were taken of the same region in different spectral bands, corresponding to random matrices of size 4×9. Two of the bands were in the visible spectrum, while the other two were in the infrared spectrum. The pixel intensities range from 0 to 255. The researchers performed site visits to identify seven classes of images based on the four central pixels, which are described in Table **??**. Note that they removed class 6 due to doubts of its validity.

Table 3: Description of the Landsat test set classes and observation count.

Class	**Description**	**Count**
1	Red soil	461
2	Cotton crop	224
3	Grey soil	396
4	Damp grey soil	211
5	Soil with vegetation stubble	237
6	Mixture of all classes	0
7	Very damp grey soil	470

For our analysis, we focused on the first three classes from the test set: red soil, cotton crop, and grey soil, resulting in a dataset of size $N = 1081$ observations with true proportions approximately $\boldsymbol{\pi} = (0.43, 0.21, 0.37)$. Surprisingly, both the EM and EA resulted in $G = 4$ components. The presence of this extra class could be attributed to the lack of distinct borders between classes of images, as the images were labeled based on their central pixels. In fact, it is possible that these bordering regions should have been assigned to class 6, which was later removed. The optimal EA also comprised $K = 2$ parents and $J = 8$ clones. Table **??** showcases the results, including the log-likelihood, ARI, total runtime, and the EA to EM likelihood ratio. Additionally, Tables **??** and **??** present the cross-tabulation of the MAP classifications for the EA and EM, respectively. The results demonstrate that, despite an increase of approximately 5 minutes in runtime, the EA discovered a superior maximum. Moreover, an EA to EM likelihood ratio of approximately 1.55 indicates a substantial improvement in likelihood. This increase in EA likelihood translated to a slight increase in ARI over the EM. Notably, both algorithms exhibit a similar clustering pattern in the cross-tabulations.

Table 4: Converged log-likelihood, ARI, runtime, and EA to EM likelihood ratio associated with the Landsat dataset.

	Log-Likelihood	**ARI**	**Runtime (sec)**	**Likelihood Ratio**
EA	-108118.26	87.8	348.48	1.55
EM	-108118.7	86.9	36.89	

Table 5: Cross-tabulation of the EA MAP classifications associated with the Landsat dataset.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**
Red Soil	452	0	0	9
Cotton Crop	0	140	0	84
Grey Soil	7	0	368	21

Table 6: Cross-tabulation of the EM MAP classifications associated with the Landsat dataset.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**
Red Soil	450	0	0	11
Cotton Crop	0	150	0	74
Grey Soil	7	0	363	26

5.2 Handwritten Digits Dataset

The final dataset comprises greyscale images of handwritten digits, which were obtained by scanning envelopes from the U.S postal service. To facilitate analysis, the Elements of Statistical Learning provides preprocessed 16×16 images with pixel intensities between -1 and 1 (**?**).

Since the writing is mostly concentrated in the center, the outer rows and columns primarily contain white space, resulting in a value of -1. Consequently, the lack of variation along the borders leads to singular updates in the scale matrices $\mathbf{\Sigma}_g$ and $\mathbf{\Psi}_g$. To address this sparsity, we introduced small random noise to the images. However, to preserve the signal integrity, a constant is applied to all entries greater than -1 before adding the noise.

Our analysis focused on clustering digits 1 and 7 from the dataset. There were a total of $N = 411$ observations from the test set, with 264 belonging to digit 1 and 147 to digit 7, corresponding to true proportions of approximately $\boldsymbol{\pi} = (0.64, 0.36)$. The optimal number of components for the EM algorithm was $G = 2$, while the optimal EA consisted of $G = 2, K = 2$, and $J = 4$. In Table **??**, we present the log-likelihood, ARI, runtime, and EA to EM likelihood ratio. Additionally, Tables **??** and **??** display the cross-tabulation of EA and EM results, respectively. Remarkably, both algorithms showed identical clustering results in terms of likelihood and ARI. The cross-tabulations for EA and EM were also identical. However, we note that the EA required significantly more runtime to achieve the same results as the EM algorithm.

Table 7: Converged log-likelihood, ARI, runtime, and EA to EM likelihood ratio associated with the handwritten digits dataset.

	Log-Likelihood	**ARI**	**Runtime (sec)**	**Likelihood Ratio**
EA	-97320.28	90.4	43.75	1.00
EM	-97320.28	90.4	2.95	

Table 8: Cross-tabulation of the EA MAP classifications associated with the handwritten digits dataset.

	Cluster 1	**Cluster 2**
Digit 1	256	8
Digit 7	2	145

Table 9: Cross-tabulation of the EM MAP classifications associated with the handwritten digits dataset.

	Cluster 1	**Cluster 2**
Digit 1	256	8
Digit 7	2	145

6 Discussion

This work developed an EA for matrix-variate normal mixtures using crossover and mutation. The EA was applied to simulated and real-world datasets and its performance was compared against the EM algorithm. Overall, the EA proves to be a competitive alternative for fitting model parameters in matrix-variate normal mixtures, consistently performing at least as well as the EM algorithm in terms of likelihood and ARI. For the Landsat data, the EA outperformed the EM showcasing its potential to handle complex and ill-defined problems more effectively. However, its limitation lies in its runtime, which is understandable given its approach of finding multiple solutions and the nature crossover and mutation operators.

Future research could focus on improving the EA's runtime performance, either by exploring more optimal implementations of genetic operators, using parallel computing, or by leveraging lower-level languages such as `C`. Additionally, extending the investigation to non-Gaussian distributions could broaden the applicability of the proposed method. Lastly, exploring new genetic operators or modified versions of crossover and mutation could lead to further enhancements in the EA's performance.

Acknowledgements

References

Aitken, A. (1926). A series formula for the roots of algebraic and transcendental equations. *Proceedings of the Royal Society of Edinburgh 45*(1), 14–22.

Anderlucci, L. and C. Viroli (2015, jun). Covariance pattern mixture models for the analysis of multivariate heterogeneous longitudinal data. *The Annals of Applied Statistics 9*(2).

Andrews, J. L. and P. D. McNicholas (2013). Using evolutionary algorithms for model-based clustering. *Pattern Recognition Letters 34*(9), 987–992.

Ashlock, D. (2010). *Evolutionary Computation for Modeling and Optimization.* New York: Springer.

Bezanson, Jeff and Edelman, Alan and Karpinski, Stefan and Shah, Viral B. (2023). The Julia Programming Language. Julia Computing.

Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay (1994). The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family. *Annals of the Institute of Statistical Mathematics 46*(2), 373–388.

Dasgupta, A. and A. E. Raftery (1998). Detecting features in spatial point processes with clutter via model-based clustering. *Journal of the American Statistical Association 93*(441), 294–302.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the em-algorithm. *Journal of the Royal Statistical Society B 39*, 1–38.

Doğru, F., Y. Bulut, and O. Arslan (2016). Finite mixtures of matrix variate t distributions. *Gazi Univ. J. Sci. 29*(2), 335–341.

Flynn, T. (2023). MatrixVariateEA: An Evolutionary Algorithm for Model-Based Clustering. https://github.com/TommyJohnFlynn/MatrixVariateEA.jl.

Gallaugher, M. P. B. and P. D. McNicholas (2018). Finite mixtures of skewed matrix variate distributions. *Pattern Recognition 80*, 83–93.

Gupta, A. and D. Nagar (1999). *Matrix Variate Distributions*. Monographs and Surveys in Pure and Applied Mathematics. Taylor & Francis.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning*. New York: Springer.

Hubert, L. and P. Arabie (1985). Comparing partitions. *Journal of Classification 2*(1), 193–218.

Keribin, C. (2000). Consistent estimation of the order of mixture models. *Sankhyā: The Indian Journal of Statistics, Series A*, 49–66.

Leroux, B. G. (1992). Consistent estimation of a mixing distribution. *The Annals of Statistics*, 1350–1360.

Lindsay, B. (1995). Mixture models: Theory, geometry, and applications. In *NSF-CBMS Regional Conference Series in Probability and Statistics*, Volume 5. Hayward: Institute of Mathematical Statistics.

Martínez, A. M. and J. Vitrià (2000). Learning mixture models using a genetic version of the em algorithm. *Pattern Recognition Letters 21*(8), 759–769.

McNicholas, P. D. (2016a). *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press.

McNicholas, P. D. (2016b). Model-based clustering. *Journal of Classification 33*(3), 331–373.

McNicholas, P. D., T. B. Murphy, A. F. McDaid, and D. Frost (2010). Serial and parallel implementations of model-based clustering via parsimonious gaussian mixture models. *Computational Statistics & Data Analysis 54*(3), 711–723.

McNicholas, S. M., P. D. McNicholas, and D. A. Ashlock (2020). An evolutionary algorithm with crossover and mutation for model-based clustering. *Journal of Classification*, 1–16.

Pernkopf, F. and D. Bouchaffra (2005). Genetic-based em algorithm for learning gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*(8), 1344–1348.

R Core Team (2023). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics 6*(2), 461–464.

Silva, A., X. Qin, S. J. Rothstein, P. D. McNicholas, and S. Subedi (2023, 04). Finite mixtures of matrix variate Poisson-log normal distributions for three-way count data. *Bioinformatics 39*(5), btad167.

Srinivasan, A. (1993). Statlog (Landsat Satellite). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C55887.

Tiedeman, D. V. (1955). On the study of types. In *Symposium on pattern analysis.*

Titterington, D., A. Smith, and U. Makov (1985). *Statistical analysis of finite mixture distributions.* Chichester: John Wiley & Sons.

Tomarchio, S. D., M. P. Gallaugher, A. Punzo, and P. D. McNicholas (2023). *MatrixMixtures: Model-Based Clustering via Matrix-Variate Mixture Models.* R package version 1.0.0.

Viroli, C. (2011). Finite mixtures of matrix normal distributions for classifying three-way data. *Statistical Computing 21*(4), 511–522.

Wolfe, J. (1965). A computer program for the maximum-likelihood analysis of types. *USNPRA Technical Bulletin*, 15–65.