Methods for Merging, Parsimony and Interpretability of Finite Mixture Models

by

Nam-Hwui Kim

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Statistics

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Volodymyr Melnykov Professor, ISM, University of Alabama
Supervisor:	Ryan Browne Associate Professor, Dept. of Stat. and Act. Sci., University of Waterloo
Internal Member:	Pengfei Li Professor, Dept. of Stat. and Act. Sci., University of Waterloo
Internal Member:	Paul Marriott Professor, Dept. of Stat. and Act. Sci., University of Waterloo
Internal-External Member:	Jeff Orchard Professor, School of Computer Science, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The chapters of this thesis describe and contain Nam-Hwui Kim's work, under the supervision of Dr. Ryan Browne, that has been published or submitted for publication in the following venues.

- Chapter 3: Kim, N.-H. and Browne, R.P. (2021). In the Pursuit of Sparseness: A New Rank-preserving Penalty for a Finite Mixture of Factor Analyzers. *Computational Statistics and Data Analysis* 160: 107244.
- Chapter 4: Kim, N.-H. and Browne, R.P.. Regularized Cluster-preserving Dimension Reduction with the Stiefel Elastic Net Discriminant Variables. Submitted to *Journal of Classification*.
- Chapter 5: Kim, N.-H. and Browne, R.P. (2022). Anderson Relaxation Test for Intrinsic Dimension Selection in Model-based Clustering. *Journal of Statistical Computation and Simulation*: 1-20.
  - The idea for this work was generated during Nam-Hwui's MMath Statistics project at the University of Waterloo, and was developed and completed during his PhD Statistics programme.
- Chapter 6: Kim, N.-H. and Browne, R.P.. Flexible Mixture Regression with the Generalized Hyperbolic Distribution. Submitted with revision to *Advances in Data Analysis and Classification*.
- Chapter 7: Kim, N.-H. and Browne, R.P. (2021). Mode Merging for the Finite Mixture of t-distributions. *Stat* 10(1): e372.

Abstract

To combat the increasing data dimensionality, parsimonious modelling for finite mixture models has risen to be an active research area. These modelling frameworks offer various constraints that can reduce the number of free parameters in a finite mixture model. However, the constraint selection process is not always clear to the user. Moreover, the relationship between the chosen constraint and the data set is often left unexplained. Such issues affect adversely the interpretability of the fitted model. That is, one may end up with a model with reduced number of free parameters, but how it was selected, and what the parameter-reducing constraints mean, remain mysterious.

Over-estimation of the mixture component count is another way in which the model interpretability may suffer. When the individual components of a mixture model fail to capture adequately the underlying clusters of a data set, the model may compensate by introducing extra components, thereby representing a single cluster with multiple components. This reality challenges the common assumption that a single component represents a cluster.

Addressing the interpretability-related issues can improve the informativeness of model-based clustering, thereby better assisting the user during the exploratory analysis and/or data segmentation step.

Acknowledgements

I am indebted to many, many people in my career thus far. Though I could not fit everyone in here, I extend my sincere gratitude to all of you.

I thank my supervisor, Dr. Ryan Browne, for our academic journey that set me up on a path toward independent research. Your consistent support through my peaks and troughs encouraged me to improve and achieve. I am glad to have been advised by you, and I will always be grateful for the chance you took on me.

I extend my gratitude to all the other teachers along the way, who have influenced my personal and academic growth. Mr. Pike, with a relaxed charisma in his classroom, suggested that I consider mathematics more seriously. Mr. Hampton-Cole pushed me to explore beyond the school curricula, and exchanged ideas with me like a fellow adult. Dr. Wagner primed my curiosity toward mathematics, and Dr. Drekic helped me gain valuable teaching experiences. I thank you all for your guidance.

Now to my family, for no tree is rootless. My parents who risked it all and emigrated in search of better opportunities for me and my sister. Time and time again I realize the monumental scale of their dedication and sacrifice - language barriers, cultural and financial stress, and safety concerns to name a few. My grandparents for their unconditional cheer and faith in me, in this life and beyond. My sister for sharing her unimaginable maturity and relaxed perspective toward life and the fun within. I am, and will always be, truly grateful for you all.

To Ziyi: No words can describe fully how fortunate I feel for having met you. Your resilience and wisdom have inspired me daily, and I would not have it any other way. You have been my anchor through turbulence, and the amplifier of my joy. Thank you for your brilliance, love and support. Thank you for being you.

Now onto the party. To Thomas, Amar and Kevin. (aka. TANK members). I would not have survived the math courses had I not bothered you all on that fateful first week of calculus. I hope your masterpiece has made you proud. Thank you for your camaraderie, friendship, and of course, cringy dad jokes.

Continuing with the party, I owe my sanity to my friends in the department. Especially Ilia for our academic struggles (I would never forget the midterm on convocation day), tea trips, gaming sessions and ploys, Gracia for listening to my rants and calling out my nonsense with surgical precision, Chi-Kuang for all the help and top-notch snacks, and Erik (now Dr. Hintz) for the chocolate bars and the spontaneous trips to McDonald's.

To all other friends and colleagues, thank you for enriching my academic journey, and packing my university days with joy and growth.

Dedication

‘남매는 단 둘이다’ 라고 가르쳐 주신 나의 아버지와
‘행복한 사람이 되라’며 길러주신 나의 어머니께.

For my father who taught me to cherish my sibling,
and for my mother who taught me to seek happiness above all.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Clustering frameworks attempt to construct heterogeneous groups in a sample without prior knowledge on the group membership status of observations within. One such example is model-based clustering, based on finite mixture models, which aims to represent the data set using a convex combination of probability mass (or density) functions (Wolfe, 1963). Thanks to its ubiquity, model-based clustering has advanced in numerous fronts. McLachlan and Peel (2004); McNicholas (2016) provide an overview on some modern developments in the field.

The nature of clustering is often exploratory, where the investigator may be conducting a more 'hands-on' type of analysis. This means that the method's capability in informing the user, in addition to the quality of its fit on the sample, is important. Mixture model interpretability concerns the above notion at large. One may draw a parellel between interpretability and happiness; because of their nebulous definitions, we are unable to measure them directly. Instead, we study their proxy measures. For example, happiness could be measured approximately via one's wealth, work-life balance, number of friends, and so on. Similarly, model interpretability can be approached from various angles.

This collection of work contributes to interpretability of model-based clustering methods through penalization, dimensionality reduction, and merging/combining of mixture components. Penalization aims to suppress signals from less important variables while

emphasizing that from more important ones. Such favouring is often manifested through sparser parameter estimates, which reduces the number of variables (and associations between them) warranting the user's attention. Dimensionality reduction combines original set of variables into a smaller set, so that the sample may exhibit better-separated groups. Mixture component merging and combining aim to collect sufficiently similar (based on a carefully chosen measure) components into a single group, so that a simpler grouping structure may be discovered from the sample.

The remainder of this thesis is organized as follows. Chapter 2 outlines some foundational concepts that appear throughout the thesis. Chapters 3 and 4 introduce a penalization framework for matrix-variate parameters and its applications. Chapter 5 introduces a hypothesis test-based method of estimating an adequate number of dimensions for projection. Chapter 6 introduces a novel mixture regression model with a component combining procedure for identifying simpler response-covariate relationships. Chapters 7 and 8 focus on mode-based component merging for various families of non-Gaussian finite mixtures. We then conclude with a brief summary in chapter 9.

Chapter 2

Background

2.1 Finite Mixture Models

A finite mixture model is a probabilistic model defined by a convex combination of finitely many probability mass or density functions, abbreviated by pmf and pdf respectively. Each pmf or pdf is referred to as a component, and the components are usually of the same family of distributions. Denote by f the pmf or pdf of a p-dimensional G-component finite mixture model. Then, we can write f as

$$f(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{g=1}^{G} \pi_g f_g(\boldsymbol{x};\boldsymbol{\Theta}_g) \quad \text{subject to} \quad \sum_{g=1}^{G} \pi_g = 1, \tag{2.1}$$

where, for each component g, f_g, π_g (where $\pi_g > 0$) and $\boldsymbol{\Theta}_g$ denote the pmf or pdf, mixing proportion parameter and the set of distribution parameters, respectively. $\boldsymbol{\Theta} = \{\pi_1, \ldots, \pi_G, \boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_G\}$ denotes the set of all parameters of f. A finite mixture distribution is often used to model the heterogeneous sub-populations within a larger population (McLachlan and Peel, 2004). Moreover, it is a highly flexible tool for density estimation, as a finite mixture model with a sufficient number of components can estimate an arbitrary pdf with an arbitrary level of accuracy (Titterington et al., 1985). A classic example is the

Gaussian finite mixture (GMM), where every component pdf follows a p-dimensional Gaussian distribution parametrized by a mean vector $\boldsymbol{\mu}_g$ and a covariance matrix $\boldsymbol{\Sigma}_g$. There is a plethora of literature on model-based clustering. Starting from the early works by Day (1969); Wolfe (1967, 1970), numerous finite mixture models with non-Gaussian component distributions have been developed so far. Examples include the parsimonious Gaussian Fraley and Raftery (2002), t (Peel and McLachlan, 2000), skew-normal (Lin et al., 2007b), skew-t (Lin et al., 2007a), shifted asymmetric Laplace (Franczak et al., 2013) and generalized hyperbolic (Browne and McNicholas, 2015) distributions. The development of these non-Gaussian finite mixtures was motivated by the increasing complexity of the available data sets and the group structure within. Of course, as noted earlier, a GMM with a large enough number of components could be fitted instead. However, such strategy would result in a verbose model; where one flexible-enough distribution could be sufficient, multiple Gaussian distributions may be needed.

The Expectation-Maximization (EM) algorithm by Dempster et al. (1977) is commonly used to fit a finite mixture model for many reasons, two of which are the relative ease of estimation and the monotonicity in terms of likelihood function. Several variations of the EM algorithm exist, such as the Expectation-Conditional Maximization (ECM) algorithm by Meng and Rubin (1993) and Alternating Expectation-Conditional Maximization (AECM) algorithm by Meng and Van Dyk (1997). The Stochastic EM (SEM) algorithm by Celeux and Diebolt (1985) is another noteworthy variant, where random sampling is incorporated to allow the convergence path to 'escape' from poor initial values.

Under the EM algorithm framework (and its variants) for finite mixture models, the observed data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is deemed incomplete, because we do not know the component to which each $\boldsymbol{x}_i$ belongs to. Thus a latent (unobserved) component membership indicator vector $\boldsymbol{Z}_i = (Z_{i1}, \ldots, Z_{iG})'$ is introduced, where $Z_{ig} = 1$ (with probability π_g) if $\boldsymbol{x}_i$ belongs to component g and 0 otherwise. We denote by $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{iG})'$ a realization of $\boldsymbol{Z}_i$. If we suppose that we observe $\boldsymbol{z}_i$ as well, then the $(\boldsymbol{x}_i, \boldsymbol{z}_i)$ pair is considered complete. Depending on the component-wise distributions, more latent variables may be introduced. As an illustration of the EM algorithm, consider a G-component GMM. The observed-data

likelihood function is

$$L(\boldsymbol{\Theta}) = \prod_{i=1}^{n} \left(\sum_{g=1}^{G} \pi_g \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g) \right),$$

where $\boldsymbol{\mu}_g$ and $\boldsymbol{\Sigma}_g$ denote component-wise mean vector and covariance matrix respectively. Manipulating the sum inside the product is often very challenging. In contrast, the complete-data likelihood with $(\boldsymbol{x}_i, \boldsymbol{z}_i)$s is

$$L_c(\boldsymbol{\Theta}) = \prod_{i=1}^{n} \left\{ \prod_{g=1}^{G} (\pi_g \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g))^{z_{ig}} \right\},$$

and the corresponding log-likelihood $l_c(\boldsymbol{\Theta})$ is

$$l_c(\boldsymbol{\Theta}) = \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} \log(\pi_g \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)).$$

In reality, however, the latent variables (z_{ig} in this case) are not observed. Therefore, at every iteration, the EM algorithm obtains the conditional expectation of $l_c(\boldsymbol{\Theta})$ with respect to the latent variables given $\boldsymbol{x}_i$s. Letting $\boldsymbol{\Theta}^{(t)}$ be the estimate of $\boldsymbol{\Theta}$ at iteration t. Then, the conditional expectation is given by

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) = \mathrm{E}[l_c(\boldsymbol{\Theta})|\boldsymbol{x}_i, \boldsymbol{\Theta}^{(t)}] = \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig}^{(t)} \log(\pi_g \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)),$$

where $z_{ig}^{(t)} = \mathrm{E}[Z_{ig}|\boldsymbol{x}_i, \boldsymbol{\Theta}^{(t)}]$ is the posterior component membership probability estimate at iteration t, and it is equal to

$$z_{ig}^{(t)} = \frac{\pi_g^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_g^{(t)}, \boldsymbol{\Sigma}_g^{(t)})}{\sum_{k=1}^{G} \pi_k^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}. \tag{2.2}$$

The new model parameter estimates $\pi_g^{(t+1)}, \boldsymbol{\mu}_g^{(t+1)}, \boldsymbol{\Sigma}_g^{(t+1)}$ are obtained by maximizing Q

with respect to the corresponding parameters. Their update formulae are given by

$$\pi_g^{(t+1)} = \frac{\sum_{i=1}^n z_{ig}^{(t)}}{n}, \quad \boldsymbol{\mu}_g^{(t+1)} = \frac{\sum_{i=1}^n z_{ig}^{(t)} \boldsymbol{x}_i}{\sum_{i=1}^n z_{ig}^{(t)}}, \quad \boldsymbol{\Sigma}_g^{(t+1)} = \frac{\sum_{i=1}^n z_{ig}^{(t)} (\boldsymbol{x}_i - \boldsymbol{\mu}_g^{(t+1)})(\boldsymbol{x}_i - \boldsymbol{\mu}_g^{(t+1)})'}{\sum_{i=1}^n z_{ig}^{(t)}}.$$

Upon convergence, the final model parameter estimates are reported, and the component membership of each observation $\boldsymbol{x}_i$ is computed as the Maximum A Posteriori (MAP) estimate of $\hat{z}_{i1}, \ldots, \hat{z}_{iG}$ (the $z_{ig}^{(t)}$s at the time of convergence) where

$$\text{MAP}(\hat{z}_{ig}) = \begin{cases} 1 & \text{if } \underset{k=1,\ldots,G}{\text{argmax}}\{\hat{z}_{i1}, \ldots, \hat{z}_{iG}\} = g, \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

The convergence of the EM algorithm in model-based clustering can be determined by the consecutive difference in log-likelihood. For instance, let $\boldsymbol{\Theta}^{(t)}$ and $\boldsymbol{\Theta}^{(t-1)}$ denote the set of parameter estimates at iterations t and $t-1$. Then, given a pre-determined positive threshold ϵ, if the the difference in log-likelihood at the aforementioned two sets is less than ϵ, then the algorithm can be terminated. This condition is algebraically translated as

$$l(\boldsymbol{\Theta}^{(t)}) - l(\boldsymbol{\Theta}^{(t-1)}) < \epsilon.$$

However, per Lindstrom and Bates (1988), such a criterion represents a lack of progress, not the actual convergence of the algorithm. Aitken's acceleration by Aitken (1926) is a tool for accelerated convergence of a linearly convergent sequence, which the EM algorithm produces. Let $\{l(\boldsymbol{\Theta}^{(t)})\}$ denote the sequence of log-likelihood values generated by the EM algorithm, and suppose that its limit is $\hat{l}$. Then, the linear convergence rate of the EM algorithm dictates that, for some $a \in (0, 1)$,

$$\frac{l(\boldsymbol{\Theta}^{(t+1)}) - \hat{l}}{l(\boldsymbol{\Theta}^{(t)}) - \hat{l}} \approx a.$$

The Aitken acceleration coefficient at iteration t is used to approximate a, and it is defined

as

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

The limit $\hat{l}$ can then be approximated by

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

and Böhning et al. (1994) suggests the termination of algorithm when

$$0 < \hat{l}^{(t+1)} - \hat{l}^{(t)} < \epsilon. \tag{2.4}$$

A finite mixture model may be accompanied by several hyperparameters, most common of which is the component count G. In practice, the model parameters are estimated over a range of G values, and the one producing the best model selection criterion value is chosen. Several selection criteria exist, such as the log-likelihood value, Akaike Information Criterion (AIC) (Akaike, 1974), Bayesian Information Criterion (BIC) (Schwarz, 1978) and Integrated Completed Likelihood (ICL) (Biernacki et al., 2000), with the BIC being a common choice. Let $\hat{\boldsymbol{\Theta}}$ and $|\hat{\boldsymbol{\Theta}}|$ denote a realization of model parameter set and the number of free parameters within, respectively. Then, assuming a sample size of n, the formula for the AIC and the BIC are

$$\begin{aligned} \text{AIC}(\hat{\boldsymbol{\Theta}}) &= 2l(\hat{\boldsymbol{\Theta}}) - 2|\hat{\boldsymbol{\Theta}}|, \\ \text{BIC}(\hat{\boldsymbol{\Theta}}) &= 2l(\hat{\boldsymbol{\Theta}}) - \log(n)|\hat{\boldsymbol{\Theta}}|, \end{aligned} \tag{2.5}$$

where both are to be maximized. While the log-likelihood itself is the simplest, it is also most susceptible to verbose models, as it ignores the number of parameters. The AIC and BIC penalize on the parameter count, but the BIC exacerbates the penalty as the sample size increases. The ICL favours component counts producing well-defined clusters per Baudry et al. (2010), as opposed to the BIC (and similarly the AIC), which prioritizes on density estimation, favouring larger G values than the ICL. Given a set of complete-data

$\{(\boldsymbol{x}_i, \boldsymbol{z}_i)\}_{i=1,\ldots,n}$, the ICL is computed as

$$ICL(\hat{\Theta}) = \log\left(\prod_{i=1}^{n}\prod_{g=1}^{G}\left\{\hat{\pi}_g f_g(\boldsymbol{x}_i; \hat{\Theta}_g)\right\}^{\text{MAP}(\hat{z}_{ig})}\right) - \frac{|\hat{\Theta}|}{2}\log(n).$$

The key point here is that the model selection criterion may have a non-trivial impact on the final model.

In addition to selection criteria, the model's performance can be measured using the components produced. If the data set is accompanied by a ground truth (a known set of labels), then the Adjusted Rand Index (ARI) by Hubert and Arabie (1985) is commonly used. The ARI is an extension of the Rand Index by Rand (1971), which measures the extent of agreement between two sets of partition. Let $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a set of objects, and let $A = \{A_1, \ldots, A_G\}$ and $B = \{B_1, \ldots, B_K\}$ be two partitions of S. Furthermore, let

$$\begin{aligned} a &= \text{number of pairs in } S \text{ that are in the same subset of } A, \text{ as well as } B, \\ b &= \text{number of pairs in } S \text{ that are in different subsets of } A, \text{ as well as } B. \end{aligned}$$

Intuitively, a and b can be interpreted as the number of object pairs where A and B agree on in terms of grouping. Then, the Rand Index (RI) is defined as

$$RI = \frac{a+b}{\binom{n}{2}}.$$

The RI ranges between 0 and 1, and higher values indicate better agreement between A and B. If A is the ground truth and B is the estimated grouping from a clustering method, then the RI measures the agreement between the two. The ARI adjusts for chance by subtracting from the agreement between the ground truth and the model-generated grouping the expected agreement between the ground truth and a randomly-assigned grouping. Therefore, while the maximum value is still 1, the ARI can be negative if the model-generated grouping agrees less with the ground truth than a random assignment. More details can be found in Steinley (2004).

When a ground truth is unavailable, then the degree of separation between clusters can be measured. Two such measures are the Between-cluster Sum of Squares (BSS) (and its Within-cluster counterpart WSS), and the Additive Margin (AM) by Ben-David and Ackerman (2009). Let $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a data set and let $\mathcal{C} = \{P_1, \ldots, P_G\}$ be a clustering or a partition of the data. For example, the MAP estimates (2.3) can form $\mathcal{C}$. Suppose each partition has an associated centre point $\boldsymbol{\mu}_g$. Then, the BSS and WSS are defined as

$$BSS = \sum_{g=1}^{G} \left(\boldsymbol{\mu}_g - \sum_{k=1}^{G} \boldsymbol{\mu}_k / G \right)^2,$$
$$WSS = \sum_{g=1}^{G} \sum_{i:\boldsymbol{x}_i \in P_g} (\boldsymbol{x}_i - \boldsymbol{\mu}_g)^2,$$

and the degree of separation between clusters can be measured via the ratio between BSS and WSS. The AM is based on the comparison of the distance between an observation $\boldsymbol{x}$ and its two closest centres $\boldsymbol{\mu}_k$ and $\boldsymbol{\mu}_l$. The Additive Point Margin (APM) of an observation $\boldsymbol{x}$ is defined as

$$APM(\boldsymbol{x}) = d(\boldsymbol{x}, \boldsymbol{\mu}_l) - d(\boldsymbol{x}, \boldsymbol{\mu}_k),$$

where $d(\cdot, \cdot)$ denotes an appropriate distance function, $\boldsymbol{\mu}_k$ is the centre closest to $\boldsymbol{x}$ and $\boldsymbol{\mu}_l$ is the second closest centre to $\boldsymbol{x}$. In this thesis, $d(\cdot, \cdot)$ is assumed to be Euclidean. The AM of a clustering or partition is defined as

$$AM(\mathcal{C}) = \frac{\sum_{i=1}^{n} APM(\boldsymbol{x}_i)/n}{\sum_{g=1}^{G} \sum_{\{\boldsymbol{x},\boldsymbol{y}\} \in P_g} d(\boldsymbol{x}, \boldsymbol{y}) / \sum_{g=1}^{G} \binom{|P_g|}{2}},$$

where $|\cdot|$ denotes the cardinality of a set. The AM is non-negative, and higher values indicate better-defined clustering.

2.2 Parsimonious Model-based Clustering

Along with the complexity in structure, the dimension of available data sets have increased in the recent past. As suggested by the Curse of Dimensionality (Bellman, 2010), high-dimensional data sets pose additional modelling challenges compared to lower-dimensional data sets in terms of the required number of observations and model performance. In particular, the growing number of model parameters can lead to poorly-fitted models. Consider a p-dimensional G-component GMM. It has $(G-1)+Gp+Gp(p+1)/2$ free parameters, where each summand comes from the mixing proportions, mean vectors and covariance matrices respectively. Holding G constant, the number of free parameters is a quadratic function of p, implying that the number of required observations for model-fitting increases very quickly as the dimension increases. Several parsimonious finite mixture modelling frameworks have been developed to mitigate this problem, where the number of free parameters is reduced by constraining the parameter structures. The trade-off is that not every method is interpretable. That is, the user may need to select model constraints without understanding their meanings with respect to the problem-at-hand. Alternatively, the fitted constrained model may not reveal the aspects of the data that resulted in that set of constraints being chosen. Here, we present some commonly-deployed frameworks and discuss briefly their interpretability concerns.

- Banfield-type Eigen-decomposition (Banfield and Raftery, 1993). The component-wise scale matrices $\boldsymbol{\Sigma}_g$ are decomposed into

$$\boldsymbol{\Sigma}_g = \lambda_g \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g',$$

  where λ_g is the first eigenvalue of Σ_g, $\boldsymbol{D}_g$ is the diagonal matrix of scaled eigenvalues of Σ_g with the first entry equal to 1, and $\boldsymbol{P}_g$ is the matrix of eigenvectors. Here, λ_g represents the volume of the space occupied by a component, and $\boldsymbol{D}_g$ and $\boldsymbol{P}_g$ represent the shape and the orientation of the component respectively. By constraining any subset of $\{\lambda_g, \boldsymbol{D}_g, \boldsymbol{P}_g\}$ to be equal across components, we can choose the aspects of the component distributions to be held equal. Moreover, such equality

restrictions reduce the number of free parameters to be estimated. This framework was applied on the finite mixture of Gaussian (Fraley and Raftery, 2002), t (Andrews and Mcnicholas, 2012), shifted asymmetric Laplace (Franczak et al., 2013), generalized hyperbolic (Browne and McNicholas, 2015) and power exponential (Dang et al., 2019) distributions. While this framework can reduce the number of free parameters in component-wise scale matrices, the user cannot find out why certain constraints are favoured over others by the data set under consideration, other than relying on the scores from some model selection criteria. For example, suppose $\lambda_g = \lambda$ and $\boldsymbol{P}_g = \boldsymbol{P}$. Then, although this model tells us that the volume of space occupied by each component and their orientations are equal, we cannot tell why the data set chose that set of constraints.

- Subspace Clustering (Bouveyron et al., 2007). The subspace clustering framework for the Gaussian finite mixture (abbreviated as SC-GMM henceforth) is another parsimonious GMM framework based on linear projections and component-wise intrinsic dimensions. The intrinsic dimensions d_g $(g = 1, \ldots, G)$ of the p-dimensional data set are estimated as the number of distinguishable directions in the component-wise orthogonal bases, and the remaining $p - d_g$ directions are deemed indistinguishable. The directions are partitioned by the magnitude of the corresponding eigenvalues. Consider the eigen-decomposition of g^{th} component's covariance matrix $\boldsymbol{\Sigma}_g$, $\boldsymbol{\Sigma}_g = \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g'$, where $\boldsymbol{P}_g$ is the orientation matrix and $\boldsymbol{D}_g = \text{diag}(\lambda_1, \ldots, \lambda_p)$ is the diagonal matrix of eigenvalues (arranged in decreasing order). If its intrinsic dimension is d_g, $\boldsymbol{D}_g$ would assume the form $\text{diag}(a_{g1}, \ldots, a_{gd_g}, \underbrace{b_g, \ldots, b_g}_{p-d_g \text{ copies}})$, where the first d_g eigenvalues correspond to distinguishable directions and the remaining ones render their associated directions indistinguishable. Then, the d_g intrinsic-dimensional covariance $\boldsymbol{\Sigma}_g$ admits the following eigen-decomposition

$$\boldsymbol{\Sigma}_g = [\boldsymbol{\Gamma}_g \boldsymbol{\Xi}_g] \, \boldsymbol{D}_g \, [\boldsymbol{\Gamma}_g \boldsymbol{\Xi}_g]' \,,$$

where $\boldsymbol{\Gamma}_g$ and $\boldsymbol{\Xi}_g$ are the matrices consisting of d_g distinguishable and $p - d_g$ in-

distinguishable directions respectively. This structure bypasses the estimation of $\boldsymbol{\Xi}_g$, because the quadratic form is simplified as follows.

$$
\begin{aligned}
(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g) &= (\boldsymbol{x}-\boldsymbol{\mu}_g)'\left[\boldsymbol{\Gamma}_g\boldsymbol{\Xi}_g\right]\boldsymbol{D}_g^{-1}\left[\boldsymbol{\Gamma}_g\boldsymbol{\Xi}_g\right]'(\boldsymbol{x}-\boldsymbol{\mu}_g) \\
&= \sum_{j=1}^{d_g}\frac{1}{a_{gj}}\left([\boldsymbol{\Gamma}_g]_{\cdot j}'(\boldsymbol{x}-\boldsymbol{\mu}_g)\right)^2 + \frac{1}{b_g}\sum_{j=d_g+1}^{p}\left([\boldsymbol{\Xi}_g]_{\cdot j}'(\boldsymbol{x}-\boldsymbol{\mu}_g)\right)^2,
\end{aligned}
$$

where the second sum is simplified to

$$
\frac{1}{b_g}\sum_{j=d_g+1}^{p}\left([\boldsymbol{\Xi}_g]_{\cdot j}'(\boldsymbol{x}-\boldsymbol{\mu}_g)\right)^2 = \frac{1}{b_g}\sum_{j=1}^{p}\left\{(\boldsymbol{x}-\boldsymbol{\mu}_g)_j^2 - \left([\boldsymbol{\Gamma}_g]_{\cdot j}'(\boldsymbol{x}-\boldsymbol{\mu}_g)\right)^2\right\},
$$

where the $\cdot j$ notation means the j^{th} column of the corresponding matrix.

This implies that only $\boldsymbol{\Gamma}_g$ and $\{a_{g1},\ldots,a_{gd_g},b_g\}$ need to be estimated instead of the full $\boldsymbol{\Sigma}_g$. Thus, the number of free parameters in $\boldsymbol{\Sigma}_g$ decreases from $p(p+1)/2$ to $d_gp-d_g(d_g+1)/2$. Moreover, further reduction in free parameters can be achieved if the orientation and/or the shape of the component-wise subspace are constrained to be equal, such as

- Equality of a_{gj} within a component: $a_{gj}=a_g$, $j=1,\ldots,d_g$,
- Equality of a_{gj} across components: $a_{gj}=a_j$, $g=1,\ldots,G$,
- Equality of b_g across components: $b_g=b$, $g=1,\ldots,G$,
- Equality of $\boldsymbol{\Gamma}_g$ across components: $\boldsymbol{\Gamma}_g=\boldsymbol{\Gamma}$, $g=1,\ldots,G$,
- Equal intrinsic dimension across components: $d_g=d$, $g=1,\ldots,G$.

The resulting submodels are denoted in the form of $[a_{gj}b_g\boldsymbol{\Gamma}_gd_g]$, $[a_gb_g\boldsymbol{\Gamma}_gd_g]$, etc., and the full list is available in Bouveyron et al. (2007). The software for the SC-GMM is available as a R package *HDclassif* (Bergé et al., 2012).

The SC-GMM framework has been extended to functional data analysis (Bouveyron et al., 2015), noisy images (Houdard et al., 2018) and a finite mixture of generalized

hyperbolic distributions (Kim and Browne, 2019). This framework is useful in revealing the number of dimensions needed to capture cluster structures of the data set. This means that estimating the component-wise intrinsic dimensions d_g and the selection of cross-component constraints are crucial. The scree test by Cattell (1966) is commonly used to determine the intrinsic dimension d_g. The scree test examines the plot of eigenvalues of $\boldsymbol{\Sigma}_g$ in decreasing order, and seeks for an "elbow" where the slope of the eigenvalue plot flattens out. The issue here is the indeterminacy of the elbow, because the eigenvalue plot rarely exhibits a clear start point of flattening. Moreover, the meaning of original variables may be lost after projection. Therefore, the way d_g and the rotation matrices $\boldsymbol{\Gamma}_g$ are estimated affects heavily the interpretability of the resultant model. Chapter 5 provides a more detailed discussion on intrinsic dimension selection.

- Factor analyzer (Rubin and Thayer, 1982; Ghahramani and Hinton, 1996). In the Gaussian factor analyzer model, a p-dimensional random vector $\boldsymbol{Y}$ is modelled as an affine function of a q-dimensional latent vector $\boldsymbol{X}$ (such that $q < p$) with a p-dimensional additive random error $\boldsymbol{\epsilon}$. This relationship is mathematically represented as

$$\boldsymbol{Y} = \boldsymbol{\mu} + \boldsymbol{\Lambda}\boldsymbol{X} + \boldsymbol{\epsilon},$$

where $\boldsymbol{\Lambda}$ is called the loading matrix, which is of $p \times q$ dimensions. It is assumed that the entries of the latent $\boldsymbol{X}$ are independent to each other, and likewise for the random error $\boldsymbol{\epsilon}$. Mathematically speaking, firstly let $N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the p-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Then, $\boldsymbol{X} \sim N_q(\boldsymbol{0}, \boldsymbol{I}_q)$, and $\boldsymbol{\epsilon} \sim N_p(\boldsymbol{0}, \boldsymbol{\Psi})$, where $\boldsymbol{\Psi}$ is diagonal and $\boldsymbol{X}$ and $\boldsymbol{\epsilon}$ are independent. An attractive feature of this model is the parsimonious modelling of the covariance matrix of the observed vector $\boldsymbol{Y}$. Namely, the covariance of $\boldsymbol{Y}$ under factor analyzer is $\boldsymbol{\Sigma}_{p\times p} = \boldsymbol{\Lambda}_{p\times q}\boldsymbol{\Lambda}'_{p\times q} + \boldsymbol{\Psi}_{p\times p}$, which reduces the number of free parameters of $\boldsymbol{\Sigma}$ from $p(p+1)/2$ to $pq - q(q-1)/2 + p$, assuming that $p > q$. Hence, under the factor analyzer, the number of free parameters in $\boldsymbol{\Sigma}$ is a linear function of p instead of

quadratic. Thus, factor analyzer can reduce the number of free parameters significantly, given that the response variable can be explained by a relatively small number of latent factors ($q << p$). (Ghahramani and Hinton, 1996) introduced a finite mixture of Gaussian factor analyzers, and it has been extended to various non-Gaussian distributions (Tortora et al., 2016; McNicholas et al., 2017; McLachlan et al., 2007; Lin et al., 2016). When modelling with a factor analyzer, the number of factors q must be determined a priori, and it is commonly estimated via BIC like the other frameworks. Moreover, like how the intrinsic dimension d_g changes the number of free parameters in the chosen submodel in subspace clustering, the factor count q changes the dimension of the factor loading $\mathbf{\Lambda}$. If q is too high, then interpreting the entries of $\mathbf{\Lambda}$ is more challenging. In addition, the resultant covariance matrix estimate $\widehat{\mathbf{\Sigma}} = \hat{\mathbf{\Lambda}}\hat{\mathbf{\Lambda}}' + \hat{\mathbf{\Psi}}$ is more likely to be dense as well. For simplicity, consider the $q = 1$ case. The factor loading is a $p \times 1$ vector, and the sparser this vector is, the sparser the estimate $\widehat{\mathbf{\Sigma}}$ will be. Another point of interest in the factor analyzer is the rotation invariance of factor loadings. Given a $q \times q$ dimensional orthogonal matrix $\boldsymbol{Q}$, a factor loading $\mathbf{\Lambda}$ and its rotated version $\mathbf{\Lambda}\boldsymbol{Q}$ yield the same covariance, since

$$\mathbf{\Lambda}\mathbf{\Lambda}' = \mathbf{\Lambda}\boldsymbol{Q}\boldsymbol{Q}'\mathbf{\Lambda}.$$

The rotational non-identifiability of the factor loading with regard to covariance estimation has led to the research in factor rotations for various 'simple' structures for interpretation. Notable criteria include the Varimax by Kaiser (1958), the Quartimax by Ferguson (1954) and the Oblimin by Clarkson and Jennrich (1988).

2.3 Mixture Model Component Merging

Individual components in a finite mixture model are often treated as clusters. While the definition of a cluster is context-dependent, if the component-wise distributions do not accommodate adequately the peculiarities of the data set, the number of components may exceed the number of underlying clusters. Unfortunately, such disparity is difficult to de-

tect a priori. Component merging refers to families of techniques that identify sufficiently similar mixture components and unify their labels without necessarily refitting the whole model. Two such families are modal clustering and component membership probability-based merging.

Modal clustering seeks regions of single dominant mode, where each of such regions may consist of several mixture components. Chacón (2019) introduced mode-merging algorithms for the GMM, and Kim and Browne (2021a) extended the algorithm to the finite mixture of t-distributions. An overview of modal clustering is provided by Chacón (2020). Modal clustering is closely related to the concept of unimodality. Ray and Lindsay (2005) introduced the ridgeline function for the GMM and outlined some conditions under which a pair of Gaussian densities is unimodal.

Merging methods based on component membership probabilities seek groups of components where observations are similarly likely to belong in any one of said components, or the observations are most likely to belong to the said group than others. Hennig (2010) introduced the Directly Estimated Misclassification Probabilities (DEMP) for the GMM, which measures the degree of overlap between components using misclassification probabilities. A robust variant of the DEMP was introduced by Melnykov (2016), called DEMP+. Baudry et al. (2010) introduced an algorithm where components are merged based on an entropy-based criterion. Scrucca (2016) used the log-odds on component membership probabilities and density level sets to merge components. We describe the DEMP+ and the entropy-based criterion below.

- Directly Estimated Misclassification Probabilities Plus (DEMP+) is a mixture component-merging procedure based on the degree of overlap between pairs of components (or component groups). A misclassification probability between two sets of components $\mathcal{G}_1$ and $\mathcal{G}_2$ is defined as

$$q_{\mathcal{G}_1|\mathcal{G}_2} = P\left(\sum_{g\in\mathcal{G}_2}\pi_g f_g(\boldsymbol{X}) < \sum_{k\in\mathcal{G}_1}\pi_k f_k(\boldsymbol{X})\middle|\boldsymbol{X} \text{ from } \mathcal{G}_2\right),$$

and a measure of overlap between $\mathcal{G}_1$ and $\mathcal{G}_2$ is defined as

$$q_{\mathcal{G}_1,\mathcal{G}_2} = q_{\mathcal{G}_1|\mathcal{G}_2} + q_{\mathcal{G}_2|\mathcal{G}_1}. \tag{2.6}$$

If $q_{\mathcal{G}_1,\mathcal{G}_2} > c$ for some pre-determined threshold c (authors suggest $c = 0.1$), then $\mathcal{G}_1$ and $\mathcal{G}_2$ are deemed be sufficiently overlapped, and their labels are merged. The authors compute a sample estimate of $q_{\mathcal{G}_1|\mathcal{G}_2}$ by sampling $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ (N pre-determined) from a mixture distribution consisting of components from $\mathcal{G}_2$ first, then computing

$$\hat{q}_{\mathcal{G}_1|\mathcal{G}_2} = \frac{1}{N} \sum_{i=1}^{N} I\left(\sum_{g \in \mathcal{G}_2} \hat{\pi}_g f_g(\boldsymbol{x}_i) < \sum_{k \in \mathcal{G}_1} \hat{\pi}_k f_k(\boldsymbol{x}_i) \right). \tag{2.7}$$

Finally, we compute $\hat{q}_{\mathcal{G}_1,\mathcal{G}_2} = \hat{q}_{\mathcal{G}_1|\mathcal{G}_2} + \hat{q}_{\mathcal{G}_2|\mathcal{G}_1}$.

- The entropy-based criterion (abbreviated as EntropyMerge hereafter) is motivated as an alternative to both the BIC and the ICL, each of which can over- and under-estimate the number of clusters respectively, per the authors. The procedure begins with a model chosen by the BIC, with G components. Once a mixture model is fitted, the MAP estimates of membership probabilities are used as an initialization: $\{\hat{z}_{i1}^{(1)}, \ldots, \hat{z}_{iG}^{(1)}\}_{i=1,\ldots,n}$. In the first iteration, the pair of components (j, k) that maximizes the following criterion

$$-\sum_{i=1}^{n} \left[\hat{z}_{ij}^{(1)} \log(\hat{z}_{ij}^{(1)}) + \hat{z}_{ik}^{(1)} \log(\hat{z}_{ik}^{(1)}) \right] + \sum_{i=1}^{n} (\hat{z}_{ij}^{(1)} + \hat{z}_{ik}^{(1)}) \log(\hat{z}_{ij}^{(1)} + \hat{z}_{ik}^{(1)}) \tag{2.8}$$

is merged. Let $(j^\star, k^\star)$ denote the merged pair. Once merged, the posterior probabilities get updated to (for $g = 1, \ldots, G-1$)

$$\hat{z}_{ig}^{(2)} = \begin{cases} \hat{z}_{ig}^{(1)} & \text{if } g \notin \{j^\star, k^\star\}, \\ \hat{z}_{ij^\star}^{(1)} + \hat{z}_{ik^\star}^{(1)} & \text{otherwise.} \end{cases}$$

Then the criterion in (2.8) is applied again to select a pair from $1, \ldots, G-1$ to be

merged. Merging is terminated when

$$\frac{\text{Entropy at iteration i} - \text{Entropy at iteration i+1}}{\text{Entropy at iteration 1}} < c, \tag{2.9}$$

where the threshold c is set at 0.05 in this thesis, and the entropy at iteration i is computed as

$$-\sum_{g}\sum_{i=1}^{n} \hat{z}_{ig}^{(i)} \log(\hat{z}_{ig}^{(i)}).$$

Chapter 3

Stiefel Elastic Net: A Novel Penalization for Matrix-variate Parameters

3.1 Introduction

Estimating interpretable model parameters has been a key interest in the recent past, where the parameter estimates are forced to be of smaller magnitude (shrinkage), or to be zero (sparsity). Such regularization allows the user to identify important variables in the model, hence improves model interpretability. There are ample literature on parameter regularization via penalized optimization, including the famed Ridge, LASSO and Elastic Net (Hoerl and Kennard, 1970; Tibshirani, 1996; Zou and Hastie, 2005). While limited, there exists some existing work on regularization of factor loadings as well, where the focus is on sparsity. An early method is the Quartimax rotation by Neuhaus and Wrigley (1954), which rotates the factor loading to find a simpler structure. Adachi and Trendafilov (2018, 2014); Trendafilov and Adachi (2015); Trendafilov et al. (2017) use projection approaches that are not model-based, and Hirose and Yamamoto (2014, 2015) considers some sparsity-inducing penalty functions where a finite mixture of Gaussian factor analyzers is assumed.

However, many sparse estimation techniques on model-based factor analyzers risk degenerate solutions at the cost of sparsity due to rank-deficient factor loading estimates. This can lead to poor model fit and interpretability. To address this issue, we develop a novel method for sparse, yet rank-preserving estimation of component-wise factor loadings in a finite mixture of Gaussian factor analyzers, and explore their theoretical properties. In addition, we extend the existing work on sparse factor analyzer by Hirose and Yamamoto (2015) to a finite mixture of Gaussian factor analyzers for completeness of the literature. We will demonstrate both contributions' performance in real and simulated data settings.

3.2 Methodology

In this section, we present two methods for estimating a sparse factor loading in a finite mixture of Gaussian factor analyzers. The first method is a direct penalization on the component-wise factor loadings $\boldsymbol{\Lambda}_g$, which is an extension of the work on a single component Gaussian factor analyzer by Hirose and Yamamoto (2015). The second and novel method is based on an alternative parametrization of the factor loadings via singular value decomposition.

The model of interest is a G-component finite mixture of Gaussian factor analyzers. Extending the single-component Gaussian factor analyzer outlined section 2.2, let $\boldsymbol{Y}_i$, $\boldsymbol{X}_{ig}$ and $\boldsymbol{\epsilon}_{ig}$ denote the i^{th} observed variable, the i^{th} latent variable from component g and the i^{th} random error variable from component g, respectively. Then, with the latent component membership indicator variable $\boldsymbol{Z}_i$ as defined in section 2.1, the conditional distribution of $\boldsymbol{Y}_i$ given $Z_{ig} = 1$ and that of $\boldsymbol{Y}_i$ given $Z_{ig} = 1$ and $\boldsymbol{x}_{ig}$ (a realization of $\boldsymbol{X}_{ig}$) are given by

$$\begin{aligned}
\boldsymbol{Y}_i | Z_{ig} = 1 &\sim N_p(\boldsymbol{\mu}_g, \boldsymbol{\Lambda}_g \boldsymbol{\Lambda}_g' + \boldsymbol{\Psi}_g), \\
\boldsymbol{Y}_i | Z_{ig} = 1, \boldsymbol{x}_{ig} &\sim N_p(\boldsymbol{\mu}_g + \boldsymbol{\Lambda}_g \boldsymbol{x}_{ig}, \boldsymbol{\Psi}_g),
\end{aligned}$$

where $\boldsymbol{\Lambda}_g$ is the factor loading parameter of component g. The marginal pdf of $\boldsymbol{Y}_i$ at $\boldsymbol{y}_i$ is

given by

$$f(\boldsymbol{y}_i;\boldsymbol{\Theta}) = \sum_{g=1}^{G} \pi_g \phi(\boldsymbol{y}_i;\boldsymbol{\mu}_g, \boldsymbol{\Lambda}_g\boldsymbol{\Lambda}_g' + \boldsymbol{\Psi}_g),$$

where $\boldsymbol{\Theta}$ denotes the set of model parameters. Under this model, a complete-data set consists of $(\boldsymbol{y}_i', \boldsymbol{z}_i', \boldsymbol{x}_{i1}', \ldots, \boldsymbol{x}_{iG}')'$ tuples, and the corresponding complete-data log-likelihood based on n independent observations is

$$l_c(\boldsymbol{\Theta}) = \sum_{i=1}^{n}\sum_{g=1}^{G} Z_{ig}[\log \pi_g + \log \phi(\boldsymbol{y}_i;\boldsymbol{\mu}_g + \boldsymbol{\Lambda}_g\boldsymbol{x}_{ig}, \boldsymbol{\Psi}_g) + \log \phi(\boldsymbol{x}_{ig};\boldsymbol{0}, \boldsymbol{I}_q)],$$

where $\boldsymbol{\Theta}$ denotes the set of model parameters. The methods to be presented in this work can be represented in a penalized complete-data log-likelihood framework given by

$$l_{pen}(\boldsymbol{\Theta}) = l_c(\boldsymbol{\Theta}) - \sum_{g=1}^{G} \rho_g h(\cdot), \tag{3.1}$$

where $h(\cdot)$ denotes the penalty function with appropriate argument, and $\rho_g > 0$ are the component-wise penalty multiplier, which are treated as hyper-parameters.

3.2.1 Alternative Parametrization of Factor Loading

In the following discussion, we drop the component subscript g for notational brevity. An unconstrained direct penalization on the factor loading can lead it to a zero matrix as the penalty multiplier increases. This behaviour can be problematic in both parameter estimation and interpretation. A zero factor loading implies that $\boldsymbol{\Lambda}\boldsymbol{\Lambda}' + \boldsymbol{\Psi} = \boldsymbol{\Psi}$, which is overly restrictive and likely uninformative. Assuming that the number of factors is correctly specified, one would expect some amount of explanatory power from each factor manifesting as non-zero entries. However, even in such cases, unconstrained direct penalization has no built-in mechanism to prevent a degenerate loading estimate. Hence, we develop a penalization method for the factor loading that can estimate a sparse and full-rank factor loading. In addition to the increased interpretability from sparseness, the full-rankness of

the estimate ensures that all q factors contribute to the model.

We begin by observing that the covariance matrix arising from the factor analyzer is identifiable up to orthogonal rotation on the factor loading; see McLachlan and Peel (2004). Now consider the thin singular value decomposition of a p-by-q dimensional factor loading $\mathbf{\Lambda}$

$$\mathbf{\Lambda}_{p\times q} = \mathbf{\Gamma}_{p\times q}\mathbf{\Xi}_{q\times q}\mathbf{\Omega}'_{q\times q},$$

where $\mathbf{\Gamma}$ and $\mathbf{\Omega}$ are p and q-dimensional orthonormal q-frames respectively, and $\mathbf{\Xi}$ is a q-dimensional diagonal matrix. Under the factor analyzer model, the covariance matrix of the observed variable is given as

$$\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{\Psi} = \mathbf{\Gamma}\mathbf{\Xi}^2\mathbf{\Gamma}' + \mathbf{\Psi},$$

as $\mathbf{\Omega}$ is a orthogonal matrix. It is clear that $\mathbf{\Omega}$ vanishes in the formula for $\mathbf{\Sigma}$ under this decomposition. Thus, we constrain $\mathbf{\Omega}$ to be the identity matrix and obtain a $(\mathbf{\Gamma}, \mathbf{\Xi})$ parametrization of $\mathbf{\Lambda}$ while preserving the identifiability of $\mathbf{\Sigma}$. Under this parametrization, we have $\mathbf{\Lambda} = \mathbf{\Gamma}\mathbf{\Xi}$. With respect to the identifiability of $\mathbf{\Lambda}$, there are two types of equivalent constraints as explained in Fokoué and Titterington (2003). One of them constrains $\mathbf{\Lambda}$ such that $\mathbf{\Lambda}'\mathbf{\Lambda}$ is a diagonal matrix. The $(\mathbf{\Gamma}, \mathbf{\Xi})$ parametrization satisfies this constraint:

$$\mathbf{\Lambda}'\mathbf{\Lambda} = \mathbf{\Xi}\mathbf{\Gamma}'\mathbf{\Gamma}\mathbf{\Xi} = \mathbf{\Xi}\boldsymbol{I}\mathbf{\Xi} = \mathbf{\Xi}^2,$$

where $\mathbf{\Gamma}'\mathbf{\Gamma} = \boldsymbol{I}$ by construction and $\mathbf{\Xi}^2$ is diagonal by definition. Therefore, under our alternative parametrization, $\mathbf{\Lambda}$ is identifiable.

3.2.2 Direct Penalization on Factor Loading

The entry-wise penalization on the factor loading is an intuitive way to estimate sparse factor loadings. Hirose and Yamamoto (2015) have contributed to solving this problem by introducing a single-component LASSO-based sparse factor analyzer. We will refer to this direct entry-wise penalization on the factor loading as the PL penalty. The function h_{PL}

for the PL penalty is

$$h_{PL}(\mathbf{\Lambda}) = \sum_{i=1}^{p}\sum_{j=1}^{q} |\mathbf{\Lambda}_{ij}|. \tag{3.2}$$

Extending this penalty to a finite mixture model is straightforward. The penalized complete-data log-likelihood in (3.1) under the PL penalty is

$$l_{PL}(\mathbf{\Theta}) = l_c(\mathbf{\Theta}) - \sum_{g=1}^{G} \rho_g h_{PL}(\mathbf{\Lambda}_g).$$

Optimization of l_{PL} with respect to each $\mathbf{\Lambda}_g$ can be formulated as an iterative least-square-type problem, as will be shown in section 3.2.4.

3.2.3 Stiefel Elastic Net (SEN)

Consider the Stiefel manifold of q vectors over p-dimensional real vector space $V_{p.q}$. The penalty function of interest in this chapter is the row-wise $L_{s,1}$ norm for $s = 1, 2$, which is defined as

$$||\mathbf{\Gamma}||_{s,1} = \sum_{i=1}^{p} \left(\sum_{j=1}^{q} |\mathbf{\Gamma}_{ij}|^s \right)^{1/s}, \tag{3.3}$$

where $\mathbf{\Gamma}_{ij}$ is the ij^{th} element of the matrix $\mathbf{\Gamma}$. The $L_{s,t}$ norm is usually defined on the columns of a matrix, and is used frequently in matrix regularization for structured sparsity; see Yuan and Lin (2006). The $L_{s,1}$ penalty over the Stiefel manifold has several desirable properties for our purpose. We will discuss the theoretical results on $L_{2,1}$ first, followed by that on $L_{1,1}$.

$L_{2,1}$ case

The $L_{2,1}$ norm penalty has an intuitive lower bound over the Stiefel manifold, and that is the column rank of its argument. To show this, we begin with the following lemma.

Lemma 1. *Let* $V_{p,q} = \{\boldsymbol{\Gamma} \in \mathbb{R}^{p\times q} : \boldsymbol{\Gamma}'\boldsymbol{\Gamma} = \boldsymbol{I}_q\}$ *where* $p \geq q$. *Denote the* i^{th} *row of a matrix* $\boldsymbol{\Gamma}$ *as* $\boldsymbol{\Gamma}_{i\cdot}$. *For any* $\boldsymbol{\Gamma} \in V_{p,q}$, $||\boldsymbol{\Gamma}_{i\cdot}||_2^2 \leq 1$ *for all* $i = 1, \ldots, p$.

Proof. Suppose, on the contrary, that there exists a row $k \in \{1, \ldots, p\}$ such that $||\boldsymbol{\Gamma}_{k\cdot}||_2^2 > 1$. Since $\boldsymbol{\Gamma}$ is an orthonormal q-frame, it is always possible to construct a p-by-p orthogonal matrix by attaching $p - q$ linearly independent unit column vectors, denoted by $\boldsymbol{U}$, that are pairwise-orthogonal to every column vector in $\boldsymbol{\Gamma}$. Then, this new orthogonal matrix $\boldsymbol{W} = [\boldsymbol{\Gamma}, \boldsymbol{U}]$ is such that $\boldsymbol{W}'\boldsymbol{W} = \boldsymbol{I}$, which implies

$$||\boldsymbol{\Gamma}_{k\cdot}||_2^2 + ||\boldsymbol{U}_{k\cdot}||_2^2 = 1,$$

However, this implies $||\boldsymbol{\Gamma}_{k\cdot}||_2^2 \leq 1$, which is a contradiction. □

Proposition 1. *The minimum value of the* $L_{2,1}$ *norm over* $V_{p,q}$ *is* q.

Proof. The column-wise orthogonality constraint implies $\text{tr}(\boldsymbol{\Gamma}'\boldsymbol{\Gamma}) = q$. Moreover, by the cyclic property of trace, we have

$$q = \text{tr}(\boldsymbol{\Gamma}'\boldsymbol{\Gamma}) = \text{tr}(\boldsymbol{\Gamma}\boldsymbol{\Gamma}') = \sum_{i=1}^{p} ||\boldsymbol{\Gamma}_{i\cdot}||_2^2.$$

If $p = q$, then $\boldsymbol{\Gamma}$ is orthogonal in $\mathbb{R}^{q\times q}$, so $\text{tr}(\boldsymbol{\Gamma}\boldsymbol{\Gamma}') = q$. Thus, assume $p > q$ without loss of generality. Lemma 1 tells us that $||\boldsymbol{\Gamma}_{i\cdot}||_2^2 \leq 1$ for every i. Hence, $||\boldsymbol{\Gamma}_{i\cdot}||_2^2 \leq ||\boldsymbol{\Gamma}_{i\cdot}||_2$. This implies

$$||\boldsymbol{\Gamma}||_{2,1} = \sum_{i=1}^{p} ||\boldsymbol{\Gamma}_{i\cdot}||_2 \geq \sum_{i=1}^{p} ||\boldsymbol{\Gamma}_{i\cdot}||_2^2 = q. \tag{3.4}$$

□

The following corollary characterizes a minimizer of the $L_{2,1}$ norm penalty.

Corollary 1. *Any minimizer of the* $L_{2,1}$ *norm penalty over* $V_{p,q}$ *has exactly* q *rows that form a* q*-by-*q *orthogonal matrix, and the remaining* $p - q$ *rows are zero vectors.*

Proof. Suppose that $\mathbf{\Gamma} \in V_{p,q}$ minimizes the $L_{2,1}$ penalty over $V_{p,q}$. Inequality (3.4) implies that

$$\sum_{i=1}^{p} \left(||\mathbf{\Gamma}_{i\cdot}||_2 - ||\mathbf{\Gamma}_{i\cdot}||_2^2 \right) = 0.$$

Since we have $||\mathbf{\Gamma}_{i\cdot}||_2 \geq ||\mathbf{\Gamma}_{i\cdot}||_2^2$ and equality is achieved if and only if $||\mathbf{\Gamma}_{i\cdot}||_2 = 1$ or $\mathbf{\Gamma}_{i\cdot} = \mathbf{0}$, there must be exactly q rows with unit vectors and the remaining $p-q$ rows must be zero vectors. □

Remark: A q-frame of signed standard basis vectors in $\mathbb{R}^p$ minimizes the $L_{2,1}$ norm penalty over $V_{p,q}$ for $p \geq q$.

$L_{1,1}$ case

The $L_{1,1}$ penalty enjoys the same lower bound as that of $L_{2,1}$, as shown below.

Proposition 2. *Consider the space $V_{p,q}$ where $p \geq q$. The minimum value of the $L_{1,1}$ norm over $V_{p,q}$ is q.*

Proof. From proposition 1 and by vector norm property, we have

$$q \leq \sum_{i=1}^{p} ||\mathbf{\Gamma}_{i\cdot}||_2 \leq \sum_{i=1}^{p} ||\mathbf{\Gamma}_{i\cdot}||_1 = \sum_{j=1}^{q} ||\mathbf{\Gamma}_{\cdot j}||_1 = ||\mathbf{\Gamma}||_{1,1}. \tag{3.5}$$

Clearly, the matrix $\boldsymbol{A} = [\boldsymbol{I}_q, \mathbf{0}_{q\times(p-q)}]'$ achieves equality for the lower bound given in (3.5), so the bound of q is attainable. □

However, the $L_{1,1}$ penalizes the matrix more aggressively, which results in a finite number of feasible minimizers. The following proposition shows that a minimizer takes the form of extreme points on the manifold.

Proposition 3. *Let $V_{p,q}$ be defined as earlier and assume $p \geq q$. The only minimizer of $L_{1,1}$ norm penalty over $V_{p,q}$ is a q-frame of signed standard basis vectors in $\mathbb{R}^p$.*

Proof. Let $\mathbf{\Gamma}$ be a q-frame of signed standard basis vectors in $\mathbb{R}^p$. Then clearly $||\mathbf{\Gamma}||_{1,1} = q$. For the converse, suppose that there exists $\mathbf{\Gamma} \in V_{p,q}$ such that $||\mathbf{\Gamma}||_{1,1} = q$. Then it follows that

$$\sum_{i=1}^{p} ||\mathbf{\Gamma}_{i\cdot}||_1 = \sum_{j=1}^{q} ||\mathbf{\Gamma}_{\cdot j}||_1 \geq \sum_{j=1}^{q} ||\mathbf{\Gamma}_{\cdot j}||_2 \geq \sum_{j=1}^{q} ||\mathbf{\Gamma}_{\cdot}||_2^2 = \text{tr}(\mathbf{\Gamma}'\mathbf{\Gamma}) = q,$$

where the first equality occurs because

$$\sum_{i=1}^{p} ||\mathbf{\Gamma}_{i\cdot}||_1 = \sum_{i=1}^{p}\sum_{j=1}^{q} |\mathbf{\Gamma}_{ij}| = \sum_{j=1}^{q}\sum_{i=1}^{p} |\mathbf{\Gamma}_{ij}| = \sum_{j=1}^{q} ||\mathbf{\Gamma}_{\cdot j}||_1.$$

The leftmost inequality between 1-norm and 2-norm follows from the property of vector norms, and the middle inequality between 2-norm and squared 2-norm follows from (3.4). Hence, we have $||\mathbf{\Gamma}_{\cdot j}||_1 = ||\mathbf{\Gamma}_{\cdot j}||_2$ for every column in $\mathbf{\Gamma}$, which occurs if and only if $\mathbf{\Gamma}_{\cdot j}$ is a zero vector or is a signed elementary basis vector for every j. □

Proposition 3 implies that the $L_{1,1}$ allocates exactly one latent factor to each of q dimensions, and it estimates remaining dimensions as noise. This is a more aggressive penalization than the $L_{2,1}$ norm.

Defining the Stiefel Elastic Net

Finally, we introduce the convex combination of the $L_{2,1}$ and $L_{1.1}$ penalties, which contains each of the two as special cases. We name this penalty as Stiefel Elastic Net, abbreviated as the SEN:

$$h_{SEN}(\mathbf{\Gamma}) = \alpha||\mathbf{\Gamma}||_{1,1} + (1-\alpha)||\mathbf{\Gamma}||_{2,1}, \tag{3.6}$$

where $\alpha \in [0, 1]$ is the hyper-parameter for mixing portion between the two penalties, and $\mathbf{\Gamma} \in V_{p,q}$. The SEN inherits the same lower bound as that of $L_{1,1}$ and $L_{2,1}$ norms. Moreover, it has a finite set of minimizers if $\alpha > 0$ due to the inclusion of $L_{1,1}$ component.

The following proposition characterizes a minimizer of the SEN.

Proposition 4. *Let $V_{p,q}$ be defined as earlier and assume $p \geq q$. The only minimizers of the SEN with $\alpha > 0$ over $V_{p,q}$ are q-frames of signed standard basis vectors in $\mathbb{R}^p$.*

Proof. Proposition 1 tells us that a minimizer of the $L_{2,1}$ penalty takes the form of a row-wise permutation of a q-by-q orthogonal matrix and $(p-q)$-by-q zero matrix. Proposition 3 tells us that a minimizer of $L_{1,1}$ penalty is a q-frame of signed standard basis vectors in $\mathbb{R}^p$. Since the SEN is minimized if and only if each of $L_{1,1}$ and $L_{2,1}$ is minimized, the set of minimizers of SEN is the intersection between that of $L_{1,1}$ and $L_{2,1}$. Finally, we observe that the set of minimizers for $L_{1,1}$ penalty is a strict subset of that for $L_{2,1}$. □

The SEN shares similarity with the Elastic Net by Zou and Hastie (2005) due to its formula, but SEN generalizes the Elastic Net to a constrained space of matrices. In the remainder of this proposal, for ease of reference, SEN with α fixed at 0 and 1 will be denoted as Stiefel penalty 1 and 2 respectively. Their function notation will be h_{SP1} and h_{SP2} respectively.

3.2.4 Parameter Estimation

Parameter updates are based on the Alternating Expectation-Conditional Maximization (AECM) algorithm by Meng and Van Dyk (1997), in conjunction with a suitable penalty-based update for the factor loading $\mathbf{\Lambda}_g$, or the orthonormal q-frame $\mathbf{\Gamma}_g$, depending on the parametrization. The AECM algorithm is used as it enjoys a faster convergence than the original EM algorithm. This is to compensate for the more complicated update for $\mathbf{\Lambda}_g$.

Recall that the penalized complete-data log-likelihood function under a G-component mixture model is given by

$$l_{pen}(\mathbf{\Theta}) = l_c(\mathbf{\Theta}) - \sum_{g=1}^{G} \rho_g h_{pen}(\cdot),$$

where $pen \in \{PL,\ SP1,\ SP2,\ SEN\}$ with an appropriate argument in place of $(\cdot)$. $\rho_g > 0$ are the component-wise penalty multiplier hyper-parameters. The SEN contains an

additional mixing coefficient α_g, which is also a hyper-parameter. In the AECM algorithm, we will update $(\pi_g, \boldsymbol{\mu}_g)$ first, then $\boldsymbol{\Lambda}_g$, and finally $\boldsymbol{\Psi}_g$. At each iteration of the algorithm, the existing and updated parameter estimates will be superscripted with (t) and $(t+1)$ respectively.

In the first stage of the AECM algorithm, we update the component-wise location and mixing proportion parameters, $\boldsymbol{\mu}_g$ and π_g. At this stage, the complete-data set is made of $(\boldsymbol{y}_i, \boldsymbol{z}_i)$ pairs the l_c function is equal to

$$\begin{aligned} & l_c(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_G, \pi_1, \ldots, \pi_G) \\ & \quad = \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} \left\{ \log \pi_g - \frac{1}{2} \text{tr} \left[\left(\boldsymbol{\Lambda}_g \boldsymbol{\Lambda}_g' + \boldsymbol{\Psi}_g \right)^{-1} (\boldsymbol{y}_i - \boldsymbol{\mu}_g)(\boldsymbol{y}_i - \boldsymbol{\mu}_g)' \right] \right\} + \text{const}, \end{aligned}$$

where 'const' represents all additive constants. Treating z_{ig}s as missing, the conditional expectation of l_c at iteration t given $\boldsymbol{y}_i$s is equal to

$$\begin{aligned} & Q(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_G, \pi_1, \ldots, \pi_G | \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_G^{(t)}, \pi_1^{(t)}, \ldots, \pi_G^{(t)}) \\ & \quad = \sum_{g=1}^{G} \left\{ n_g^{(t)} \log \pi_g - \frac{1}{2} \text{tr} \left[\left(\boldsymbol{\Lambda}_g \boldsymbol{\Lambda}_g' + \boldsymbol{\Psi} \right)^{-1} \sum_{i=1}^{n} z_{ig}^{(t)} (\boldsymbol{y}_i - \boldsymbol{\mu}_g)(\boldsymbol{y}_i - \boldsymbol{\mu}_g)' \right] \right\} + \text{const}, \end{aligned}$$

where $z_{ig}^{(t)} = \dfrac{\pi_g^{(t)} \phi \left(\boldsymbol{y}_i; \boldsymbol{\mu}_g^{(t)}, \boldsymbol{\Lambda}_g^{(t)}, \boldsymbol{\Psi}_g^{(t)} \right)}{\sum_{h=1}^{G} \pi_h^{(t)} \phi \left(\boldsymbol{y}_i; \boldsymbol{\mu}_h^{(t)}, \boldsymbol{\Lambda}_h^{(t)}, \boldsymbol{\Psi}_h^{(t)} \right)}$ and $n_g^{(t)} = \sum_{i=1}^{n} z_{ig}^{(t)}$. Upon differentiation with respect to each of $\boldsymbol{\mu}_g$ and π_g, we obtain the updates

$$\boldsymbol{\mu}_g^{(t+1)} = \frac{\sum_{i=1}^{n} \hat{z}_{ig} \boldsymbol{y}_i}{\sum_{i=1}^{n} \hat{z}_{ig}}, \qquad \text{and } \pi_g^{(t+1)} = \frac{\hat{n}_g}{n}.$$

In the next stage of the AECM algorithm, we update the component-wise factor loading and noise covariance $\boldsymbol{\Lambda}_g$ and $\boldsymbol{\Psi}_g$, while incorporating the updates from the previous stage. Here, the complete-data set is made of $(\boldsymbol{y}_i', \boldsymbol{z}_i', \boldsymbol{x}_{i1}', \ldots, \boldsymbol{x}_{iG}')'$ tuples, as $\boldsymbol{\Lambda}_g$ and $\boldsymbol{\Psi}_g$ need to

be separated. The complete-data log-likelihood is equal to

$$
\begin{aligned}
&l_c\left(\boldsymbol{\Lambda}_1, \ldots, \boldsymbol{\Lambda}_G, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_G\right) \\
&\quad \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} \Bigg\{ -\frac{1}{2} \log \left|\boldsymbol{\Psi}_g\right| - \frac{1}{2} \operatorname{tr}\left[\boldsymbol{\Psi}_g^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu}_g^{(t+1)} - \boldsymbol{\Lambda}_g \boldsymbol{x}_{ig})(\boldsymbol{y}_i - \boldsymbol{\mu}_g^{(t+1)} - \boldsymbol{\Lambda}_g \boldsymbol{x}_{ig})'\right] \Bigg\} + \text{const},
\end{aligned}
$$

and its conditional expectation given $\boldsymbol{y}_i$s is equal to

$$
\begin{aligned}
&Q(\boldsymbol{\Lambda}_1, \ldots, \boldsymbol{\Lambda}_G, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_G | \boldsymbol{\Lambda}_1^{(t)}, \ldots, \boldsymbol{\Lambda}_G^{(t)}, \boldsymbol{\Psi}_1^{(t)}, \ldots, \boldsymbol{\Psi}_G^{(t)}) \\
&\quad = \sum_{g=1}^{G} \left\{ \frac{n_g^{(t)}}{2} \log \left|\boldsymbol{\Psi}_g\right| - \frac{1}{2} \operatorname{tr}\left(\boldsymbol{\Psi}_g^{-1} \boldsymbol{S}_g\right) + \operatorname{tr}\left(\boldsymbol{\Psi}_g^{-1} \boldsymbol{\Lambda}_g \boldsymbol{\beta}_g^{(t)} \boldsymbol{S}_g\right) - \frac{1}{2} \operatorname{tr}\left(\boldsymbol{\Lambda}_g' \boldsymbol{\Psi}_g^{-1} \boldsymbol{\Lambda}_g \boldsymbol{\Phi}_g\right) \right\} + \text{const},
\end{aligned} \tag{3.7}
$$

where $z_{ig}^{(t)}$ and $n_g^{(t)}$ are now computed using $\boldsymbol{\mu}_g^{(t+1)}$ and $\pi_g^{(t+1)}$, and

$$
\begin{aligned}
\boldsymbol{S}_g &= \hat{n}_g^{-1} \sum_{i=1}^{n} \hat{z}_{ig} (\boldsymbol{y}_i - \boldsymbol{\mu}_g^{(t+1)})(\boldsymbol{y}_i - \boldsymbol{\mu}_g^{(t+1)})', \\
\boldsymbol{\beta}_g^{(t)} &= \boldsymbol{\Lambda}_g^{(t)'} \left(\boldsymbol{\Lambda}_g^{(t)} \boldsymbol{\Lambda}_g^{(t)'} + \boldsymbol{\Psi}_g^{(t)}\right)^{-1}, \\
\boldsymbol{\Phi}_g &= \boldsymbol{I}_q - \boldsymbol{\beta}_g^{(t)} \boldsymbol{\Lambda}_g^{(t)} + \boldsymbol{\beta}_g^{(t)} \boldsymbol{S}_g \boldsymbol{\beta}_g^{(t)'}.
\end{aligned}
$$

The specific update formulae for $\boldsymbol{\Lambda}_g$ and $\boldsymbol{\Phi}_g$ differ based on the penalty function. We present the update for the PL first, then the SEN.

Update for PL

When applying the PL, We follow the procedure given in Hirose and Yamamoto (2015) and use the co-ordinate descent. The expected complete-data log-likelihood is written as a quadratic function of $\boldsymbol{\Lambda}_g$, and the entry-wise update for $\boldsymbol{\Lambda}_g$ is the solution to the following

objective function

$$\underset{\boldsymbol{\Lambda}_{ij}}{\operatorname{argmin}} \frac{1}{2}\left(\boldsymbol{\Lambda}_{ij} - c_{ij}\right)^2 + \frac{\rho_g \left[\boldsymbol{\Psi}_g^{(t)}\right]_{ii}}{\left[\boldsymbol{\Phi}_g\right]_{jj}} |\boldsymbol{\Lambda}_{ij}|,$$

where $c_{ij} = \dfrac{\left[\boldsymbol{\beta}_g^{(t)}\right]'_{j\cdot} \left[\boldsymbol{S}_g\right]_{\cdot i} - \sum_{k \neq j} \left[\boldsymbol{\Phi}_g\right]_{kj} \left[\boldsymbol{\Lambda}_g^{(t)}\right]_{ik}}{\left[\boldsymbol{\Phi}_g\right]_{jj}}$, $\left[\boldsymbol{A}\right]_{ij}$ denotes the ij^{th} element of a matrix $\boldsymbol{A}$, $\left[\boldsymbol{A}\right]_{i\cdot}$ denotes the i^{th} row of $\boldsymbol{A}$ and $\left[\boldsymbol{A}\right]_{\cdot j}$ denotes the j^{th} column of $\boldsymbol{A}$. The solution is given by

$$\left[\boldsymbol{\Lambda}_g^{(t+1)}\right]_{ij} = sgn(c_{ij}) \cdot \kappa \left(|c_{ij}| - \frac{\rho_g \left[\boldsymbol{\Psi}_g^{(t)}\right]_{ii}}{\left[\boldsymbol{\Phi}_g\right]_{jj}} \right),$$

where the function $\kappa(z)$ is defined as

$$\kappa(z) = \begin{cases} z & \text{if } z > 0, \\ 0 & \text{otherwise.} \end{cases}$$

The updated estimate replaces $\left[\boldsymbol{\Lambda}_g^{(t)}\right]_{ij}$ in $\boldsymbol{\Lambda}_g^{(t)}$, and the descent on the next entry begins. In this work, the entries are searched in row-major order.

3.2.5 Update for SEN

With the SEN, we update $\boldsymbol{\Gamma}_g$ and $\boldsymbol{\Xi}_g$ separately, then estimate $\boldsymbol{\Lambda}_g = \boldsymbol{\Gamma}_g \boldsymbol{\Xi}_g$. The diagonal matrix $\boldsymbol{\Xi}_g$ is updated first, followed by $\boldsymbol{\Gamma}_g$ with a Minorize-Maximization update based on Browne and McNicholas (2014). The expected complete-data log-likelihood with respect

to $\boldsymbol{\Gamma}_g$s and $\boldsymbol{\Xi}_g$s can be written as

$$Q(\boldsymbol{\Gamma}_1, \ldots, \boldsymbol{\Gamma}_G, \boldsymbol{\Xi}_1, \ldots, \boldsymbol{\Xi}_G | \boldsymbol{\Gamma}_1^{(t)}, \ldots, \boldsymbol{\Gamma}_G^{(t)}, \boldsymbol{\Xi}_1^{(t)}, \ldots, \boldsymbol{\Xi}_G^{(t)})$$
$$= \sum_{g=1}^{G} \left\{ \text{tr}\left(\boldsymbol{\Xi}_g \boldsymbol{\beta}_g^{(t)} \boldsymbol{S}_g \left(\boldsymbol{\Psi}_g^{(t)} \right)^{-1} \boldsymbol{\Gamma}_g \right) - \frac{1}{2} \text{tr}\left(\left(\boldsymbol{\Psi}_g^{(t)} \right)^{-1} \boldsymbol{\Gamma}_g \boldsymbol{\Xi}_g \boldsymbol{\Phi}_g \boldsymbol{\Xi}_g \boldsymbol{\Gamma}_g' \right) \right\} + \text{const.} \tag{3.8}$$

The following matrix identity from Horn and Johnson (2012) can be applied to the second trace term:

$$\text{tr}\left(\boldsymbol{\Xi}_g \boldsymbol{\Phi}_g \boldsymbol{\Xi}_g \boldsymbol{M}_g \right) = \text{diag}\left(\boldsymbol{\Xi}_g \right)' \left(\boldsymbol{\Phi}_g \odot \boldsymbol{M}_g \right) \text{diag}\left(\boldsymbol{\Xi}_g \right),$$

where $\boldsymbol{M}_g = \boldsymbol{\Gamma}_g^{(t)'} \left(\boldsymbol{\Psi}_g^{(t)} \right)^{-1} \boldsymbol{\Gamma}_g^{(t)}$, which lets us re-write the summands in equation (3.8) as

$$\text{vecdiag}\left(\boldsymbol{N}_g \right)' \text{diag}\left(\boldsymbol{\Xi}_g \right) - \frac{1}{2} \text{diag}\left(\boldsymbol{\Xi}_g \right)' \left(\boldsymbol{\Phi}_g \odot \boldsymbol{M}_g \right) \text{diag}\left(\boldsymbol{\Xi}_g \right),$$

where $\boldsymbol{N}_g = \boldsymbol{\beta}_g^{(t)} \boldsymbol{S}_g \left(\boldsymbol{\Psi}_g^{(t)} \right)^{-1} \boldsymbol{\Gamma}_g^{(t)}$, and 'vecdiag' denotes a vector consisting of the diagonal entries of the matrix argument within. This is a quadratic form in terms of diag($\boldsymbol{\Xi}_g$), and upon differentiation, we obtain

$$\text{diag}\left(\boldsymbol{\Xi}_g^{(t+1)} \right) = \left(\boldsymbol{\Phi}_g \odot \boldsymbol{M}_g \right)^{-1} \text{vecdiag}\left(\boldsymbol{N}_g \right).$$

For $\boldsymbol{\Gamma}_g$, we compute the updates for SP1 and SP2 each, then assemble them to obtain the SEN update. While outlining the component-wise updates via SP1 and SP2, the component subscript g will be dropped for notational brevity.

The MM Algorithm

The MM (Majorize-Minimization or Minorize-Maximization) algorithm, popularized by Hunter and Lange (2000), is an indirect optimization approach to an otherwise-challenging functions through so-called majorizer or minorizer, depending on the objective. A majorizer

g of a function f is a surrogate function with the two properties: $g\left(x^{(t)}|x^{(t)}\right) = f\left(x^{(t)}\right)$ and $g(x|x^{(t)}) \geq f(x)$ for all x, where $x^{(t)}$ denotes the current position of the algorithm. For example, a quadratic majorizer of absolute value $|x|$ is given by de Leeuw and Lange (2009).

$$g(x|x^{(t)}) = \frac{x^2}{2\sqrt{x^{(t)2} + \epsilon}} + \frac{|x^{(t)}|}{2},$$

where ϵ is a small positive constant added as a computational provision to avoid division by zero, and $x^{(t)}$ is the t-th iterative estimate of the argument x.

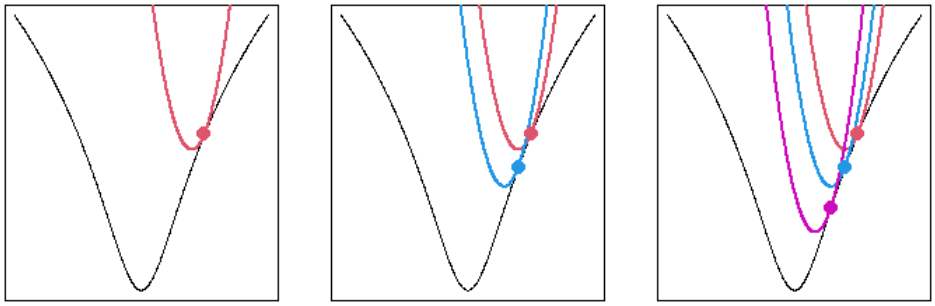

Figure 3.1: An illustration of MM algorithm. Black curve is the objective $f(x)$, and the red, blue, and magenta curves are the majorizers at iterations t, $t+1$ and $t+2$. We see that the the majorizer's minimum approaches that of $f(x)$.

MM Optimization on Stiefel Manifold

Matrix optimization problems in statistics frequently involve the minimization of a function of the form

$$\min_{\boldsymbol{\Gamma}} f(\boldsymbol{\Gamma}) = \min_{\boldsymbol{\Gamma}} \operatorname{tr}(\boldsymbol{A}\boldsymbol{\Gamma}) + \sum_{r=1}^{R} \operatorname{tr}(\boldsymbol{B}_r\boldsymbol{\Gamma}\boldsymbol{C}_r\boldsymbol{\Gamma}'), \tag{3.9}$$

for arbitrary matrices of matching dimensions $\boldsymbol{A}$, $\boldsymbol{B}_r$ and $\boldsymbol{C}_r$, for $r = 1, \ldots, R$, and the argument $\boldsymbol{\Gamma}$ confined to the Stiefel manifold of q vectors over p-dimensional real vector

space, denoted as

$$V_{p,q} = \left\{ \boldsymbol{\Gamma} \in \mathbb{R}^{p \times q} : \boldsymbol{\Gamma}'\boldsymbol{\Gamma} = \boldsymbol{I}_q \right\}. \tag{3.10}$$

This manifold is the space of p-by-q matrices consisting of q orthonormal columns, known as orthonormal q-frames. Optimization on Stiefel manifold is difficult in general, and the general-purpose algorithms are complicated. Fortunately, for the above-mentioned trace-based objective function, works from Browne and McNicholas (2014); Kiers (2002) allow an iterative update based on the MM algorithm. If we assume that $\boldsymbol{B}_r$ are positive definite and $\boldsymbol{C}_r$ are diagonal with positive diagonal entries for all $r = 1, \ldots, R$, then the trace minimization problem in (3.9), admits a majorizer over the Stiefel manifold from Kiers (2002):

$$f(\boldsymbol{\Gamma}) \leq \operatorname{tr}(\boldsymbol{F}_{(t)}\boldsymbol{\Gamma}) + a, \tag{3.11}$$

where $\boldsymbol{F}_{(t)} = \boldsymbol{A} + \sum_{r=1}^{R} \left(\boldsymbol{B}_r \boldsymbol{\Gamma}'_{(t)} \boldsymbol{C}_r - c_r^* \boldsymbol{B}_r \boldsymbol{\Gamma}'_{(t)} \right)$ with $\boldsymbol{\Gamma}_{(t)}$ being the current position of $\boldsymbol{\Gamma}$, c_r^* is the largest eigenvalue of $\boldsymbol{C}_r$, and a is a constant independent of $\boldsymbol{\Gamma}$. Then, the solution to (3.11) is

$$\boldsymbol{\Gamma}_{(t+1)} = \boldsymbol{Q}_{(t)} \boldsymbol{P}'_{(t)}, \tag{3.12}$$

where $\boldsymbol{F}_{(t)} = \boldsymbol{P}_{(t)} \boldsymbol{D}_{(t)} \boldsymbol{Q}'_{(t)}$ is the singular value decomposition of $\boldsymbol{F}_{(t)}$.

MM Update for SP1

For the SP1, we obtain the following majorizer by applying the approximation formula for the absolute value and its sharp quadratic majorization from de Leeuw and Lange (2009); Ramirez et al. (2014).

$$\sum_{i=1}^{p} \sum_{j=1}^{q} |\boldsymbol{\Gamma}_{ij}| \leq \sum_{i=1}^{p} \sum_{j=1}^{q} \left(\frac{\boldsymbol{\Gamma}_{ij}^2}{2\sqrt{|\boldsymbol{\Gamma}_{ij}^{(t)}|^2 + \epsilon}} + \frac{\sqrt{|\boldsymbol{\Gamma}_{ij}^{(t)}|^2 + \epsilon}}{2} \right),$$

and similar to Hunter and Lange (2000), a small perturbation constant $\epsilon > 0$ for computational accommodation of absolute value around zero. In Hunter and Lange (2000), ϵ is set at $1/5$, and in this work, ϵ is set at 10^{-6} for an increased accuracy of approximation. The majorizer from equation (3.13) admits the following trace form

$$\begin{aligned}
&\sum_{i=1}^{p}\sum_{j=1}^{q}\left(\frac{\boldsymbol{\Gamma}_{ij}^2}{2\sqrt{|\boldsymbol{\Gamma}_{ij}^{(t)}|^2+\epsilon}}+\frac{\sqrt{|\boldsymbol{\Gamma}_{ij}^{(t)}|^2+\epsilon}}{2}\right)\\
&=\sum_{j=1}^{q}\operatorname{tr}\left(\boldsymbol{A}_j\boldsymbol{\Gamma}\boldsymbol{e}_j\boldsymbol{e}_j'\boldsymbol{\Gamma}'\right)+c\\
&\leq \operatorname{tr}\left(\boldsymbol{K}\boldsymbol{\Gamma}\right)+\text{const},
\end{aligned} \tag{3.13}$$

where $\boldsymbol{e}_j$ is the j^{th} elementary basis vector, $\boldsymbol{A}_j = \operatorname{diag}\left(2\sqrt{|\boldsymbol{\Gamma}_{1j}^{(t)}|^2+\epsilon},\ldots,2\sqrt{|\boldsymbol{\Gamma}_{pj}^{(t)}|^2+\epsilon}\right)^{-1}$ and $\boldsymbol{K} = \sum_{j=1}^{q}\left[\boldsymbol{e}_j\boldsymbol{e}_j'\left(\boldsymbol{\Gamma}^{(t)'}\right)\boldsymbol{A}_j - \max\left(\boldsymbol{A}_j\right)\boldsymbol{e}_j\boldsymbol{e}_j'\left(\boldsymbol{\Gamma}^{(t)'}\right)\right]$.

MM Update for SP2

For the SP2, we begin with the following row-wise majorizer similar to Nie et al. (2010), where

$$||\boldsymbol{\Gamma}_{i\cdot}||_2 \leq \frac{||\boldsymbol{\Gamma}_{i\cdot}||_2^2}{2\sqrt{||\boldsymbol{\Gamma}_{i\cdot}^{(t)}||_2^2+\epsilon}}+\frac{\sqrt{||\boldsymbol{\Gamma}_{i\cdot}^{(t)}||_2^2+\epsilon}}{2}.$$

With the above, h_{SP2} obtains the following majorizer per Browne and McNicholas (2014)

$$||\boldsymbol{\Gamma}||_{2,1} \leq \operatorname{tr}\left(\boldsymbol{W}\boldsymbol{\Gamma}\boldsymbol{\Gamma}'\right)+\text{const} \leq \operatorname{tr}\left(\boldsymbol{G}\boldsymbol{\Gamma}\right)+\text{const}, \tag{3.14}$$

where
$\boldsymbol{W} = \operatorname{diag}\left(2\sqrt{||\boldsymbol{\Gamma}_{1\cdot}^{(t)}||_2^2+\epsilon},\ldots,2\sqrt{||\boldsymbol{\Gamma}_{p\cdot}^{(t)}||_2^2+\epsilon}\right)^{-1}$ and $\boldsymbol{G} = \boldsymbol{\Gamma}^{(t)'}\boldsymbol{W} - \max(\boldsymbol{W})\boldsymbol{\Gamma}^{(t)'}$.

MM update for SEN

Majorization of h_{SEN} is straightforward, as we already have majorizers for h_{SP1} and h_{SP2}:

$$SEN(\boldsymbol{\Gamma}_g) \leq \operatorname{tr}\left[(\alpha_g \boldsymbol{K}_g + (1-\alpha_g)\boldsymbol{G}_g)\,\boldsymbol{\Gamma}_g\right] + c. \tag{3.15}$$

The final step in updating $\boldsymbol{\Lambda}_g$ is the double minorization of the expectation in (3.8), minus the penalty function. The change from majorization to minorization is a direct consequence of multiplying the penalty function by -1. The conditional expectation in equation (3.8) is minorized by $\operatorname{tr}(\boldsymbol{F}_g\boldsymbol{\Gamma}_g) + c$, where

$$\begin{aligned}
\boldsymbol{A}_g &= \boldsymbol{\Xi}_g^{(t+1)}\boldsymbol{\beta}_g^{(t)}\boldsymbol{S}_g\left(\boldsymbol{\Psi}_g^{(t)}\right)^{-1}, \\
\boldsymbol{B}_g &= \boldsymbol{\Xi}_g^{(t+1)}\boldsymbol{\Phi}_g\boldsymbol{\Xi}_g^{(t+1)} \\
\boldsymbol{F}_g &= \boldsymbol{A}_g - \frac{1}{2}\left[\boldsymbol{B}_g\boldsymbol{\Gamma}_g^{(t)'}\left(\boldsymbol{\Psi}_g^{(t)}\right)^{-1} - \max\left\{\left(\boldsymbol{\Psi}_g^{(t)}\right)^{-1}\right\}\boldsymbol{B}_g\boldsymbol{\Gamma}_g^{(t)'}\right].
\end{aligned}$$

Hence, the double-minorized penalized expectation is given by

$$\operatorname{tr}\left[(\boldsymbol{F}_g - \rho_g\boldsymbol{H}_g)\,\boldsymbol{\Gamma}_g\right] + \text{const},$$

where

$$\boldsymbol{H}_g = \begin{cases} \boldsymbol{K}_g & \text{if } SP1, \\ \boldsymbol{G}_g & \text{if } SP2, \\ \alpha_g\boldsymbol{K}_g + (1-\alpha_g)\boldsymbol{G}_g & \text{if } SEN. \end{cases}$$

The updated estimate is $\boldsymbol{\Gamma}_g^{(t+1)} = \boldsymbol{R}_g\boldsymbol{P}_g'$, where $\boldsymbol{P}_g\boldsymbol{D}_g\boldsymbol{R}_g'$ is the singular value decomposition of $\boldsymbol{F}_g - \rho_g\boldsymbol{H}_g$.

After computing $\boldsymbol{\Lambda}_g^{(t+1)}$, $\boldsymbol{\Psi}_g^{(t+1)}$ can be obtained by differentiating with respect to itself and applying the diagonal matrix constraint:

$$\boldsymbol{\Psi}_g^{(t+1} = \operatorname{diag}\left(\boldsymbol{S}_g - 2\boldsymbol{S}_g\boldsymbol{\beta}_g^{(t)'}\boldsymbol{\Lambda}_g^{(t+1)'} + \boldsymbol{\Lambda}_g^{(t+1)}\boldsymbol{\Phi}_g\boldsymbol{\Lambda}_g^{(t+1)'}\right).$$

3.2.6 Computational Aspects

Care must be taken when counting the number of free parameters. It is known that the factor loading $\mathbf{\Lambda}_{p\times q}$ contains $pq - q(q-1)/2$ free parameters, as in McNicholas and Murphy (2008). However, in PL, SP1, SP2, and SEN models, there may be fewer free parameters as sparsity penalty coerces some entries to be zero. To account for this, we adopt the strategy proposed in Pan and Shen (2007); Städler et al. (2010); Xie et al. (2008), where we discount the zero entries in the factor loadings, up to $pq - q(q-1)/2$ many zeros. The penalty coefficients ρ_g and the mixing coefficient for SEN α_g are treated as hyperparameters, and they are selected in via BIC during model selection process. The component-wise range of coefficient values need to be pre-determined. During the penalty coefficient selection process, the minimum is set at 0, and the maximum is set using the method presented in Hirose and Yamamoto (2015) for consistency, where the maximum is estimated as the largest ρ such that the factor loading is still a non-zero matrix.

The sparsity of factor loading estimate are measured using two metrics. One is the proportion of zero entries (rounded to 2 decimal places) in the loading, and the mean and standard deviation of the loading entries on absolute value scale.

3.3 Numerical Experiments

In this section, we discuss various computational aspects of PL and SEN along with some other mixture models in literature in both simulated and real data settings. Various subsets of the models listed below are fitted in each experiment and illustration. The italicized abbreviations will be used henceforth when a model is referred. For each model, its description, hyperparameter and model selection process are outlined below.

- *GMM* from the R package *mclust* (Scrucca et al., 2016): This is a parsimonious Gaussian mixture model with constraints on the modified eigen-decomposition of component-wise covariance matrices $\mathbf{\Sigma}_g = \lambda_g \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g'$. Each element of the decomposition (λ_g, $\boldsymbol{P}_g$, $\boldsymbol{D}_g$) can be constrained for equality across mixture components.

The Bayesian Information Criterion (BIC) is used to select the best-fitting component count and covariance constraint.

- *HDDC* from R package **HDclassif** (Bergé et al., 2012): This another parsimonious Gaussian mixture model with projection-based constraints that seeks the intrinsic dimension d_g of each component in the mixture. Each d_g is assumed to be less than the observed dimension p, and the observations in each component is projected onto a d_g-dimensional subspace during model-fitting process. Additional parsimony can be achieved by constraining the eigen-decomposition of component-wise covariance matrices. The scree test by Cattell (1966) is used to approximate d_g for each component. Then, the BIC is used to select the best-fitting component count and covariance constraint.

- *tMM* from the R package *teigen* (Andrews et al., 2018): This is a parsimonious mixture of t-distributions with the same type of covariance constraints as that in GMM from mclust package. The BIC is used to select the best-fitting component count and covariance constraint.

- *PGMM* from the R package *pgmm* (McNicholas et al., 2018): This is a parsimonious mixture of Gaussian factor analyzers, where the factor loadings $\mathbf{\Lambda}_g$ and random error covariance $\mathbf{\Psi}_g$ are constrained to reduce the number of free parameters. The BIC is used to select the best component count, factor count and the model constraint.

- *PL, SP1, SP2, SEN*: They are the four finite mixtures of penalized factor analyzers employing the penalty corresponding to the abbreviations. Hyperparameter selection is done in two stages: factor analyzer-related quantities first, then penalty-related quantities. An un-penalized finite mixture of factor analyzers are fitted to select the best component and factor counts. Then, if SEN is used, for each $\alpha \in \{0, 0.1, 0.2, \ldots, 1\}$, the penalty coefficients ρ_g are selected, then the penalized model is fitted. Selection of ρ_g is done over a grid its range is set according to section 3.2.6. For PL, SP1 and SP2, the α selection process is skipped.

To avoid premature stopping of the parameter estimation, Aitken's acceleration with threshold $\epsilon = 0.01$ is used as the model convergence criterion whenever the software package accommodates it. Otherwise, the packages' default convergence criteria were used. The clustering performance is measured by the Adjusted Rand Index (ARI). We consider two simulations and two real data analyses. For simulations, we conduct the following.

i) **Change in factor loading sparsity and rank as penalty increases.** We study the change in sparsity and the rank of the factor loading estimate as the penalty coefficient ρ increases. This experiment is intended to show that direct penalization on the loading can result in rank-deficient estimates, and that the SP1, SP2 and SEN are robust to rank-deficiency.

ii) **The effect of mixing coefficient α_g.** We study the effect of varying α_g value on the resulting mixture model.

For real data illustration, we discuss the following data sets.

i) **Wine data.** We perform clustering on the Wine data set, as a benchmarking test against other model-based clustering methods in a high-dimensional setting.

ii) **Movehub data.** We perform clustering on the Movehub quality-of-life data set. We pay particular attention to the interpretability of the resulting model.

3.3.1 Change in Factor Loading Sparsity and Rank

We study the effect of increasing penalty coefficient ρ on the sparsity and the rank of the factor loading estimate. The four penalized methods - PL, SP1, SP2 and SEN - are tested. Since the factor loading estimate is of primary concern, we simulated 1-component Gaussian data set with zero mean for this experiment. The considered sample sizes (n) and data dimensions (p) are $n = 100, 500$ and $p = 5, 50$ respectively. For $p = 5$, the true number

of factors is $q = 2$. For $p = 50$, the true number of factors is $q = 5$. The factor loading and the covariance for the random noise used to simulate the data set are as follows.

$$\boldsymbol{\Lambda}_{5\times 2} = \begin{bmatrix} 0.1 & 1.4 \\ -0.5 & 2 \\ 1 & -2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \boldsymbol{\Lambda}_{50\times 5} = \begin{bmatrix} 0.1 & 1.4 & -0.5 & 2 & 1 \\ -2 & 1 & 1.5 & -3 & 0 \\ 3.1 & 2.5 & 0 & 1 & -1 \\ 0 & 0.1 & -4 & 2 & 0 \\ 0.5 & 0 & 2 & 0 & -1 \\ & & \mathbf{0}_{45\times 5} & & \end{bmatrix}$$

$$\boldsymbol{\Psi}_{5\times 5} = 2\boldsymbol{I}_5, \qquad \boldsymbol{\Psi}_{50\times 50} = 2\boldsymbol{I}_{50}$$

With regards to the hyper-parameter setup, to avoid confounding effect between parameters, the true number of components and factors are used when fitting the model. The g subscript is dropped for notational brevity. For each combination of (n, p), the experiment was replicated 500 times, each with a newly-generated data set.

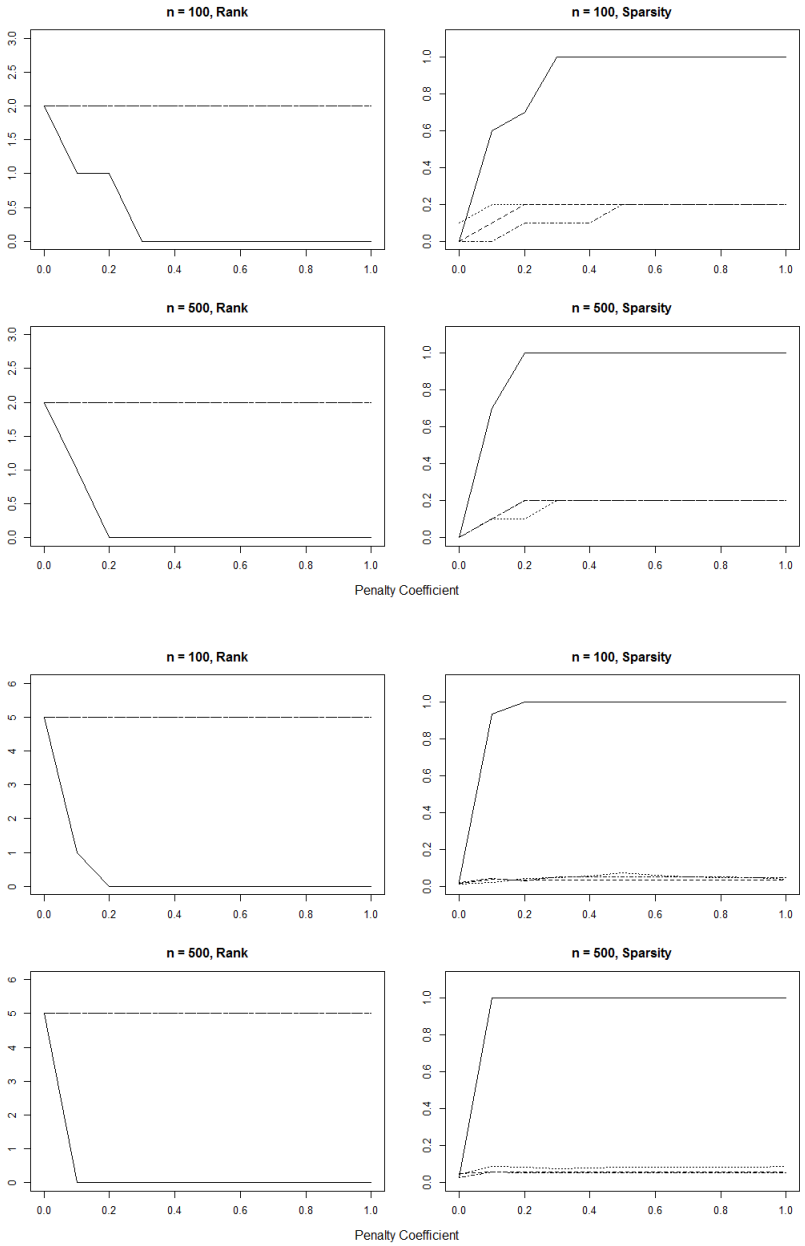

Figure 3.2: Plots of median rank and sparsity against penalty coefficient $\rho \in [0, 1]$. The top plot is for $(p, q) = (5, 2)$ dimensional case, and the bottom plot is for $(p, q) = (50, 5)$ dimensional case. In each panel, the top row is for sample size $n = 100$, and the bottom row is for sample size $n = 500$. The left column contains the plots of estimated factor loading's column rank, and the right column contains the plots of the proportion of zero entries in the estimated factor loading. The solid line is for PL, the dashed line is for SP1, the dotted line is for SP2, and the dot-dash line is for SEN.

$p = 5$	PL	SP1	SP2	SEN
$n = 100$	0.40	0.64	2.69	4.11
$n = 500$	1.48	2.95	4.05	5.04

$p = 50$	PL	SP1	SP2	SEN
$n = 100$	16.49	6.27	15.03	14.72
$n = 500$	30.36	21.34	30.14	29.95

Table 3.1: Table of median elapsed time (in seconds) all (n, p) pairs tested. The top table is for the $p = 5$ case, and the bottom table is for the $p = 50$ case. In each table, the top and bottom rows are for $n = 100$ and $n = 500$ cases respectively. The columns correspond to each model tested, marked by their abbreviations.

Figure 3.2 is the set of plots generated from experiments with $(p, q) = (5, 2)$ dimensional data. The left column contains the plots of estimated factor loading's column rank, and the right column contains the plots of the proportion of zero entries in the estimated factor loading. The top row is for sample size $n = 100$, and the bottom row is for sample size $n = 500$. Consider the left column of this figure. The PL shows a rapid reduction in factor loading column rank as ρ increases. Contrarily, all of SP1, SP2 and SEN maintain the full column rank, as expected. On the right column, we see a rapid increase in the proportion of zero entries in the factor loading estimates generated by PL model. A trade-off for the SP1, SP2 and SEN is the reduced sparsity proportion. In practice, one may use the upper bound-setting method for ρ outlined in Hirose and Yamamoto (2015) to avoid over-penalization of factor loading. However, in all simulated cases in this experiment, the estimated upper bound on ρ was 0 for every replication despite the true factor loading being quite sparse. Because an upper bound of 0 forbids any penalization, the merit of a penalized model is lost. The SP1, SP2 and SEN are robust against this issue, as they are a lot less sensitive to the increasing ρ value. Thus, they can alleviate the burden of penalty coefficient tuning. Another trade-off is the increased computation time in a low-dimensional setting. Table 3.1 shows the median elapsed time for the tested models under each of (n, p) case. Here, we see that the PL is the most computationally efficient with the median elapsed time of 0.40 seconds and 1.48 seconds for $(n = 100, p = 5)$ and $(n = 500, p = 5)$ cases respectively. However, the rank-preserving penalties gain an edge in a high-dimensional setting. The

bottom table in table 3.1 shows similar levels of median elapsed time for both $n = 100$ and $n = 500$ cases. Indeed, this is promising for the SEN since sparse parameter estimates are more desirable in higher dimensions. The SP1 was consistently cheaper computationally than the SP2 in all scenarios.

3.3.2 The Effect of Mixing Coefficient α_g

As SEN allows a flexible mixture of sparsity from SP1 and shrinkage from SP2, one might be inquisitive of the effect of mixing coefficients α_g on the resultant model. To emulate a realistic use case, we simulate a 2-component mixture of 2-dimensional Shifted Asymmetric Laplace (SAL) distributions from the R package MixSAL by Franczak et al. (2018). SAL distribution is a skewed distribution parametrized by location vector $\boldsymbol{\mu}$, skewness direction vector $\boldsymbol{\delta}$, and a positive definite scale matrix $\boldsymbol{\Sigma}$. An example of a 2-component mixture of SAL distributions is given below.

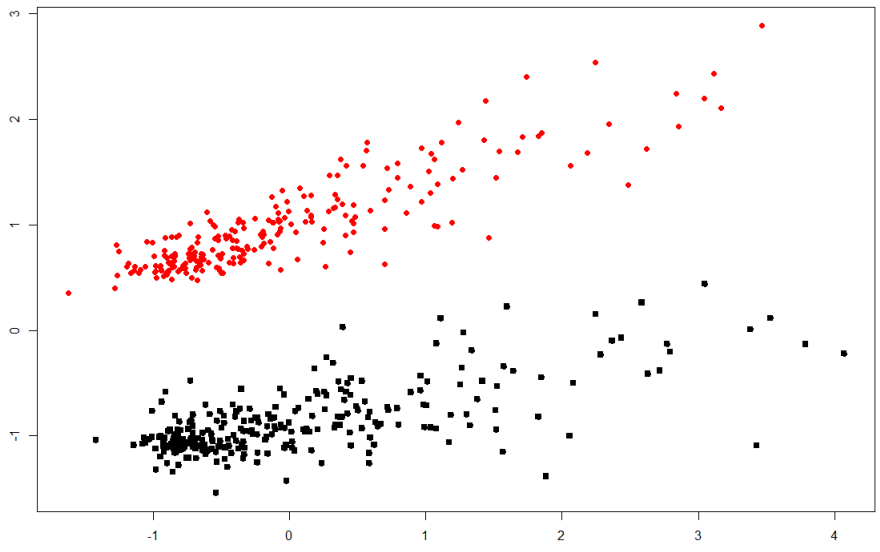

Figure 3.3: An example of 2-component mixture of 2-dimensional SAL distributions.

The considered sample sizes are $n = 100, 200, 300, 400$, and the parameter set for data generation is given below.

$$
\begin{aligned}
&\pi_1 = \pi_2 = 0.5 \\
&\boldsymbol{\mu}_1 = (0, 0)', \quad \boldsymbol{\mu}_2 = (-2, 5)' \\
&\boldsymbol{\delta}_1 = (2, 2)', \quad \boldsymbol{\delta}_2 = (1, 2)' \\
&\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}
\end{aligned}
$$

In the experiment, we fit a GMM with SEN on the simulated data, where the number of components and factors are fixed at $G = 2$ and $q = 1$ to isolate the effect of the mixing coefficient on the model, and we set $\alpha_1 = \alpha_2$. At each sample size, the experiment was replicated 500 times, each with a newly-generated data set.

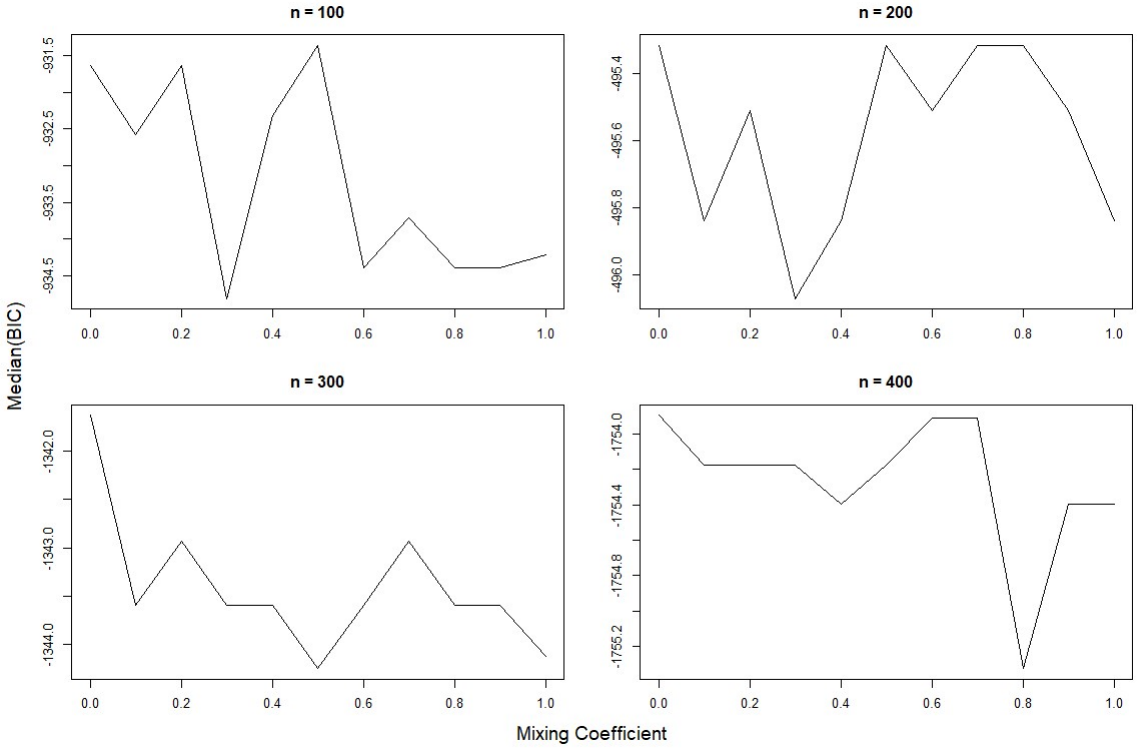

Figure 3.4: A set of median BIC vs α_g plots at $n = 100, 200, 300, 400$. In each plot, we see that pure shrinkage ($\alpha_g = 0$) results in a higher BIC than pure sparsity ($\alpha_g = 1$), and a local peak is observed at some point within $\alpha_g \in (0, 1)$.

Figure 3.4 shows that pure shrinkage results in a better-fitted model than pure sparsity, and at each sample size, a local peak in median BIC is observed at an in-between value within $\alpha_g \in (0, 1)$. We reported the median instead of mean, as EM-type algorithms are sensitive to initialization. As the occasional poor initialization can result in abnormally ill-fitted models, the median would be a more befitting summary of performance than the mean. The varying α_g did not change the ARI values, however. This indicates that the mixing coefficient primarily influences the model fit, hence it is related to model interpretability.

3.3.3 Real Data Illustration 1: Wine

Regularized estimation is emphasized in high-dimensional settings, as the number of free parameters in a finite mixture model grows rapidly with dimension. A large number of free parameters leads to an overly verbose model, reducing model interpretability. The Wine data from the R package *pgmm* (McNicholas et al., 2018) is one such data set. It consists of 27 chemical and physical properties recorded from 178 wines from the Piedmont region of Italy. Each observation belongs to one of the three types - Barolo, Grignolino, and Barbera. In this illustration, we use a subset of this data, Grignolino and Barbera, which reduces the sample size to 119. This puts us closer to the "high dimension relative to sample size" scenario, where regularized estimation may be needed. We cluster this data set using all 8 models in the list of models considered. The models were initialized with k-means clustering, except for GMM, which were initialized with the default method of hierarchical clustering as k-means was not an option. With k-means, the groups were initialized by running the algorithm a fixed number of iterations (10 in this case), instead of running until convergence. By starting the algorithm with a different seed for each replication, different initializations were obtained. With hierarchical initialization, the *mclust* package allows a subset of the data to be used in initialization stage, and the grouping to be extended to the remaining portion of the data. The hyperparameter and model selection processes follow that outlined in section 3.3. The models' performance was measured by BIC and ARI. Below is the table of median BIC and ARI values from 500 replications. Similar to earlier experiments, we report the median instead of mean, as EM-type algorithms are sensitive to initialization.

	GMM	HDDC	tMM	PGMM
BIC	-8327	-7948	-8204	-8046
ARI	0.97	0.49	0.93	0.64
Time	3.58	1.33	7.99	26.56

	PL	SP1	SP2	SEN
BIC	-8058	-8146	-8126	-8134
ARI	0.97	0.97	0.93	0.97
Time	30.88	39.42	45.61	49.89

Table 3.2: Table of median BIC, ARI and elapsed time (in seconds) for each model.

The factor analyzer-based models (PGMM, PL, SP1, SP2, SEN) obtained higher BIC values than the rest, except for HDDC. This is expected, as the factor analyzer is a sub-model of HDDC (Bouveyron et al., 2007). However, it favoured BIC over clustering performance, as shown by the lowest ARI among all tested models. Additionally, the penalized models (PL, SP1, SP2, SEN) exhibited high ARI, whereas PGMM did not despite having the highest median BIC value among the aforementioned five models. Both GMM and tMM identified clusters quite well, but they trailed behind the other models in term of BIC. Overall, the penalized factor analyzers present themselves as an attractive tool for model-based clustering. However, their drawback is the extended elapsed time. That is due to the computationally intensive penalty coefficient tuning, and in case of SEN, the added step of mixing coefficient tuning process. This drawback could be mitigated by a more efficient software implementation, which is definitely an avenue for future work.

Among the Stiefel-based trio (SP1, SP2, SEN), the 2-norm-based SP2 achieved the highest median BIC of -8126, whereas the 1-norm-based SP1 took the last place. It indicates that shrinkage produces better-fitting models than sparsity, which is reasonable, as shrinkage still allows all parameters to vary, instead of coercing complete vanishment. Consequently, SEN's goodness of fit is in between that of SP1 and SP2. A point of interest is that the mean value of mixing coefficient α is 0.25. This phenomenon suggests that, given the freedom of choice, shrinkage is favoured over sparsity for model selection.

	PL	SP1	SP2	SEN
q	2.85	2.61	2.58	2.55

Table 3.3: Table of mean factor counts for PL, SP1, SP2, SEN. The Stiefel-based trio show more parsimony than PL

As shown in table 3.3, the Stiefel-based trio are more parsimonious than PL in terms of factor counts. This is expected, as the full rank constraint on $\mathbf{\Gamma}$ does not allow linearly dependent columns. Hence, the resulting factor loadings would possess only the "necessary" number of columns. In particular, SEN achieve additional savings via variable mixing coefficients α_g.

3.3.4 Real Data Illustration 2: Movehub

This section focuses on how the proposed methods may be used in practice to generate insights from the data set. We analyze the Movehub City Rankings data from Movehub (2019) that measures the quality of life in 216 cities around the world. This version of the data set is available on Kaggle (2017). The data consists of 216 rows (each row represents a city) and 6 features excluding the city name. The description of the 6 features are as follows:

- Movehub Rating: A combination of all scores for an overall rating for a city (the higher the better).
- Purchase Power: Comparison of the average cost of living with the average local wage (the higher the better).
- Health Care: Compiled from how citizens feel about their access to healthcare and its quality (the higher the better).
- Pollution: A score of how polluted people find a city, including air, water and noise pollution (the lower the better).

- Crime Rating: The extent of crimes in a city (the lower the better).
- Quality of Life: A balance of healthcare, pollution, purchase power, crime rate to give an overall quality of life score (the higher the better).

For a fair comparison, only the three parsimonious factor analyzer-based models are deployed: PL, SEN and PGMM. SP1 and SP2 are excluded, since a data-driven tuning of mixing coefficients will result in a superior fit of the model. Model selection is done via BIC over component and factor ranges $G = 1, 2, \ldots, 6$ and $q = 1, 2, 3$ respectively. We begin by presenting the resulting clusters from the three models. SEN obtained the BIC value of -3176, PL obtained -3035 and PGMM obtained -2971. This observation is consistent with that in section 3.3.3.

		PL			PGMM	
		1	2	3	1	2
	1	88	0	0	82	6
SEN	2	0	48	0	42	6
	3	0	0	80	1	79

	q		
	1	2	3
SEN	2	1	2
PL	1	1	2
PGMM	2	2	NA

Table 3.4: Left: Cross-tabulation of component labels generated by SEN against that of PL and PGMM, respectively. Right: Table of estimated factor counts per component. The SEN and PL models generated 3 clusters, but the PGMM model generated 2 clusters.

Table 3.4 shows that the SEN and PL models show perfect cluster agreement with 3 components. On the contrary, the PGMM model generated 2 clusters, where the largest disagreement occurs in its component 1. All models produced a relatively low number of factors, ranging between 1 and 2. This is expected, given that the data set has only 6 features. Next is the study of covariance structures. We examine the component-wise correlation network graphs for each model.

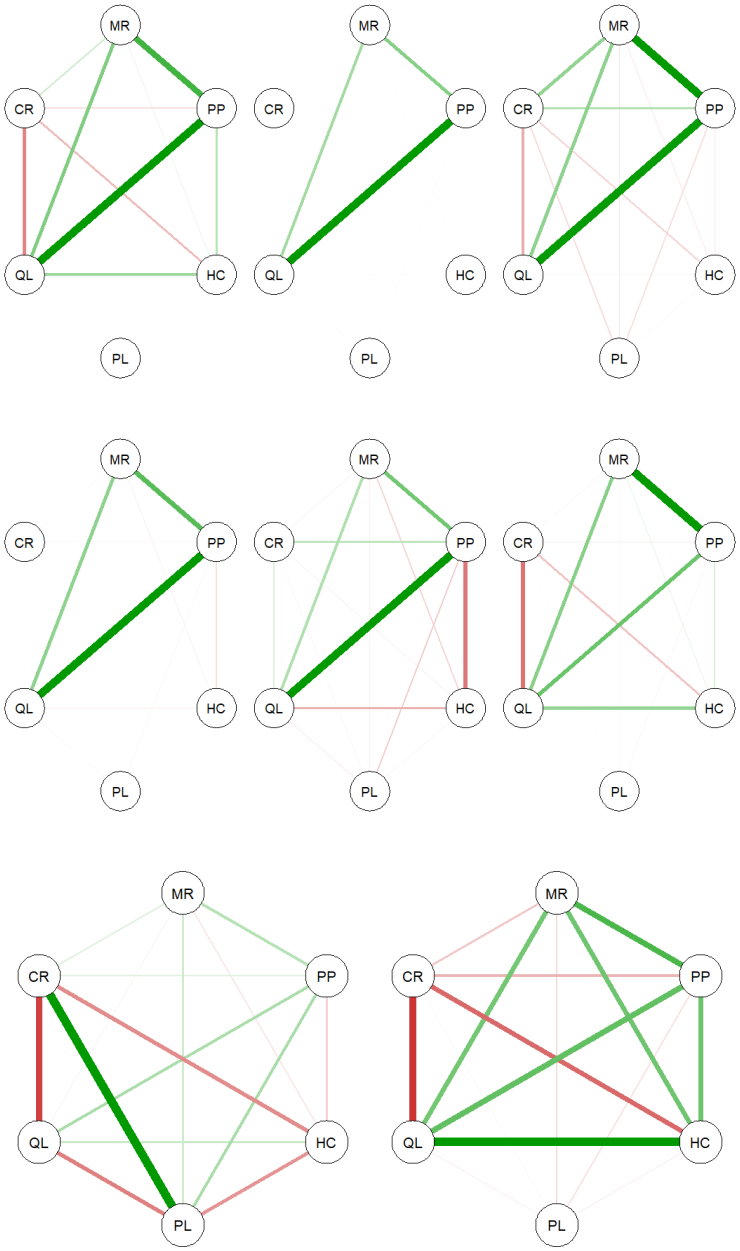

Figure 3.5: Top to bottom: Component-wise correlation network graphs for the SEN, PL and PGMM models. Green colour indicates positive correlation, and red colour indicates negative correlation. The stronger the correlation, the thicker the line. The plots are generated by the R package *qgraph* (Epskamp et al., 2012).

The two clusters generated by PGMM have a similarity that both of them have a strong negative correlation between crime rate (CR) and quality of life (QL). However, the left-side cluster has a strong positive correlation between crime rate and pollution (PL), whereas such correlation is absent in the right-side cluster. The right-side cluster is characterized by a web of positive correlations between QL, HC, PP and MR, with the QL-HC correlation being the strongest. Most of these correlations make sense, except for the mutual exclusion between the CR-PL correlation on the left and QL-HC correlation on the right. Examples of cities in the left cluster are Johannesburg, Rotterdam, Los Angeles and Dallas. Examples of cities in the right cluster are Caracas, Nairobi, Sao Paulo and Rome. In comparison, all three clusters in SEN have strong positive correlation between purchasing power and quality of life. This result is en lieu with the existing understanding on human welfare, which enhances the credibility the models generated by PL and SEN. Examples of cities in each cluster from the SEN model are as follows.

- Left: Miami, Brussels, Melbourne
- Middle: Johannesburg, Philadelphia, Dallas
- Right: Moscow, Cordoba, Colombo

Despite the complete agreement of cluster membership between SEN and PL, the visible difference in component-wise correlation networks is indeed fascinating. However, the noticeable PP-HC negative correlation and the weakened PP-QL positive correlation generated from PL could indicate that the model fitted by SEN may have captured the reality better. Depending on the user, the SEN and PL models could be over-estimating the number of components, or the PGMM model could be under-estimating the said number. However, all three methods provide ample amount of insightful leads, which is a hallmark of interpretable model-based clustering. Therefore, the methods presented in this work can be promising addition to the literature on finite mixture models.

3.3.5 Discussion

In this chapter, we extended a sparse Gaussian factor analyzer based on direct penalization of factor loading to a finite mixture model variant. More importantly, we developed a new method that can estimate sparse, yet full-rank, factor loadings in a finite mixture of Gaussian factor analyzers. We have shown its significance through its desirable theoretical bounds and promising empirical results in both simulated and real data settings. Future directions include dynamic estimation of factor counts instead of a computationally-expensive brute force search, and extension of the SEN to non-Gaussian factor analyzers.

Chapter 4

Stiefel Elastic Net Discriminant Variables: A Regularized Cluster-preserving Dimension Reduction

4.1 Introduction

In the recent past, dimension reduction for model-based clustering has received much attention. Indeed, the Curse of Dimensionality (Bellman, 2010) suggests that one should be mindful of the dimensionality of the data. Several families of approaches have been proposed that combine model-based clustering and dimension reduction. They include the eigen-decomposition of component-wise scale matrices (Bouveyron et al., 2007; Fraley and Raftery, 2002; McNicholas and Murphy, 2008), and projection-based methods such as the Invariant Coordinate Selection (ICS) (Tyler et al., 2009; Peña et al., 2010) and the Sliced Inverse Regression (SIR) (Li, 1991; Cook and Yin, 2001; Scrucca, 2010, 2014). The eigen-decomposition approach reduces the number of free parameters present in component-wise scale matrices by imposing various equality restrictions on their eigen-decomposition. De-

note the scale matrix of component g by $\boldsymbol{\Sigma}_g$. Then, we can write its eigen-decomposition as $\boldsymbol{\Sigma}_g = \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}'$. For instance, imposing $\boldsymbol{P}_g = \boldsymbol{P}$ for all g, decreases the number of free parameters required to estimate all $\boldsymbol{\Sigma}_g$. The ICS aims to find interesting structures in the data by finding a common set of coordinates present in more than one scatter matrix.

Tyler et al. (2009) proposes the projection of the data set onto the eigenvectors of $\boldsymbol{S}_1^{-1}\boldsymbol{S}_2$, where $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are two affine equivalent scatter matrices. Peña et al. (2010) investigated the adoption of multivariate kurtosis in ICS and explored its application to a 2-component Gaussian mixture model under certain forms of component-wise covariance matrices. The SIR by Li (1991) was applied in works by Cook and Yin (2001); Scrucca (2010, 2014) as a tool for dimensionality reduction and visualization for the GMM. The SIR family is the method of interest in this work, because aims to explain the fitted mixture model instead of imposing restrictions during the fitting process, and it is mathematically tractable in an arbitrary number of components. The SIR estimates the subspace that captures the clustering structure and projects the data onto it, and it could be useful when the investigator wants to identify the variables that influence the cluster structure the most. However, the combined variables generated by a projection may not always be interpretable, as they may be represented as the linear combination of a large number of original variables. In the absence of relevant domain knowledge, understanding a linear combination of many variables can be challenging. Since clustering is an exploratory task, there is a need for finding a projection of simpler structure involving fewer original variables. While several regularization techniques exist for vector-valued parameters (Hoerl and Kennard, 1970; Tibshirani, 1996; Zou and Hastie, 2005; Zou, 2006; Wang et al., 2006; Candes et al., 2007), such works for matrix-valued parameters are relatively scarce (Zhou and Li, 2014; Zhang et al., 2017; Cai et al., 2007). Moreover, because constraints are imposed usually on the matrix-valued parameters, finding a simple estimation scheme appears to be tricky.

In light of this problem, the Stiefel Elastic Net (SEN) was introduced by Kim and Browne (2021b), which enabled the constrained regularized estimation of factor loadings in the finite mixture of Gaussian factor analyzers. The two-fold benefits of the SEN are the constrained optimization of a matrix-valued parameter (factor loading) while reaping the flexibility in penalization similar to the Elastic Net by Zou and Hastie (2005), and

the straightforward iterative updates via the MM (Minorize-Maximization or Majorize-Minimization) algorithm (Hunter and Lange, 2004; Browne and McNicholas, 2014; Kiers, 2002).

In this chapter, two versions of the Stiefel Elastic Net Discriminant Variable (SENDV) for the GMM are introduced, where we estimate a regularized discriminant matrix that projects the clustered data onto a common subspace for all components using a variant of the SEN. Our version of the SEN allows a fine-tuned regularization via row-wise or column-wise penalties on the discriminant matrix, while preserving the simplicity of a MM-style update and the desirable theoretical properties.

4.2 Methodology

In this section, we present a row-wise and a column-wise version of the regularized estimation of the discriminant matrix based on the SEN. To do so, we first describe its foundational framework, the Sliced Inverse Regression, and introduce a variant of the SEN. Then, we construct our algorithms for regularized estimation and explore their theoretical properties. To simplify the estimation process in the subsequent sections, we will assume that the data set has been whitened so that the sample covariance matrix is the identity matrix, $\boldsymbol{\Sigma} = \boldsymbol{I}$. One such method is the Cholesky decomposition, where $\boldsymbol{\Sigma} = \boldsymbol{U}'\boldsymbol{U}$ such that $\boldsymbol{U}$ is upper triangular. The original variables $\boldsymbol{X}$ are then transformed to $\boldsymbol{U'}^{-1}\boldsymbol{X}$ so that $\text{Var}(\boldsymbol{U'}^{-1}\boldsymbol{X}) = \boldsymbol{I}_p$.

4.2.1 Dimension Reduction with Sliced Inverse Regression

Sliced Inverse Regression (SIR) by Li (1991) is a dimension reduction technique that aims to approximate the functional relationship between the response Z and p-dimensional covariate $\boldsymbol{x}$ using its q-dimensional linear combination ($q < p$). Formally speaking, suppose the functional relationship between Z and $\boldsymbol{x}$ is represented as

$$Z = f_{p+1}(\boldsymbol{x}, \epsilon),$$

where ϵ denotes a 1-dimensional noise that is independent from $\boldsymbol{x}$. SIR estimates a $(p \times q)$-dimensional discriminant matrix $\boldsymbol{\beta} = [\boldsymbol{\beta}_{\cdot 1} \cdots \boldsymbol{\beta}_{\cdot q}]$ such that

$$Z = f_{q+1}(\boldsymbol{\beta}'\boldsymbol{x}, \epsilon). \tag{4.1}$$

An attractive property of the SIR is the conditional independence of Z and $\boldsymbol{x}$ given $\boldsymbol{\beta}'\boldsymbol{x}$, meaning that the projected variables contain as much information on Y as the original variables. The "sliced" part refers to the partitioning of Y into G pieces, which is equivalent to assuming a piecewise constant distribution on Y. The estimation process uses the generalized eigenvalue problem. Scrucca (2010, 2014) applied the SIR to the GMM to project the data onto a subspace that captures the estimated clustering structure. We denote this technique hereafter as the GMM Dimension Reduction, or GMMDR. Under a G-component GMM, since an observation $\boldsymbol{x}$ is assigned to a component via the MAP estimate of the membership indicator $\boldsymbol{Z} = (Z_1, \ldots, Z_G)$ (as illustrated in section 2.1), the conditional independence of $\boldsymbol{Z}$ and $\boldsymbol{x}$ given $\boldsymbol{\beta}'\boldsymbol{x}$ implies $P(Z_g = 1|\boldsymbol{x}) = P(Z_g = 1|\boldsymbol{\beta}'\boldsymbol{x})$. Therefore, the GMMDR-projected variables preserve the component membership information of the original data. In practice, a G-component GMM is fitted, and its parameter estimates are used in solving the GMMDR problem. The discriminant matrix $\boldsymbol{\beta}$ arising from the GMMDR problem is the solution of

$$\max_{\boldsymbol{\beta}} \operatorname{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta}) \quad \text{subject to } \boldsymbol{\beta}'\boldsymbol{\Sigma}\boldsymbol{\beta} = \boldsymbol{I},$$

where $\boldsymbol{\Sigma}$ is the sample covariance matrix, and

$$\begin{aligned}
\hat{\pi}_g(\boldsymbol{x}) &= \frac{\pi_g \phi_g(\boldsymbol{x})}{\sum_{h=1}^{G} \pi_h \phi_h(\boldsymbol{x})}, \quad \boldsymbol{\mu} = \sum_{g=1}^{G} \pi_g \boldsymbol{\mu}_g \\
\boldsymbol{\Sigma}_g &= \sum_{i \in I_g} \hat{\pi}_g(\boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}_g)(\boldsymbol{x}_i - \boldsymbol{\mu}_g)', \quad \overline{\boldsymbol{\Sigma}} = \sum_{g=1}^{G} \pi_g \boldsymbol{\Sigma}_g \\
\boldsymbol{M}_I &= \sum_{g=1}^{G} \pi_g (\boldsymbol{\mu}_g - \boldsymbol{\mu})(\boldsymbol{\mu}_g - \boldsymbol{\mu})' \\
\boldsymbol{M}_{II} &= \sum_{g=1}^{G} \pi_g \big(\boldsymbol{\Sigma}_g - \overline{\boldsymbol{\Sigma}}\big) \boldsymbol{\Sigma}^{-1} \big(\boldsymbol{\Sigma}_g - \overline{\boldsymbol{\Sigma}}\big)' \\
\boldsymbol{M} &= \lambda \boldsymbol{M}_I \boldsymbol{\Sigma}^{-1} \boldsymbol{M}_I + (1 - \lambda) \boldsymbol{M}_{II} \quad (\lambda \in [0, 1]).
\end{aligned}$$

The $\boldsymbol{M}_I$ and $\boldsymbol{M}_{II}$ matrices contain the information on the variation between component-wise means and covariances, respectively. The $\boldsymbol{M}$ matrix combines the two using a pre-determined mixing coefficient λ. Higher λ value means more emphasis on the variation among mean vectors. Under the assumption of $\boldsymbol{\Sigma} = \boldsymbol{I}_p$, the formulation of the GMMDR simplifies accordingly, In particular, the discriminant matrix $\boldsymbol{\beta}$ now exists in the Stiefel manifold $V_{p,q} = \{\boldsymbol{\beta} \in \mathbb{R}^{p \times q} : \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q\}$ with $p \geq q$.

$$\max_{\boldsymbol{\beta}} \operatorname{tr}(\boldsymbol{\beta}' \boldsymbol{M} \boldsymbol{\beta}) \quad \text{subject to } \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q.$$

4.2.2 Stiefel Elastic Net

The Stiefel Elastic Net introduced by Kim and Browne (2021b) is a penalty function for regularizing matrix-valued parameters over the Stiefel manifold $V_{p,q}$. For $\boldsymbol{\beta} \in V_{p,q}$, the Stiefel Elastic Net is defined as

$$SEN(\boldsymbol{\beta}; \alpha, \rho) = \rho \left(\alpha \sum_{i=1}^{p} ||\boldsymbol{\beta}_{i\cdot}||_1 + (1 - \alpha) \sum_{i=1}^{p} ||\boldsymbol{\beta}_{i\cdot}||_2 \right),$$

where $\boldsymbol{\beta}_{i\cdot}$ denotes the i^{th} row of $\boldsymbol{\beta}$, $\alpha \in [0,1]$ is the weight hyperparameter and $\rho \geq 0$ is the penalty multiplier. The SEN is an attractive penalty function for applicable matrix parameters due to desirable theoretical bounds and the straightforward optimization. For instance, for $\alpha \in (0,1)$ and $\rho > 0$, a minimizer of $SEN(\cdot;\alpha,\rho)$ is a matrix of q many signed standard basis vectors, with the corresponding minimum being ρq. As for its minimization, the MM (Majorize-Minimization or Minorize-Maximization) algorithms by Hunter and Lange (2004); Kiers (2002) lead to a convenient iterative procedure. A succinct description of the MM algorithm is provided in chapter 3.2.5.

Kiers (2002) introduced a MM algorithm for the following form of trace minimization.

$$\min_{\boldsymbol{\beta}} \sum_{k=1}^{K} \text{tr}(\boldsymbol{B}_k \boldsymbol{\beta} \boldsymbol{C}_k \boldsymbol{\beta}') \quad \text{subject to} \quad \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q, \tag{4.2}$$

where $\mathbf{B}_k$ are square and $\mathbf{C}_k$ are positive semidefinite. Its majorizer at iteration t is $\text{tr}(\mathbf{F}^{(t)}\boldsymbol{\beta}) + \text{constant}$ where

$$\boldsymbol{F}^{(t)} = \sum_{k=1}^{K} \boldsymbol{C}_k \boldsymbol{\beta}^{(t)'} \boldsymbol{B}_k + \boldsymbol{C}_k' \boldsymbol{\beta}^{(t)'} \boldsymbol{B}_k' - 2\lambda_k \boldsymbol{\beta}^{(t)'},$$

and $\lambda_k =$ the product of the highest or of the lowest eigenvalues of $\boldsymbol{B}_k$ and $\boldsymbol{C}_k$, whichever is the highest. For indices k where one of the $\boldsymbol{B}_k$ or $\boldsymbol{C}_k$ is positive semidefinite and the other is negative semidefinite, $\lambda_k = 0$. Let $\boldsymbol{PDQ'}$ denote the singular value decomposition (SVD) of $-\boldsymbol{F}^{(t)}$. Then, the new estimate is given by $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{QP'}$. Also, de Leeuw and Lange (2009) developed a quadratic majorizer for $|\boldsymbol{\beta}_{ij}|$

$$|\boldsymbol{\beta}_{ij}| \leq \frac{\boldsymbol{\beta}_{ij}^2}{2|\boldsymbol{\beta}_{ij}^{(t)}| + \epsilon} + \frac{|\boldsymbol{\beta}_{ij}^{(t)}|}{2}, \tag{4.3}$$

where $\epsilon > 0$ is a small constant added to avoid singularity at 0. In this work, we set $\epsilon = 10^{-5}$. This process is iterated until the distance between the current and the previous estimates of $\boldsymbol{\beta}$ is smaller than a pre-determined threshold $c > 0$. The Frobenius norm $||\cdot||_F$

is used to compute the distance, where

$$||\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}||_F = \sqrt{\sum_{i=1}^{p}\sum_{j=1}^{q}\left(\boldsymbol{\beta}_{ij}^{(t)} - \boldsymbol{\beta}_{ij}^{(t-1)}\right)^2}.$$

For $\boldsymbol{\beta} \in V_{p,q}$ with $p \geq q$, the row-wise and column-wise variants of the SEN will be denoted by SEN_r and SEN_c respectively, where

$$\begin{aligned} &SEN_r(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_p) \\ &\quad = \alpha \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_1 + (1-\alpha) \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_2^2, \text{ and} \end{aligned} \tag{4.4}$$

$$\begin{aligned} &SEN_c(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_q) \\ &\quad = \alpha \sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_1 + (1-\alpha) \sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_2^2 \\ &\quad = \alpha \sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_1 + (1-\alpha) \sum_{j=1}^{q} \rho_j. \end{aligned} \tag{4.5}$$

The three key differences between the original SEN and the above variants are

- The $||\boldsymbol{\beta}_{i\cdot}||_2$ is replaced by its square $||\boldsymbol{\beta}_{i\cdot}||_2^2$. In Kim and Browne (2021b), both the 1-norm and 2-norm components of the SEN were majorized. However, the squared 2-norm need not be majorized. Since majorization is a form of approximation, squaring allows a more direct estimation of $\boldsymbol{\beta}$.

- The original SEN penalizes row-wise. However, a column-wise penalization is also an option. The row-wise penalty can be seen as minimizing the the number of discriminant variables wherein a data set's variable appears. Contrarily, the column-wise penalty aims to minimize the total number of data set's variables appearing in each discriminant variable. Both approaches are valid, and the preference will depend on

the application. Therefore, we consider both row-wise and column-wise penalties.

- Instead of a single penalty multiplier ρ, we assign each row (or column) its own multiplier. This allows a more fine-tuned penalization on the discriminant matrix $\boldsymbol{\beta}$.

In the remainder of this section, we present the algorithm for estimating the regularized $\boldsymbol{\beta}$ based on the row-wise and column-wise penalties, and they will be named hereafter as rSENDV (row-wise SEN Discriminant Variables) and cSENDV (column-wise SEN Discriminant Variables) respectively. The two algorithms will be collectively referred to as the SENDV.

4.2.3 Row-wise Penalization

The row-wise penalized objective for the rSENDV is

$$\begin{aligned} &\max_{\boldsymbol{\beta}} \operatorname{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta}) - SEN_r(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_p) \\ &\text{subject to} \quad \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q. \end{aligned} \tag{4.6}$$

The original objective $\operatorname{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta}) = \sum_{j=1}^{q} \boldsymbol{\beta}_{\cdot j}'\boldsymbol{M}\boldsymbol{\beta}_{\cdot j}$ is column-oriented but the penalty is row-oriented. Thus, we need an approach that estimates $\boldsymbol{\beta}$ as a whole. Since the objective in (4.6) becomes a minimization problem with multiplication by -1, we will utilize the above two majorizers to develop a MM algorithm for the rSENDV.

Parameter Estimation

The estimation of rSENDV is based on the majorization of the 1-norm component of SEN_r, followed by the majorization of the penalized objective function and the MM-based

iterative update of $\boldsymbol{\beta}$. The 1-norm component of SEN_r is majorized using (4.3).

$$\begin{aligned}\alpha \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_1 &\leq \alpha \sum_{i=1}^{p} \sum_{j=1}^{q} \frac{\boldsymbol{\beta}_{ij}^2}{2|\boldsymbol{\beta}_{ij}^{(t)}| + \epsilon} + \text{constant} \\ &= \alpha \sum_{i=1}^{p} \text{tr}\left(\boldsymbol{e}_i \boldsymbol{e}_i' \boldsymbol{\beta} \boldsymbol{A}_i \boldsymbol{\beta}'\right) + \text{constant},\end{aligned}$$

where $\boldsymbol{A}_i = (\rho_i/2)\text{diag}\left(1/(|\boldsymbol{\beta}_{i1}^{(t)}| + \epsilon), \cdots, 1/(|\boldsymbol{\beta}_{iq}^{(t)}| + \epsilon)\right)$ is a positive diagonal matrix and $\boldsymbol{e}_i$ is the standard ith basis vector. The 2-norm component can also be written in trace form as

$$(1-\alpha) \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_2^2 = (1-\alpha)\text{tr}\left(\text{diag}(\boldsymbol{\rho})\boldsymbol{\beta}\boldsymbol{\beta}'\right),$$

where $\text{diag}(\boldsymbol{\rho}) = \text{diag}(\rho_1, \cdots, \rho_p)$ is the diagonal matrix of penalty multipliers. Combining the above trace forms, the majorizer of (4.6) (multiplied by -1) is given by $\text{tr}(\boldsymbol{F}^{(t)}\boldsymbol{\beta}) + c$ where

$$\begin{aligned}\boldsymbol{F}^{(t)} = -2\boldsymbol{\beta}^{(t)'}\boldsymbol{M} &+ 2\alpha \sum_{i=1}^{p} \left(\boldsymbol{A}_i \boldsymbol{\beta}^{(t)'} \boldsymbol{e}_i \boldsymbol{e}_i' - \max(\boldsymbol{A}_i)\boldsymbol{\beta}^{(t)'}\right) \\ &+ 2(1-\alpha)\left(\boldsymbol{\beta}^{(t)'}\text{diag}(\boldsymbol{\rho}) - \max(\boldsymbol{\rho})\boldsymbol{\beta}^{(t)'}\right).\end{aligned}$$

where $\max(\boldsymbol{A}_i)$ is the maximum diagonal entry of $\boldsymbol{A}_i$. We then compute the new $\boldsymbol{\beta}$ via the MM algorithm from section 4.2.3 and update $\boldsymbol{A}_i$. The rSENDV algorithm is provided below.

Algorithm 1 rSENDV

initialize:
  $t = 0$ and $\boldsymbol{\beta}^{(0)}$
  $\boldsymbol{M} = \lambda \boldsymbol{M}_I \boldsymbol{M}_I + (1-\lambda)\boldsymbol{M}_{II}$ and $\alpha \in [0, 1]$
  $\rho_1, \ldots, \rho_p > 0$
  $c > 0$, $\epsilon > 0$, diff $= \infty$
while diff $\geq c$ **do**
  $\boldsymbol{A}_i \leftarrow (\rho_i/2)\text{diag}(1/(|\boldsymbol{\beta}_{i1}^{(t)}| + \epsilon), \cdots, 1/(|\boldsymbol{\beta}_{iq}^{(t)}| + \epsilon))$
  $\boldsymbol{F}_1^{(t)} \leftarrow 2\alpha \sum_{i=1}^{p} \left(\boldsymbol{A}_i \boldsymbol{\beta}^{(t)\prime} \boldsymbol{e}_i \boldsymbol{e}_i' - \max(\boldsymbol{A}_i)\boldsymbol{\beta}^{(t)\prime}\right)$
  $\boldsymbol{F}_2^{(t)} \leftarrow 2(1-\alpha)\left(\boldsymbol{\beta}^{(t)\prime}\text{diag}(\boldsymbol{\rho}) - \max(\boldsymbol{\rho})\boldsymbol{\beta}^{(t)\prime}\right)$
  $\boldsymbol{F}^{(t)} \leftarrow -2\boldsymbol{\beta}^{(t)\prime}\boldsymbol{M} + \boldsymbol{F}_1^{(t)} + \boldsymbol{F}_2^{(t)}$
  SVD of $-\boldsymbol{F}^{(t)}$: $\boldsymbol{PDQ'}$
  $\boldsymbol{\beta}_{(t+1)} \leftarrow \boldsymbol{QP'}$
  diff $\leftarrow ||\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}||_F$
  $t \leftarrow t + 1$
end while
return $\boldsymbol{\beta}^{(t)}$

Theoretical Property

The SEN_r enjoys desirable theoretical properties similar to that of the original SEN. Namely, its bounded below by the sum of q smallest penalty multipliers, and a matrix of q many signed standard basis vectors achieve that lower bound. Since such a matrix is the sparsest on $V_{p,q}$, SEN_r is a natural choice for regularized estimation. Proposition 5 states this property formally.

Proposition 5. *For $\boldsymbol{\beta} \in V_{p,q}$ $(p \geq q)$ with $\rho_1, \ldots, \rho_p > 0$ such that $\rho_{(1)} \leq \cdots \leq \rho_{(p)}$,*

$$SEN_r(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_p) \geq \sum_{j=1}^{q} \rho_{(j)}.$$

In particular, equality is achieved if $\boldsymbol{\beta}$ is a matrix of q many signed standard basis column vectors where each column is associated with exactly one of $\rho_{(1)}, \ldots, \rho_{(q)}$. In the case of

equal multipliers, they are ordered by their row indices from low to high.

Proof. Without loss of generality, suppose that $\rho_{(i)} = \rho_i$ for $i = 1, \ldots, p$. Firstly, some facts are listed for the rest of the proof.

a. By norm properties, $||\boldsymbol{\beta}_{i\cdot}||_1 \geq ||\boldsymbol{\beta}_{i\cdot}||_2$.

b. For $\boldsymbol{\beta} \in V_{p,q}$, $\boldsymbol{\beta}\boldsymbol{\beta}' = \boldsymbol{\beta}(\boldsymbol{\beta}'\boldsymbol{\beta})^{-1}\boldsymbol{\beta}'$ is a hat matrix with diagonals equal to $\boldsymbol{\beta}_{i\cdot}'\boldsymbol{\beta}_{i\cdot} = ||\boldsymbol{\beta}_{i\cdot}||_2^2$. From Seber (2008), we know that $||\boldsymbol{\beta}_{i\cdot}||_2^2 \leq 1$, thus $||\boldsymbol{\beta}_{i\cdot}||_2^2 \leq ||\boldsymbol{\beta}_{i\cdot}||_2$.

$$\begin{aligned}\sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_1 &\overset{(a)}{\geq} \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_2 \\ &\overset{(b)}{\geq} \sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_2^2 = \sum_{j=1}^{p} \boldsymbol{\beta}_{\cdot j}' \text{diag}(\boldsymbol{\rho}) \boldsymbol{\beta}_{\cdot j},\end{aligned} \tag{4.7}$$

where $\text{diag}(\boldsymbol{\rho}) = \text{diag}(\rho_1, \ldots, \rho_p)$ is the diagonal matrix with diagonals $\rho_1, \ldots, \rho_p$. Thus, we can find the lower bound for (4.7) by solving

$$\begin{aligned}&\min_{\boldsymbol{\beta}} \sum_{i=1}^{p} \boldsymbol{\beta}_{i\cdot}' \text{diag}(\boldsymbol{\rho}) \boldsymbol{\beta}_{i\cdot} = \text{tr}\left(\boldsymbol{\beta}' \text{diag}(\boldsymbol{\rho}) \boldsymbol{\beta}\right) \\ &\text{subject to } \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q\end{aligned}$$

As in Horn and Johnson (2012), we can solve via differentiating the Lagrangian

$$\text{tr}\left(\boldsymbol{\beta}' \text{diag}(\boldsymbol{\rho}) \boldsymbol{\beta}\right) - \text{tr}\left(\boldsymbol{\Lambda}(\boldsymbol{\beta}'\boldsymbol{\beta} - \boldsymbol{I}_q)\right),$$

where $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_q)$ is a diagonal matrix of Lagrange multipliers. Upon differentiating with respect to $\boldsymbol{\beta}$, we end up solving $\text{diag}(\boldsymbol{\rho})\boldsymbol{\beta} = \boldsymbol{\beta}\boldsymbol{\Lambda}$, which is equivalent with finding q many eigenvalues and corresponding eigenvectors of $\text{diag}(\boldsymbol{\rho})$. Suppose $\rho_1, \ldots, \rho_p > 0$. Then, the q smallest eigenvalues $\rho_{(1)}, \ldots, \rho_{(q)}$ are chosen. Because $\text{diag}(\boldsymbol{\rho})$ is diagonal, the only non-zero vector in the eigenspace corresponding to each $\rho_{(j)}$ is the signed standard

basis column vector associated with $\rho_{(j)}$. In case of equality between multipliers, selection is done lexicographically. For example, if $p = 5$ such that $\rho_1 = \rho_3 = \rho_5 > \rho_2 > \rho_4$ and $q = 4$, then ρ_5 is omitted and the solution is $\boldsymbol{\beta} = [\boldsymbol{e}_4,\ \boldsymbol{e}_2,\ \boldsymbol{e}_1,\ \boldsymbol{e}_3]$. Otherwise, some ρ_is are 0. In this case, the signed standard basis vectors are no longer the unique solution, since any non-zero orthonormal vectors in the null space of $[\boldsymbol{\rho}]$ would qualify. In all cases, a matrix of appropriately chosen signed standard basis column vectors achieve the equality $\sum_{i=1}^{p} \rho_i ||\boldsymbol{\beta}_{i\cdot}||_1 = \sum_{j=1}^{q} \rho_{(j)}$. □

4.2.4 Column-wise Penalization

The column-wise penalized objective is

$$\begin{aligned} &\max_{\boldsymbol{\beta}} \operatorname{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta}) - SEN_c(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_q) \\ &\text{subject to} \quad \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q. \end{aligned} \tag{4.8}$$

The SEN_c is easier to combine with the original objective because both are column-oriented. In particular, after minorizing the 1-norm component (multiplying the majorizer by -1), (4.8) can be minorized by the following heterogeneous quadratic form

$$\max_{\boldsymbol{\beta}} \sum_{j=1}^{q} \boldsymbol{\beta}'_{\cdot j} \left(\boldsymbol{M} - \alpha \boldsymbol{A}_j\right) \boldsymbol{\beta}_{\cdot j} \quad \text{subject to} \quad \boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q, \tag{4.9}$$

where $\boldsymbol{A}_j = (\rho_j/2)\text{diag}\left(1/|\boldsymbol{\beta}_{1j}^{(t)}|, \cdots, 1/|\boldsymbol{\beta}_{pj}^{(t)}|\right)$. In this formulation, the definiteness of $\boldsymbol{M} - \alpha\boldsymbol{A}_j$ affects the solution. In particular, ρ_j can change the definiteness because it is not bounded above initially. As an illustration, consider a single q-dimensional quadratic maximization problem with a positive diagonal matrix $\boldsymbol{D}$ with decreasing diagonals and $\rho \geq 0$

$$\max_{\boldsymbol{x}} \boldsymbol{x}'(\boldsymbol{D} - \rho\mathbf{I})\boldsymbol{x} = \sum_{j=1}^{q} (\boldsymbol{D}_{jj} - \rho)\boldsymbol{x}_j^2 \quad \text{subject to} \quad \boldsymbol{x}'\boldsymbol{x} = 1.$$

If $\rho \leq \boldsymbol{D}_{qq}$, then every $\boldsymbol{x}_j^2$ can be non-zero. However, if $\rho > \boldsymbol{D}_{kk}$ for $k \in \{1, \ldots, q\}$, then any optimal $\boldsymbol{x}$ will have all $\boldsymbol{x}_{j \leq k}^2 = 0$, because the corresponding $(\boldsymbol{D}_{jj} - \rho)$ are negative. This means that excessive values of ρ confiscate the opportunity for some of $\boldsymbol{x}_j^2$s to be estimated. This phenomenon can be compared to a fair race. In every race, there is the fastest, as well as the slowest. Even if we could identify the slowest runner a priori, they should be allowed to compete until the finish line, otherwise the result becomes biased due to the preemptive disqualification. This illustration suggests that ρ_j should be bounded so that $\boldsymbol{M} - \alpha \boldsymbol{A}_j$ remains positive semidefinite throughout the estimation procedure.

Iterative Penalty Multiplier Tuning

A key decision in a penalized estimation method is the penalty multiplier selection. In rSENDV, the multipliers $\rho_1, \ldots, \rho_p$ are pre-determined and held constant throughout. Moreover, setting the multipliers can be challenging without a computationally expensive strategy like cross-validation. However, with cSENDV, an upper bound for each ρ_j can be obtained iteration-wise. Let $\lambda_{\min}(\cdot)$ denote the minimum eigenvalue of the argument matrix. We present the following upper bound on $\rho_j^{(t)}$.

Proposition 6. *Let* $\boldsymbol{A}_j^\star = (1/2)\,diag(1/|\boldsymbol{\beta}_{1j}^{(t)}|, \cdots, 1/|\boldsymbol{\beta}_{pj}^{(t)}|)$. *If* $\rho_j^{(t)} \leq \lambda_{\min}(\boldsymbol{M}) / \max(\alpha \boldsymbol{A}_j^\star)$, *then* $\boldsymbol{M} - \alpha \boldsymbol{A}_j$ *is positive semidefinite at iteration* t. *If any of* $\boldsymbol{\beta}_{1j}^{(t)}, \ldots, \boldsymbol{\beta}_{pj}^{(t)}$ *is 0, then set* $\rho_j^{(t)} = 0$.

Proof. We first show that $\boldsymbol{M}$ and $\alpha \boldsymbol{A}^\star$ are positive semidefinite, and derive a condition on which $\boldsymbol{M} - \rho_j \alpha \boldsymbol{A}^\star$ is positive semidefinite. For $\boldsymbol{x}$ such that $\boldsymbol{x}'\boldsymbol{x} = 1$, we have

$$
\begin{aligned}
\boldsymbol{x}'\boldsymbol{M}\boldsymbol{x} &= \lambda \boldsymbol{x}' \boldsymbol{M}_I' \boldsymbol{M}_I \boldsymbol{x} \\
&\quad + (1 - \lambda) \sum_{g=1}^{G} \pi_g \boldsymbol{x}' (\boldsymbol{\Sigma}_g - \overline{\boldsymbol{\Sigma}})(\boldsymbol{\Sigma}_g - \overline{\boldsymbol{\Sigma}})' \boldsymbol{x} \\
&= \lambda ||\boldsymbol{M}_I \boldsymbol{x}||_2^2 + (1 - \lambda) \sum_{g=1}^{G} \pi_g ||(\boldsymbol{\Sigma}_g - \overline{\boldsymbol{\Sigma}})' \boldsymbol{x}||_2^2 \\
&\geq 0.
\end{aligned}
$$

Moreover, since $\boldsymbol{A}^\star$ is clearly positive semidefinite by definition. This implies that $\boldsymbol{x}'\boldsymbol{M}\boldsymbol{x} \geq \lambda_{\min}(\boldsymbol{M})$ and $\rho_j \alpha \boldsymbol{x}'\boldsymbol{A}^*\boldsymbol{x} \leq \rho_j \lambda_{\max}(\alpha \boldsymbol{A}^*)$, where $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimum and maximum eigenvalues of the argument matrix respectively. Hence, if

$$\lambda_{\min}(\boldsymbol{M}) - \rho_j \lambda_{\max}(\alpha \boldsymbol{A}^\star) \geq 0,$$

then $\boldsymbol{M} - \rho_j \alpha \boldsymbol{A}^*$ is positive semidefinite. This happens when

$$\rho_j \leq \frac{\lambda_{\min}(\boldsymbol{M})}{\lambda_{\max}(\alpha \boldsymbol{A}^\star)}.$$

Thus, at iteration t, as long as $\rho_j^{(t)}$ is below the above bound, $\boldsymbol{M} - \rho_j^{(t)} \alpha \boldsymbol{A}^\star$ is positive semidefinite as required. □

Computationally, due to the presence of ϵ, $\max(\alpha \boldsymbol{A}_j^\star)$ will be finite, so $\rho_j^{(t)}$ will not be coerced into an exact zero. Using this bound, we can select the penalty multipliers $\rho_1, \ldots, \rho_q$ that do not over-penalize. Returning to the fair race analogy, this bound allows the would-be-zero entries to vanish naturally without the multiplier overpowering them.

Parameter Estimation

We apply the optimization algorithm by Bolla et al. (1998) for a heterogeneous quadratic form constrained on $V_{p,q}$. Given the objective

$$\max_{\boldsymbol{\beta}} \sum_{j=1}^{q} \boldsymbol{\beta}_{\cdot j}' \boldsymbol{G}_j \boldsymbol{\beta}_{\cdot j}$$

and the current estimate $\boldsymbol{\beta}^{(t)}$, we compute

$$\boldsymbol{G}(\boldsymbol{\beta}^{(t)}) := \left[\boldsymbol{G}_1 \boldsymbol{\beta}_{\cdot 1}^{(t)}, \cdots, \boldsymbol{G}_q \boldsymbol{\beta}_{\cdot q}^{(t)} \right].$$

Then, given the SVD of $\boldsymbol{G}(\boldsymbol{\beta}^{(t)}) = \boldsymbol{PDQ}'$, the next estimate is given by

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{PQ}'.$$

For cSENDV, we let $\boldsymbol{G}_j = \boldsymbol{M} - \alpha \boldsymbol{A}_j$ $(j = 1, \ldots, q)$. We can now solve for $\boldsymbol{\beta}$ by updating $\boldsymbol{\beta}^{(t)}$ and $\boldsymbol{A}_j$ iteratively. We will consider the algorithm as converged if the distance between the the current and previous estimates of $\boldsymbol{\beta}$ is smaller than a pre-determined threshold $c > 0$. The full cSENDV algorithm is provided below.

Algorithm 2 cSENDV

initialize:
  $t = 0$ and $\boldsymbol{\beta}^{(0)}$
  $\boldsymbol{M} = \lambda \boldsymbol{M}_I \boldsymbol{M}_I + (1 - \lambda)\boldsymbol{M}_{II}$ and $\alpha \in [0, 1]$
  $c > 0$, $\epsilon > 0$ and diff $= \infty$
while diff $\geq c$ **do**
  $\boldsymbol{A}_j^{\star} \leftarrow (1/2)\text{diag}(1/(|\boldsymbol{\beta}_{1j}^{(t)}| + \epsilon), \cdots, 1/(|\boldsymbol{\beta}_{pj}^{(t)}| + \epsilon))$
  $\rho_j^{(t)} \leftarrow \lambda_{\min}(\boldsymbol{M}) / \max(\alpha \boldsymbol{A}_j^{\star})$
  $\mathbf{G}(\boldsymbol{\beta}^{(t)}) \leftarrow \boldsymbol{M}\boldsymbol{\beta}^{(t)} - \dfrac{\alpha}{2}\text{sgn}(\boldsymbol{\beta}^{(t)})\text{diag}(\boldsymbol{\rho}^{(t)})$
  SVD of $\boldsymbol{G}(\boldsymbol{\beta}^{(t)})$: $\boldsymbol{PDQ}'$
  $\boldsymbol{\beta}_{(t+1)} \leftarrow \boldsymbol{PQ}'$
  diff $\leftarrow ||\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}||_F$
  $t \leftarrow t + 1$
end while
return $\boldsymbol{\beta}^{(t)}$

Theoretical Property

The SEN_c enjoys desirable theoretical properties similar to SEN_r in the sense that it is bounded below by the sum of penalty multipliers and a matrix of q many signed standard basis vectors achieves this bound.

Proposition 7. *For $\boldsymbol{\beta} \in V_{p,q}$ $(p \geq q)$,*

$$SEN_c(\boldsymbol{\beta}; \alpha, \rho_1, \ldots, \rho_q) \geq \sum_{j=1}^{q} \rho_j.$$

In particular, equality is achieved if $\boldsymbol{\beta}$ is a matrix of q many signed standard basis column vectors, pairwise orthonormal.

Proposition 7 indicates SEN_c is also a desirable penalty function that can help with estimating a maximally sparse projection $\boldsymbol{\beta}$.

Proof. By norm properties,

$$\sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_1 \geq \sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_2 = \sum_{j=1}^{q} \rho_j,$$

where the equality comes from the constraint $\boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q$. If $\boldsymbol{\beta}$ consists of q many distinct signed standard basis column vectors, then $\sum_{j=1}^{q} \rho_j ||\boldsymbol{\beta}_{\cdot j}||_1 = \sum_{j=1}^{q} \rho_j$. □

4.2.5 Computational Aspects

In this section, we discuss two computational aspects related to the rSENDV and cSENDV: initialization and heuristic rule for variable selection.

Initialization

Initialization of the SENDV involves the selection of hyperparameters and the initial parameter $\boldsymbol{\beta}^{(0)}$. The hyperparameters involved in the SENDV algorithms are the dimension of projected space q, the weight parameter λ for $\boldsymbol{M}$ matrix, the penalty weight parameter α and the convergence threshold c, where first two are inherited from GMMDR. While there are no clear-cut methods for choosing q, α and c, we propose a condition number-based approach for selecting λ. We begin by noting that the initial guess on $\boldsymbol{\beta}$ is likely sensitive

to some degree to that guess. This raises the stake on the initialization of $\boldsymbol{\beta}$, which could be a concern in the absence of a guideline or prior knowledge. The condition number by Trefethen and Bau III (1997) is a tool measures the sensitivity of the output against the change in the input. As a function of λ, the condition number of our objective function is

$$\text{cond}(\lambda) = \frac{\lambda_{\max}(\boldsymbol{M})}{\lambda_{\min}(\boldsymbol{M})}. \tag{4.10}$$

Thus, by minimizing cond(λ), we can select λ that makes rSENDV and cSENDV more robust against initialization. The expression in 4.10 is derived as follows. The condition number of a differentiable multivariate function f is given in Trefethen and Bau III (1997):

$$\frac{||J(\boldsymbol{x})|| \times ||\boldsymbol{x}||}{||f(\boldsymbol{x})||},$$

where $||J(\boldsymbol{x})||$ denotes the induced norm of the Jacobian matrix of f at $\boldsymbol{x}$. Since all finite-dimensional norms are equivalent, we will use the induced 2-norm $||A||_2 = \sqrt{\lambda_{\max}(A'A)}$. In our case, the function f is the objective function for SENDV, parametrized by λ: $f(\boldsymbol{\beta}; \lambda) = \text{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta})$. Due to the orthogonality constraint on $\boldsymbol{\beta}$, its norm $||\boldsymbol{\beta}||$ will be constant, so we can scale it to 1 without affecting the optima of the condition number. Finally, we assume that $\boldsymbol{\beta}$ is a square matrix. This assumption is equivalent to $\boldsymbol{\beta}$ containing the maximum number of variables for a given number of rows, which gives it the largest extent of variability. Hence, optimizing for λ in this case gives us a conservative estimate.

Then, the scaled condition number for our problem, as a function of λ, can be written as

$$\frac{||J(\boldsymbol{\beta}; \lambda)||}{\text{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta})},$$

where its maximum is found by maximizing the numerator and minimizing the denominator.

The numerator simplifies to the largest eigenvalue of $\boldsymbol{M}$, for any feasible $\boldsymbol{\beta}$. Let $\boldsymbol{PDP'}$

be the eigen-decomposition of $\boldsymbol{M}$. Using the definition of the induced 2-norm, we have

$$\begin{aligned}
||J(\boldsymbol{\beta};\lambda)||_2 &= \sqrt{\lambda_{\max}(\boldsymbol{\beta}'\boldsymbol{M}^2\boldsymbol{\beta})} \\
&= \sqrt{\lambda_{\max}(\boldsymbol{\beta}'\boldsymbol{P}\boldsymbol{D}^2\boldsymbol{P}'\boldsymbol{\beta})} \\
&= \sqrt{\lambda_{\max}(\boldsymbol{D}^2)} \\
&= \lambda_{\max}(\boldsymbol{D}) \\
&= \lambda_{\max}(\boldsymbol{M}).
\end{aligned}$$

Hence, the maximum of $||J(\boldsymbol{\beta};\lambda)||_2$ is $\lambda_{\max}(\boldsymbol{M})$.

The minimum of the numerator is $p \times \lambda_{\min}(\boldsymbol{M})$, where p can be scaled to 1 with respect to λ. Using again the eigen-decomposition of $\boldsymbol{M}$, we have

$$\operatorname{tr}(\boldsymbol{\beta}'\boldsymbol{M}\boldsymbol{\beta}) = \operatorname{tr}(\boldsymbol{D}) = \sum_{j=1}^{p} \boldsymbol{D}_{jj} \geq p \times \boldsymbol{D}_{pp} = p \times \lambda_{\min}(\boldsymbol{M}).$$

Therefore, the scaled maximum condition number for SENDV is given by

$$\operatorname{cond}(\lambda) = \frac{\lambda_{\max}(\boldsymbol{M})}{\lambda_{\min}(\boldsymbol{M})}.$$

The dimension of projected space q is estimated via the scree test by Cattell (1966) on the eigenvalues of $\boldsymbol{M}$. Let $\lambda_1 \geq \cdots \geq \lambda_p$ denote the eigenvalues of $\boldsymbol{M}$. The scree test examines the absolute difference between two consecutive eigenvalues $|\lambda_i - \lambda_{i+1}|$ and sets q to be the smallest index $i \in \{1, \ldots, p-1\}$ such that $|\lambda_i - \lambda_{i+1}| < r$. If no such i exists, then $q = p$. In this work, the scree test threshold r is set at 0.01. Also, the convergence threshold c for rSENDV and cSENDV is set at 0.001. Selecting the penalty weight α is more nuanced than the others, because it depends on the desired style of penalization. Thus, there is no single rule on choosing the optimal α. Nevertheless, to demonstrate the effect of α on $\boldsymbol{\beta}$, we will deploy rSENDV and cSENDV with various α values in the numerical experiments. In terms of selecting $\boldsymbol{\beta}^{(0)}$, we set it equal to the solution obtained from the GMMDR.

Heuristics for Variable Selection

One of the main goals of the SENDV algorithms is the identification of important variables in each projected dimension. For the purpose of preliminary assessment, we propose the following heuristic procedure for the GMMDR and SENDV. For each column of the estimated $\boldsymbol{\beta}$, we identify the entry with the largest magnitude. That entry is set to ± 1, where the sign follows that of the original entry, and the remaining entries are set to 0. After the modification, there may be identical columns. In that case, we simply remove the duplicates. An even quicker, but more crude, method is to round off $\boldsymbol{\beta}$ to the nearest digit. In this case, an all-zero column may appear, which we remove from the matrix. The above methods are suggested because no single entry in $\boldsymbol{\beta}$ can be greater than 1 due to the constraint $\boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{I}_q$. If variable selection from the unscaled data set is desired, then $\boldsymbol{\beta}$ is transformed first to $\boldsymbol{U}^{-1}\boldsymbol{\beta}$. Then, each column of $\boldsymbol{U}^{-1}\boldsymbol{\beta}$ is scaled to be of unit length, and the entry with the largest magnitude within is identified, similar to the procedure on $\boldsymbol{\beta}$.

4.3 Numerical Experiments

In this section, we present three data analyses to study the performance of the SENDV and compare it against the GMMDR and some of existing projection methods. Specifically, we present one simulated data analysis, and two real data illustrations using the Auto (James et al., 2017) and Wine (Hurley, 2019) data sets.

4.3.1 Performance Assessment

Along with visualization, the following metrics are used to assess the performance of the tested methods in all experiments.

- Ratios of the within-cluster (WSS) and between-cluster (BSS) sums of squares relative to the total sums of squares for the projected data set, denoted by rWSS and

rBSS respectively. Given a $(n \times p)$-dimensional data set $\boldsymbol{X}$, the sums of squares are computed on the projected data set $\boldsymbol{X}\bar{\boldsymbol{\beta}}$, where $\bar{\boldsymbol{\beta}}$ is the discriminant matrix generated from a method scaled to have unit length columns. For the GMM, the sums of squares are taken on the unprojected data, as there is no projection involved. The higher rWSS is relative to rBSS, the tighter-knitted and better-separated the clusters would be. Therefore, rWSS and rBSS could be viewed as a measure of cluster separation in the projected space.

- The proportion of zeros in $\bar{\boldsymbol{\beta}}$, denoted by Prop. It is the number of entries with value zero (after rounding to 3 decimal places) divided by the total number of entries in $\bar{\boldsymbol{\beta}}$. This proportion serves as a measurement on the number of original variables involved in the projected space. The higher this proportion, the fewer original variables are used for projection, indicating a more efficient representation in the projected space.
- The Bayesian Information Criterion (BIC) by Schwarz (1978).

4.3.2 Simulated Data Analysis

In this section, we compare and contrast the performance of the GMMDR and SENDV in the presence of informative and noise variables. The data set contains two informative variables, and an increasing number of noise variables is attached. The goal is to examine how much of the noise is filtered out by the GMMDR and SENDV. The informative variables are generated by a 2-dimensional 2-component GMM with equal weights $\pi_1 = \pi_2$. The component-wise mean vectors $\boldsymbol{\mu}_g$ and covariance matrices $\boldsymbol{\Sigma}_g$ are given by

$$\boldsymbol{\mu}_1 = (1,1)',\ \boldsymbol{\mu}_2 = (0,0)',\ \boldsymbol{\Sigma}_1 = \begin{bmatrix} 4 & 0 \\ 0 & 0.1 \end{bmatrix},\ \boldsymbol{\Sigma}_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 4 \end{bmatrix}.$$

A data set sampled from this distribution is plotted in figure 4.1. It is off-centre cross-shaped. To this data set, we attach d many noise variables, where each noise variable is independently Gaussian with mean 1 and variance 4. $d = 0, 2, 4, 6, 8, 10$ are considered. The number of observations in the data set is $n = 10 \times (d + 2)$. The list of tested methods

is given below. For all methods, the target dimension for projection is $q = 2$, and the true labels are used for all observations to focus on the comparison of the projection only. The BIC is computed from the GMM fitted to the projected data, based on the true label.

- GMMDR with $\lambda = 0.5$ as the baseline. This is the default setting from the function MclustDR in the R package mclust (Scrucca et al., 2016).
- rSENDV with $\alpha = 0, 0.5, 1$. λ is estimated using the objective function in (4.10). The row-wise penalty multipliers are fixed at $\rho = 0.01$.
- cSENDV with $\alpha = 0.1, 0.5, 1$, as $\alpha = 0$ is equivalent to no penalization. λ is estimated using the objective function in (4.10). The column-wise penalty multipliers are fixed at $\rho = 0.01$.

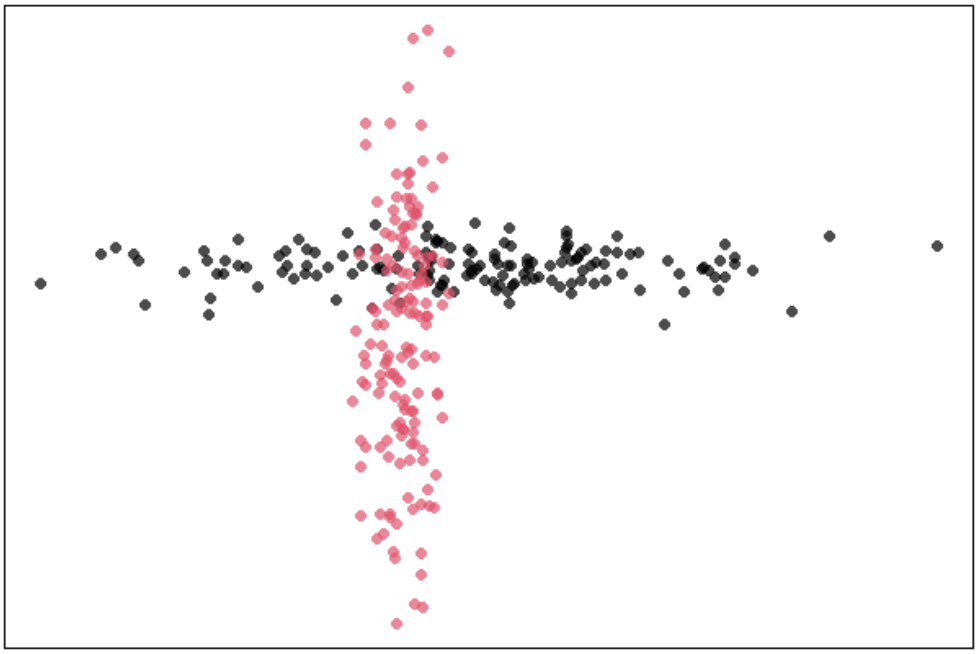

Figure 4.1: A scatterplot of the data set generated from the informative 2-component GMM.

$d = 0$	rWSS	rBSS	BIC	Prop
GMMDR	0.879	0.121	-196.721	0.002
	(0.05)	(0.05)	(11.33)	(0.02)
rSENDV [0]	0.874	0.126	-258.316	0.002
	(0.06)	(0.06)	(13.89)	(0.02)
rSENDV [0.5]	0.874	0.126	-258.328	0.132
	(0.06)	(0.06)	(13.89)	(0.13)
rSENDV [1]	0.875	0.125	-258.359	0.171
	(0.06)	(0.06)	(13.84)	(0.12)
cSENDV [0.1]	0.874	0.126	-258.321	0.001
	(0.06)	(0.06)	(13.89)	(0.01)
cSENDV [0.5]	0.874	0.126	-258.268	0.001
	(0.06)	(0.06)	(13.92)	(0.01)
cSENDV [1]	0.874	0.126	-258.161	0.001
	(0.06)	(0.06)	(13.96)	(0.01)

Table 4.1: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data with no noise variables. The numbers within the square brackets are the α values used for the SENDV method.

$d = 2$	rWSS	rBSS	BIC	Prop
GMMDR	0.896	0.104	-371.599	0.007
	(0.04)	(0.04)	(14.65)	(0.03)
rSENDV [0]	0.889	0.111	-498.375	0.006
	(0.04)	(0.04)	(19.60)	(0.03)
rSENDV [0.5]	0.934	0.066	-593.072	0.049
	(0.05)	(0.05)	(92.13)	(0.07)
rSENDV [1]	0.920	0.080	-556.041	0.068
	(0.05)	(0.05)	(85.17)	(0.07)
cSENDV [0.1]	0.891	0.109	-497.924	0.015
	(0.04)	(0.04)	(19.45)	(0.05)
cSENDV [0.5]	0.893	0.107	-499.917	0.016
	(0.04)	(0.04)	(24.48)	(0.04)
cSENDV [1]	0.894	0.106	-505.242	0.014
	(0.04)	(0.04)	(31.47)	(0.04)

Table 4.2: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data analysis. The number of noise variables d is 2. The numbers within the square brackets are the α values used for the SENDV method.

$d = 4$	rWSS	rBSS	BIC	Prop
GMMDR	0.901	0.099	-539.106	0.012
	(0.03)	(0.03)	(18.58)	(0.03)
rSENDV [0]	0.895	0.105	-733.164	0.010
	(0.03)	(0.03)	(24.36)	(0.03)
rSENDV [0.5]	0.927	0.073	-828.003	0.037
	(0.05)	(0.05)	(135.36)	(0.05)
rSENDV [1]	0.906	0.094	-764.762	0.047
	(0.04)	(0.04)	(92.79)	(0.05)
cSENDV [0.1]	0.896	0.104	-732.608	0.019
	(0.03)	(0.03)	(24.39)	(0.04)
cSENDV [0.5]	0.898	0.102	-734.027	0.017
	(0.03)	(0.03)	(26.97)	(0.04)
cSENDV [1]	0.899	0.101	-741.431	0.015
	(0.03)	(0.03)	(35.33)	(0.03)

Table 4.3: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data analysis. The number of noise variables d is 4. The numbers within the square brackets are the α values used for the SENDV method.

$d = 6$	rWSS	rBSS	BIC	Prop
GMMDR	0.903	0.097	-709.326	0.018
	(0.03)	(0.03)	(20.49)	(0.03)
rSENDV [0]	0.897	0.103	-968.550	0.011
	(0.03)	(0.03)	(28.81)	(0.03)
rSENDV [0.5]	0.915	0.085	-1042.994	0.031
	(0.04)	(0.04)	(151.89)	(0.04)
rSENDV [1]	0.900	0.100	-981.349	0.034
	(0.03)	(0.03)	(71.92)	(0.04)
cSENDV [0.1]	0.898	0.102	-967.841	0.021
	(0.03)	(0.03)	(28.73)	(0.04)
cSENDV [0.5]	0.900	0.100	-969.331	0.018
	(0.03)	(0.03)	(29.86)	(0.03)
cSENDV [1]	0.901	0.099	-978.887	0.014
	(0.03)	(0.03)	(39.34)	(0.03)

Table 4.4: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data analysis. The number of noise variables d is 6. The numbers within the square brackets are the α values used for the SENDV method.

$d = 8$	rWSS	rBSS	BIC	Prop
GMMDR	0.905	0.095	-877.662	0.019
	(0.02)	(0.02)	(24.17)	(0.03)
rSENDV ($\alpha = 0$)	0.898	0.102	-1200.257	0.015
	(0.03)	(0.03)	(31.79)	(0.03)
rSENDV ($\alpha = 0.5$)	0.908	0.092	-1246	0.026
	(0.03)	(0.03)	(140.84)	(0.03)
rSENDV ($\alpha = 1$)	0.901	0.099	-1208.405	0.028
	(0.03)	(0.03)	(66.96)	(0.04)
cSENDV ($\alpha = 0.1$)	0.900	0.100	-1199.225	0.025
	(0.03)	(0.03)	(31.72)	(0.04)
cSENDV ($\alpha = 0.5$)	0.902	0.098	-1200.866	0.023
	(0.02)	(0.02)	(32.54)	(0.03)
cSENDV ($\alpha = 1$)	0.903	0.097	-1212.343	0.017
	(0.02)	(0.02)	(41.14)	(0.03)

Table 4.5: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data analysis. The number of noise variables d is 8. The numbers within the square brackets are the α values used for the SENDV method.

$d = 10$	rWSS	rBSS	BIC	Prop
GMMDR	0.906	0.094	-1046.040	0.022
	(0.02)	(0.02)	(25.84)	(0.03)
rSENDV [0]	0.900	0.100	-1433.164	0.016
	(0.02)	(0.02)	(35.39)	(0.03)
rSENDV [0.5]	0.906	0.094	-1472.024	0.030
	(0.03)	(0.03)	(148.88)	(0.03)
rSENDV [1]	0.901	0.099	-1435.138	0.029
	(0.02)	(0.02)	(49.28)	(0.03)
cSENDV [0.1]	0.902	0.098	-1431.836	0.027
	(0.02)	(0.02)	(35.31)	(0.03)
cSENDV [0.5]	0.903	0.097	-1433.437	0.027
	(0.02)	(0.02)	(35.72)	(0.03)
cSENDV [1]	0.904	0.096	-1446.505	0.017
	(0.02)	(0.02)	(45.34)	(0.03)

Table 4.6: Table of average rWSS, rBSS, BIC and Prop (and standard deviation in brackets) over 500 replications of the simulated data with $d = 10$ noise variables. The numbers within the square brackets are the α values used for the SENDV method.

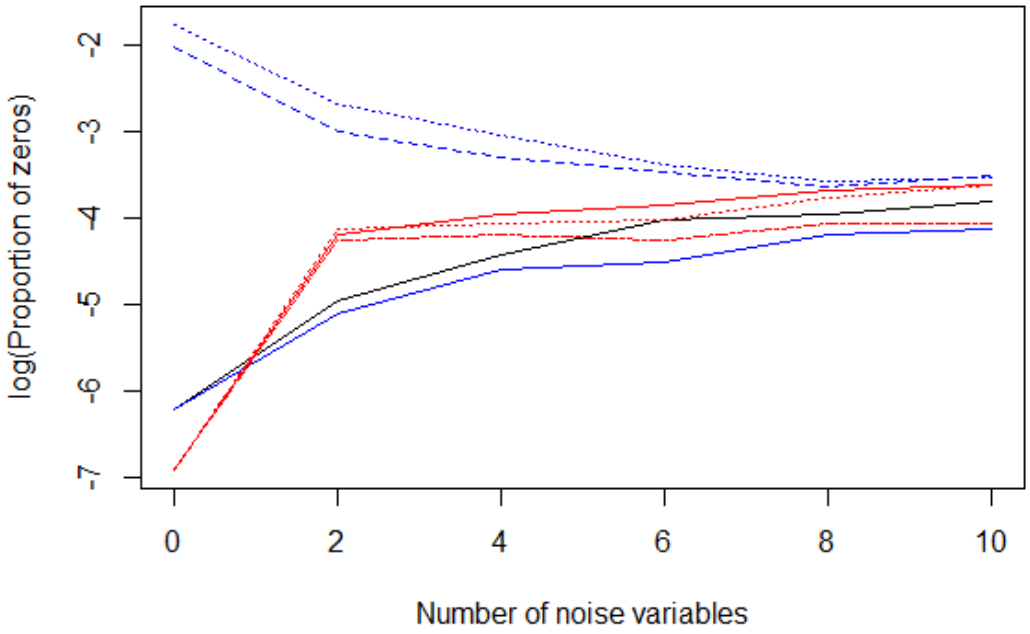

Figure 4.2: Logged average proportion of zeros in the discriminant matrices against the number of noise variables. Black, blue and red lines correspond to GMMDR, rSENDV and cSENDV respectively. For rSENDV, the solid, long dot, and short dot lines correspond to $\alpha = 0, 0.5, 1$ respectively. For cSENDV, the aforementioned three lines correspond to $\alpha = 0.1, 0.5, 1$. The gap in the proportion of zeros between methods shows a decreasing trend as the number of noise variables increases, while the rSENDV with $\alpha = 0.5, 1$ maintain a higher proportion than other methods.

Tables 4.1 to 4.6 show the average rWSS, rBSS, BIC and Prop for the tested methods and their standard deviations in brackets. In both $d = 0$ and $d = 10$, the rWSS and rBSS values are similar across the board. Moreover, for all d, the SENDV's BIC was lower on average than that of the GMMDR, but this is expected as a trade-off for penalization. When $d = 0$, the rSENDV estimates sparser discriminant matrices than the GMMDR based on Prop. As figure 4.2 shows, the rSENDV maintains in general higher Prop values than the other methods, although the gap becomes narrower at higher d values. Interestingly, the cSENDV discriminant matrix improves in sparsity as d increases, as the red lines show.

This suggests that the cSENDV may be more effective in high-noise scenarios, whereas the rSENDV may be better in low-noise cases. With respect to α, the rSENDV appears to be more sensitive to it than the cSENDV. When $\alpha = 0$, rSENDV behaves similarly to the cSENDV in terms of sparsity. The black line representing the GMMDR is in between that of the rSENDV and cSENDV in general. Overall, we see that the rSENDV is more effective in estimating a sparse discriminant matrix than the GMMDR and cSENDV.

4.3.3 Real Data Illustration 1: Auto

In this section, we illustrate the rSENDV and cSENDV using the variables from the Auto data set, available in the R package *ISLR* (James et al., 2017), and compare them to the set of existing methods. The data set consists of 6 numerical variables on 392 vehicles: mpg (miles per gallon), displacement (engine displacement), horsepower (engine horsepower), weight (vehicle weight), acceleration (time to accelerate from 0 to 60 miles per hour), and year (model year modulo 100). The 'cylinders' variable is used as the class label. It consists of 3, 4, 5, 6 and 8 cylinders. We consider the following methods. For all methods, the GMM is fitted using the R package *mclust* (Scrucca et al., 2016), with the range on the possible number of components being $G = 1, 2, \ldots, 10$

- GMM: The baseline clustering model.
- GMMDR: Fitted using *mclust* Scrucca et al. (2016).
- rSENDV: $\alpha = 0, 0.5, 1$ are used, and all penalty multipliers are set at 0.01.
- cSENDV: $\alpha = 0.1, 0.5, 1$ are used, as $\alpha = 0$ is equivalent to no penalization. The penalty multipliers are set as 0 initially, and are updated iteratively afterward.
- PCA-GMM: The dimension of the data set is reduced via Principal Component Analysis (PCA), then GMM is fitted.

- SPCA-GMM: The dimension of the data set is reduced via Sparse PCA (SPCA) (Erichson et al., 2020), then a GMM is fitted. The SPCA is a penalized PCA algorithm, with sparsity and shrinkage control parameters a and b respectively. In this work, we set $a = b = 0.01$, and the remaining parameters are set as the default value. The R package *sparsepca* (Erichson et al., 2018) is used for SPCA.

- LDA-GMM: The dimension of the data set is reduced via the Linear Discriminant Analysis (LDA) (Rao, 1948), then GMM is fitted. The R package *MASS* (Venables and Ripley, 2002) is used for LDA.

For the GMM, GMMDR, PCA-GMM, SPCA-GMM and LDA-GMM, the data set is centred and scaled to have column-wise standard deviation equal to 1. For the SENDV, the above data is scaled via the R package *whitening* (Strimmer et al., 2020), with method set to "PCA-cor". The projected dimension is fixed at $q = 2$ for all applicable methods for visualization.

	rWSS	rBSS	BIC	ARI	G
GMM	0.297	0.703	-3484.983	0.542	5
GMMDR	0.366	0.634	-829.766	0.572	2
rSENDV [0]	0.201	0.799	-606.793	**0.664**	4
rSENDV [0.5]	0.203	0.797	-605.776	0.663	4
rSENDV [1]	0.202	0.798	-605.252	0.657	4
cSENDV [0.1]	0.205	0.795	-598.855	**0.664**	4
cSENDV [0.5]	0.237	0.763	-538.617	**0.664**	4
cSENDV [1]	0.589	0.411	**-401.756**	0.628	2
PCA-GMM	0.200	0.800	-2580.136	0.268	5
SPCA-GMM	**0.199**	**0.801**	-2574.227	0.271	5
LDA-GMM	0.448	0.552	-1585.474	0.437	5

Table 4.7: Table of rWSS, rBSS, BIC and ARI, and the estimated number of components G for all tested methods, rounded to 3 decimal places. The numbers within the square brackets are the α values used for the SENDV method. For the first 4 columns, the best values are bolded.

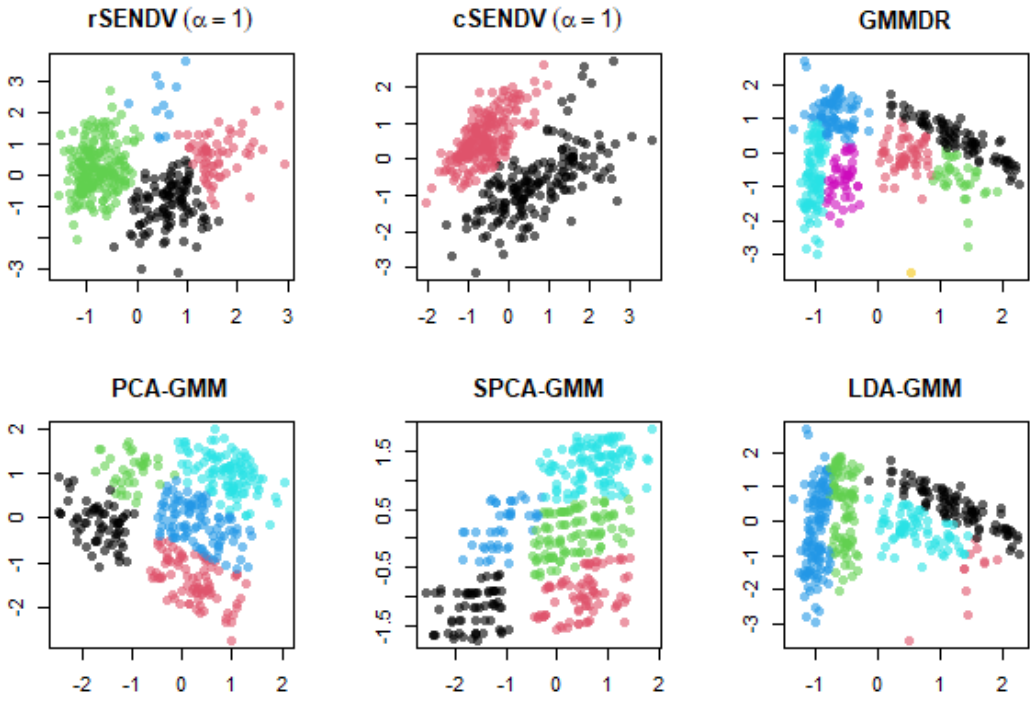

Figure 4.3: Examples of projected Auto data using the projection methods, coloured by the estimated cluster labels. From left to right, the top row corresponds to rSENDV ($\alpha = 1$), cSENDV ($\alpha = 1$), GMMDR. The bottom row corresponds to PCA-GMM, SPCA-GMM and LDA-GMM.

Table 4.7 shows favourable performances from the SENDV. Except for cSENDV ($\alpha = 1$), the SENDV methods create tighter-knitted clusters than the baseline GMM and GMMDR, as the decreased rWSS shows. Simiarly, PCA and SPCA treatment also tightened the clusters after projection. In terms of BIC, all SENDV methods outperformed other projection methods significantly. An interesting observation is that the PCA and SPCA did not improve the BIC as much as the other methods did. This is reasonable as they project the data before fitting a GMM, implying that they do not incorporate any clustering information. The ARI from the SENDV is also superior compared to its peers.

Which contributory variables for clustering did each method identify? Figure 4.3 shows the scatterplots of the projected data corresponding to some of the methods tested, colour-

coded by the estimated clusters. Figure 4.4 shows the back-to-back bar graphs plotting the absolute value of the entries in the discriminant matrices, where the left-side plot is for the x-axis, and the right-side plot is for the y-axis. For all plots, the blue bars correspond to the rSENDV ($\alpha = 1$). For instance, consider the top plot. The red bars represent the GMMDR. The left-side graph plots the absolute value of the discriminant matrices' first column for the two methods. While both methods assigned the largest coefficient to the cars' weight, the difference between it and the runner-up coefficient is larger for the rSENDV. This feature can be useful in variable selection, as the most contributory one would be highlighted. Moreover, the projected clusters created by the rSENDV are better-separated than those from the GMMDR. The ease of variable selection and the well-divided projected clusters compliment each other, since a discriminant matrix would not be very informative, even if it is easy to read, if the associated projection cannot separate the clusters. The cSENDV produced two clusters with an arguably clearer separation than that from the rSENDV. However, the borderline between the two clusters is oblique, meaning that both dimensions need to be jointly interpreted. In that regard, the rSENDV may be more convenient, as its row-wise penalization aims to avoid the prioritization of the same variable across multiple projected dimensions. This feature of the rSENDV promotes the borderlines to be orthogonal to each projected dimension. The PCA treatment heavily favoured the year variable in one dimension, but created a linear combination of all remaining variables in the other dimension, which is difficult to interpret. The structure of the SPCA's discriminant matrix is same as that of PCA, but just more sparse. In both cases, the vertical borderline creates two lumps of observations, but there seem to be too many variables involved in the x-axis, where each variable has similar magnitude of importance, as shown in figure 4.4. The LDA's separation also seems difficult to interpret, so it appears to be a suboptimal projection method for this data set. Overall, this illustration shows that the SENDV can yield a greater margin of improvement in clustering performance than the GMMDR and some of existing methods, and we also see the different potential use cases for the rSENDV and the cSENDV.

4.3.4 Real Data Illustration 2: Indian Chronic Kidney Disease

In this section, we cluster the Indian chronic kidney disease data set available in the R package teigen (Andrews et al., 2018). There are 203 observations with 2 classes: diseased and not-diseased, and there are 12 variables consisting of various biomarker measurements. The number of clusters considered are $G = 1, 2, \ldots, 10$. The target dimension q is estimated using the scree test with cutoff $= 0.1$. For the GMM, GMMDR, PCA-GMM, SPCA-GMM and LDA-GMM, the data set is centred and scaled to have column-wise standard deviation equal to 1. For the SENDV, the above data is scaled via the R package whitening (Strimmer et al., 2020), with method set to "PCA-cor". Below are the tested methods and their hyperparameter setup. For the GMMDR and SENDV, $\lambda = 1$.

- GMM: The baseline clustering model.
- GMMDR: Fitted using the R package *mclust* (Scrucca et al., 2016).
- rSENDV: $\alpha = 0, 0.5, 1$ are used, and the penalty multiplier for all rows is set at 0.1.
- cSENDV: $\alpha = 0.1, 0.5, 1$ are used. The penalty multipliers are set as 0 initially, and are updated iteratively afterward.
- PCA-GMM: The dimension of the data set is reduced via Principal Component Analysis (PCA), then GMM is fitted.
- SPCA-GMM: The dimension of the data set is reduced via Sparse Principal Component Analysis (SPCA) by Erichson et al. (2020), then a GMM is fitted. The SPCA is a penalized PCA algorithm, with sparsity and shrinkage control parameters a and b respectively. In this work, we set $a = b = 0.01$, and the remaining parameters are set as the default value. The R package *sparsepca* (Erichson et al., 2018) is used for SPCA.
- LDA-GMM: The dimension of the data set is reduced via the Linear Discriminant Analysis (LDA) by Rao (1948), then GMM is fitted. The R package *MASS* (Venables and Ripley, 2002) is used for LDA.

	rWSS	rBSS	BIC	ARI	G
GMM	0.586	0.414	-4138.466	0.782	3
GMMDR	0.243	0.757	-1038.918	0.809	3
rSENDV [0]	0.518	0.482	-867.263	0.941	2
rSENDV [0.5]	0.552	0.448	-850.751	0.941	2
rSENDV [1]	0.580	0.420	-836.994	0.941	2
cSENDV [0.1]	0.598	0.402	**-823.806**	0.961	2
cSENDV [0.5]	0.538	0.462	-857.835	0.941	2
cSENDV [1]	0.460	0.540	-891.881	**0.980**	2
PCA-GMM	0.589	0.411	-4038.303	0.510	4
SPCA-GMM	0.573	0.427	-4036.494	0.507	4
LDA-GMM	**0.164**	**0.836**	-1043.864	0.496	4

Table 4.8: Table of rWSS, rBSS, BIC, ARI, and the estimated number of components G for all tested methods on the kidney disease data, rounded to 3 decimal places. The numbers within the square brackets are the α values used for the SENDV method. For the first four columns, the best results are bolded.

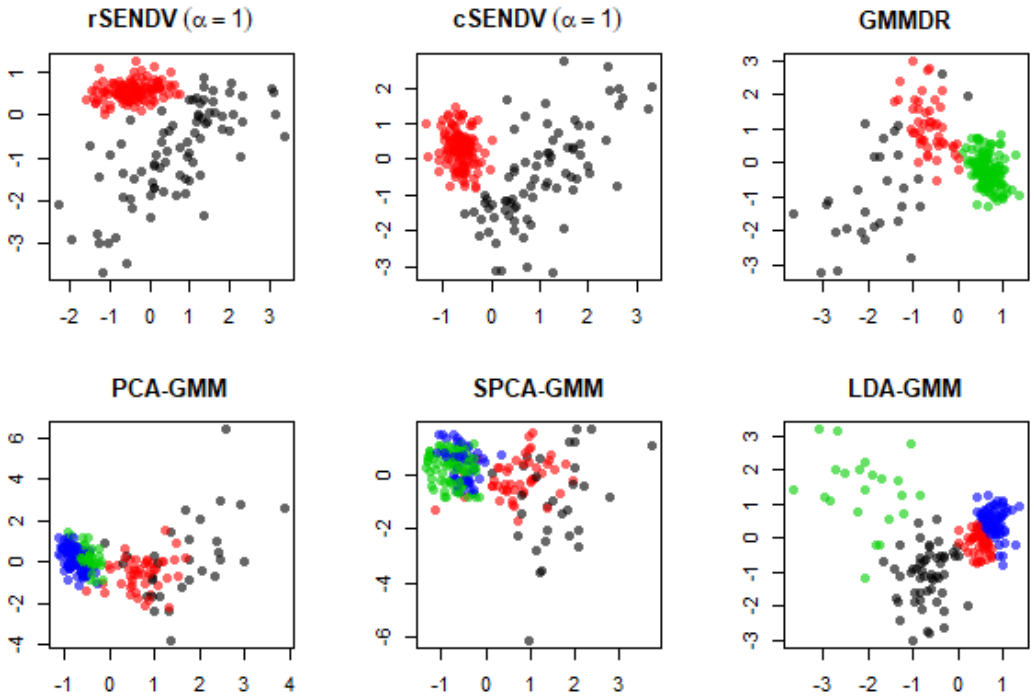

Figure 4.5: Scatterplot of the first two dimensions of the projected data from rSENDV ($\alpha = 1$), cSENDV ($\alpha = 1$), GMMDR, PCA-GMM, SPCA-GMM and LCA-GMM. The plots indicate that all six methods could separate the estimated clusters using two dimensions. However the data set from PCA-GMM and SPCA-GMM show more overlaps between clusters than the remaining methods.

Table 4.8 shows a marked improvement in model fit and clustering performance by the SENDV, compared to the baseline GMM and the GMMDR. Moreover, the projection methods that incorporated clustering information (GMMDR, SENDV and LDA-GMM) reduced the dimension to $q = 2$, whereas the PCA and the SPCA, which are model-agnostic. reduced the dimension to $q = 9$. This tells us that the initially-fitted mixture model can help with improving the efficiency of the projection. Figure 4.5 indicates that the estimated clusters are well-separated by the SENDV. The GMMDR also produced good separation, though not as clear as the SENDV. The PCA-GMM and SPCA-GMM do not appear to respect the clusters, as shown by the significant overlaps between the estimated

clusters. Overall, this illustration demonstrates the importance of introducing the clustering information to dimension reduction of the data set, and the benefit of regularization therein.

4.4 Discussion

In this chapter, the row-wise and column-wise Stiefel Elastic Net Discriminant Variable are introduced (rSENDV and cSENDV respectively). Equipped with flexible penalization, attractive theoretical bound and accessible estimation procedure, the SENDV algorithms allow the user to identify a simpler explanation on the clustering structure generated by the GMM in comparison to some of the existing methods. Indeed, cluster-preserving projection is not a problem unique to the GMM, as there are many non-Gaussian finite mixtures available in literature. Moreover, more advanced penalty functions have been developed in the recent past such as the SCAD by Fan and Li (2001) and MC+ by Zhang et al. (2010). Therefore, the potential for future work on the SENDV lies in its extension to non-Gaussian finite mixtures, as well as the adaptation to other modern penalization schemes. In addition, a further analysis into the SENDV's convergence properties could yield a tailored convergence criterion.

Figure 4.4: Back-to-back bar graphs plotting the discriminant matrix entry magnitudes. For instance, the top plot compares the rSENDV ($\alpha = 1$) (in blue) against GMMDR (in red). The left panel plots the absolute value of the entries in the first column of the unscaled and unit-normed discriminant matrix from each method, and the right panel plots that in the second column. From the top, each blue-red bar pair corresponds to year, weight, mpg, horsepower, displacement, and acceleration. The two methods estimated similar discriminant matrices, as shown by the closely-matching bars for each variable. The remaining plots are interpreted in a similar manner. The second, third and bottom plots compare rSENDV ($\alpha = 1$) (in blue) against LDA, SPCA and PCA (in red) respectively.

Chapter 5

Anderson Relaxation Test for Intrinsic Dimension Selection in Model-based Clustering

5.1 Introduction

Modern data analysis often demands the accommodation of high-dimensional observations, and model-based clustering is no exception. Finite mixtures of multivariate parametric distributions usually involve a positive definite scale matrix, whose number of entry increases in the quadratic order of the number of variables. This growth exposes the user to numerical instability in computation and possibly degenerate model parameter estimates, unless a commensurate number of observations is supplied (which is often not the case). To combat this reality, many parsimonious finite mixtures that curb the growth of the parameter count have been developed (Ghahramani and Hinton, 1996; Bouveyron et al., 2007; McNicholas and Murphy, 2008; Vrbik and McNicholas, 2014; Andrews and McNicholas, 2011; Murray et al., 2014; Tortora et al., 2016; Murray et al., 2020; Sharp and Browne, 2021; Kim and Browne, 2019, 2021b).

The subspace clustering method for the GMM by Bouveyron et al. (2007) is one such method, and it offers a great degree of flexibility by allowing a wide variety of assumptions to be imposed on the subspace structure of the data. However, like other parsimonious finite mixtures (Andrews and Mcnicholas, 2012; Kim and Browne, 2019; Ghahramani and Hinton, 1996; Tortora et al., 2016), the dimension of the subspace, also known as the intrinsic dimension, must be pre-determined. Intrinsic dimension estimation problem is not unique to the subspace clustering framework. It appears in numerous contexts (Takens et al., 1985; Fukunaga and Olsen, 1971; Trunk, 1976; Pestov, 2008), but they all strive to represent the data using a minimal number of dimensions. In subspace clustering, the intrinsic dimensions are estimated via the scree test (Cattell, 1966) or the Bayesian Information Criterion (BIC) (Schwarz, 1978). The former, while faster, needs a cutoff threshold to be pre-determined, and its selection process can be ad-hoc. The latter, while free of hyper-parameters, could be computationally prohibitive in high dimensions due to the large number of model log-likelihood evaluations. Therefore, the current status of literature leaves a gap for a middle-ground approach that is more principled yet computationally viable.

5.1.1 Intrinsic Dimension Estimation

Intrinsic dimension's definitions are context-dependent, among which are Takens et al. (1985); Fukunaga and Olsen (1971); Trunk (1976); Pestov (2008), but they describe commonly the smallest number of dimensions the data can be sufficiently compressed into. Estimating the intrinsic dimension of the data has grown in popularity due to the aforementioned dominance of high-dimensional data sets. Among others, contributions to this topic include Cattell (1966); Fukunaga and Olsen (1971); Bruske and Sommer (1998); Camastra (2003); Levina and Bickel (2005); Carter et al. (2009); Fan et al. (2010); Johnsson et al. (2014). In model-based clustering, intrinsic dimensions can be defined via the subspaces associated with the component-wise scale matrices $\boldsymbol{\Sigma}_g$, like in Bouveyron et al. (2007); Pesevski et al. (2018); Kim and Browne (2019), because numerous density functions can be written as a function of the squared Mahalanobis distance $(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)$ associated with $\boldsymbol{\Sigma}_g$ and a location parameter $\boldsymbol{\mu}_g$. As the Mahalanobis distance can be viewed

as the projection of observations onto the orthonormal basis generated by $\mathbf{\Sigma}_g$. The intrinsic dimension (of component g) can then be estimated by counting the number of significant directions, where the direction's significance is quantified by the associated eigenvalue. Reducing dimensionality via linear projection is well-understood (dating back to the Principal Component Analysis (PCA) by Pearson (1901)), and it offers a tidy geometric interpretation. However, to our best knowledge, selecting the dimension of the projection space remains ad-hoc (such as the scree test by Cattell (1966)) or computationally intensive (such as evaluating a model selection criterion at each possible dimension).

5.1.2 Intrinsic Dimension Selection in SC-GMM

The two existing methods for intrinsic dimension selection in the SC-GMM (outlined in chapter 2.1) are the scree test (Cattell, 1966) or the Bayesian Information Criterion (BIC) Schwarz (1978) (outlined in chapter 2.1). The scree test estimates the component-wise intrinsic dimension d_g by detecting where the consecutive difference between eigenvalues falls below a pre-determined threshold $\alpha_S > 0$. Let $\lambda_1 \geq \cdots \geq \lambda_p$ denote the eigenvalues of a covariance matrix $\mathbf{\Sigma}$, and let $\delta_j = \lambda_i - \lambda_{i+1}$ $(i = 1, \ldots, p-1)$ denote their consecutive differences. Then, The intrinsic dimension estimate $\hat{d}$ according to the scree test is defined as

$$\hat{d} = \min_{j=1,\ldots,p-1} \{j : \delta_j < \alpha_S\}. \tag{5.1}$$

While the scree test is quick and intuitive, selecting α_S is often an opaque process, unless the user tests along a pre-set grid of candidate values, hoping that a suitable value will lie within.

The BIC-based dimension selection would require the parameter estimates at every $(d_1, \ldots, d_G) \in \{1, \ldots, p-1\}^G$. This strategy is evidently computationally infeasible, so the software implementation of SC-GMM approximates the model's BIC by estimating d_g separately for each component, instead of a joint evaluation at $(d_1, \ldots, d_G)$. Hence, the current dichotomy leaves the user with an often-faster scree test with an ad-hoc threshold, or an approximation of BIC, whose rigourous version would be computationally prohibitive in

higher dimensions. This observation reiterates a gap to be mended with a more principled, yet computationally feasible, strategy.

Thus, in this chapter, we address this unmet need with a novel, hypothesis test-based, intrinsic dimension selection method. Section 5.2 will introduce the methodology, and section 5.3 will demonstrate it using simulated and real-world data sets. Finally, we will conclude with a brief discussion on our contribution and future directions.

5.2 Methodology

In this section we present the Anderson Relaxation Test (ART), which is a hypothesis test-based intrinsic dimension estimation method for the submodels of SC-GMM. The ART is parametrized by a single interpretable threshold, and it is tailored to each submodel offered by the SC-GMM. The tailoring is done via a two-pronged approach based on the inter-component dependence structure. We begin by introducing the single-component test, then leverage it to build a test for different submodels.

5.2.1 Single-component Test

Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be an IID sample from a p-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Anderson (2003) describes a likelihood ratio test for the hypothesis that $\boldsymbol{\Sigma}$ is equal to a given matrix. In particular, Anderson (2003) notes the unbiased version of it developed by Sugiura and Nagao (1968), which we will adopt in our work. Because the intrinsic dimension of a component in SC-GMM is determined by the number of distinguishable eigenvalues in its covariance matrix, we can test the adequacy of a given intrinsic dimension value for the data set using the equality-of-covariance test mentioned above. The null and alternative hypotheses can be stated as

$$H_0\text{: The intrinsic dimension is } d.$$
$$H_1\text{: The intrinsic dimension is not } d.$$

To translate this into the SC-GMM language, for a fixed submodel, let $\boldsymbol{PDP}'$ and $\boldsymbol{PD}_d\boldsymbol{P}'$ denote the unrestricted eigen-decomposition of $\boldsymbol{\Sigma}$ and its submodel-based alternative with intrinsic dimension equal to d, respectively. Then $\boldsymbol{D} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)$ and $\boldsymbol{D}_d = \text{diag}(a_1, a_2, \ldots, a_d, \underbrace{b, \ldots, b}_{p-d \text{ copies}})$. The above hypotheses can be expressed algebraically as

$$H_0 : \boldsymbol{\Sigma} = \boldsymbol{PD}_d\boldsymbol{P}' \quad \text{vs} \quad H_1 : \boldsymbol{\Sigma} \neq \boldsymbol{PD}_d\boldsymbol{P}', \tag{5.2}$$

and the corresponding test statistic T_d is written as

$$T_d = \left(\frac{e}{n-1}\right)^{p(n-1)/2} \det\left(n\boldsymbol{\Sigma}\boldsymbol{\Sigma}_d^{-1}\right)^{(n-1)/2} \exp\left\{\frac{-1}{2}\text{tr}\left(n\boldsymbol{\Sigma}\boldsymbol{\Sigma}_d^{-1}\right)\right\}.$$

Properties of the determinant and the trace cancel out the orientation matrix $\boldsymbol{P}$, so $-\log T_d$ can be written as

$$-2\log T_d = -p(n-1)\log\left(\frac{ne}{n-1}\right) - (n-1)\sum_{i=1}^{p}\log\left(\frac{[\boldsymbol{\Delta}]_{ii}}{[\boldsymbol{\Delta}_d]_{ii}}\right) + n\sum_{i=1}^{p}\frac{[\boldsymbol{\Delta}]_{ii}}{[\boldsymbol{\Delta}_d]_{ii}}, \tag{5.3}$$

where $[\boldsymbol{A}]_{ii}$ denotes the i^{th} diagonal entry of a matrix $\boldsymbol{A}$. Anderson (2003) showed that $-2\log T_d$ converges in distribution to χ^2 distribution with the degrees of freedom (df_d) equal to the difference in the number of free parameters between $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}_d$. Then, the intrinsic dimension selection strategy is to test the hypothesis at each $d = 1, 2, \ldots, p-1$ until the null hypothesis is not rejected, given a pre-determined critical level $\alpha \in (0, 1)$. The estimated intrinsic dimension $\hat{d}$ can be written mathematically as

$$\hat{d} = \min_{1,2,\ldots,p-1}\left\{d : -2\log T_d \leq \chi^2_{df_d,1-\alpha}\right\}, \tag{5.4}$$

where $\chi^2_{df_d,1-\alpha}$ denotes the $(1-\alpha)100^{\text{th}}$ percentile of the $\chi^2_{df_d}$ distribution. From the single component test, we will build the multi-component intrinsic dimension selection tests for various SC-GMM submodels in sections 5.2.2 and 5.2.3.

5.2.2 Submodels without Inter-component Sharing

The submodels in SC-GMM can be divided into two groups; one where the components share estimates and one where they do not share. The non-sharing submodels are $[a_{gj}b_g\boldsymbol{\Gamma}_g d_g]$ and $[a_g b_g \boldsymbol{\Gamma}_g d_g]$, and the rest share at least one aspects of the four (a, b, $\boldsymbol{\Gamma}$, d) between the components. This parameter sharing needs to be accounted for when building a multi-component test, and we begin with the non-sharing submodels.

For the non-sharing submodels $[a_{gj}b_g\boldsymbol{\Gamma}_g d_g]$ and $[a_g b_g \boldsymbol{\Gamma}_g d_g]$, we propose a component-wise test for the intrinsic dimensions $(d_1, d_2, \ldots, d_G)$, because each component has its own set of covariance parameters. Let $-2\log T_{gd}$ denote the g^{th} component analogue of the test statistic 5.3, where n, $\boldsymbol{D}$ and $\boldsymbol{D}_d$ are replaced by n_g, $\boldsymbol{D}_g = \text{diag}(\lambda_{g1}, \ldots, \lambda_{gp})$ and $\boldsymbol{D}_{gd} = \text{diag}(a_{g1}, \ldots, a_{gd_g}, \underbrace{b, \ldots, b}_{p-d_g \text{copies}})$ respectively. For the $[a_g b_g \boldsymbol{\Gamma}_g d_g]$ submodel, $a_{g1} = \cdots = a_{gd_g}$. Then, for $g = 1, 2, \ldots, G$, the intrinsic dimension is estimated by

$$\hat{d}_g = \min_{1,2,\ldots,p-1} \left\{ d : -2\log T_{gd} \leq \chi^2_{df_d, 1-\alpha} \right\}, \tag{5.5}$$

where we use the same α across all components for ease of tuning and interpretation.

5.2.3 Submodels with Inter-component Sharing

The remaining submodels share some, or all, of the covariance parameters. This inter-component dependence invalidates the component-wise approach given in section 5.2.2. Furthermore, conventional multiple testing remedies like the Bonferroni correction may be inappropriate since the tests are not independent. Hence, we adopt the harmonic mean p-value (HMp) by Wilson (2019), which is a multiple comparison technique for dependent tests. Given the p-values of G many individual tests $p_1, \ldots, p_G$, the HMp is defined as

$$\text{HM}p = \frac{\sum_{g=1}^{G} w_g}{\sum_{g=1}^{G} (w_g / p_g)}, \tag{5.6}$$

where $w_g > 0$ $(g = 1, \ldots, G)$ are the pre-determined test-wise weights that sum to 1. Higher weight w_g corresponds to heavier prior belief on the null hypothesis being false. Since no such prior information is available to us, we let $w_1 = \cdots = w_G$. The null and alternative hypotheses under consideration are

$$H_0\text{: The intrinsic dimension of components } 1, \ldots, G \text{ are } d_1, \ldots, d_G \text{ respectively.}$$
$$H_1\text{: The intrinsic dimension of components } 1, \ldots, G \text{ are not } d_1, \ldots, d_G \text{ respectively.}$$

Using the HMp, an iterative intrinsic dimension selection procedure can be constructed. The high-level idea is as follows. For a given vector of intrinsic dimensions $(d_1, \ldots, d_G)$, the component-wise p-values $p_1, \ldots, p_G$ are computed using $-2 \log T_{1d_1}, \ldots, -2 \log T_{Gd_G}$. If the resultant HM$p$ is above the critical level α, then the current dimension vector $(d_1, \ldots, d_G)$ is returned. Otherwise, the intrinsic dimension of the component with the lowest p-value is raised by 1. This process is iterated until either HMp exceeds α or $d_g = p - 1$ for all $g = 1, \ldots, G$. The full algorithm is provided below.

1. Initialize $\alpha \in (0, 1)$ and set $(d_1, \ldots, d_G) = (\underbrace{1, \ldots, 1}_{G \text{ copies}})$.
2. For $g = 1, \ldots, G$:
   (a) Compute the component-wise test statistic $-2 \log T_{gd_g}$.
   (b) Compute the component-wise p-value, $p_g = P\left(-2 \log T_{gd_g} > \chi^2_{df_{d_g}, 1-\alpha}\right)$, where df_{d_g} denotes the component-wise degrees of freedom evaluated at the intrinsic dimension d_g.
3. Compute HM$p = G / \sum_{g=1}^{G} (1/p_g)$.
4. If HM$p \geq \alpha$: return current $(d_1, \ldots, d_G)$.
5. Else if all $d_g = p - 1$: return current $(d_1, \ldots, d_G)$.
6. Else if the submodel holds component-wise intrinsic dimensions to be equal $(d_1 = \cdots = d_G)$:

(a) $d_g \leftarrow d_g + 1$ for all $g = 1, \ldots, G$.

(b) Return to step 2.

7. Else:

(a) Let $g^\star$ be the lowest index such that $p_{g^\star} = \min\{p_1, \ldots, p_G\}$.

(b) $d_{g^\star} \leftarrow d_{g^\star} + 1$.

(c) Return to step 2.

5.2.4 Note on the Degrees of Freedom

Because the degrees of freedom in the ART depends on the intrinsic dimension, it is also submodel-dependent. In particular, integer-valued degrees of freedom may be inappropriate for the sharing submodels in section 5.2.3, since a single free parameter estimated collectively by multiple components. Non-integer degrees of freedom appears in numerous modern techniques such as the locally weighted regression (Cleveland, 1981) and smoothing methods (Friedman et al., 2001). Thus, in this section, we describe our degrees of freedom calculation approach and tabulate the values for each submodel.

It is most straightforward to begin with the most flexible submodel, $[a_{gj} b_g \mathbf{\Gamma}_g d_g]$. A $(p \times p)$-dimensional covariance matrix has $p(p+1)/2$ free parameters, and in a single component of the said submodel, there are d_g and 1 distinguishable and indistinguishable eigenvalues, and $pd_g - d_g(d_g - 1)/2$ free parameters from the truncated orientation matrix $\mathbf{\Gamma}_g$. Therefore, the component-wise degrees of freedom at intrinsic dimension d_g is $p(p+1)/2 - (d_g + 1) - (pd_g - d_g(d_g + 1)/2)$. Similarly, for the $[a_g b_g \mathbf{\Gamma}_g d_g]$ submodel, $d_g + 1$ changes to $1 + 1$.

For the sharing submodels, weighted fractional degrees of freedom are adopted to account for the shared parameters. In the SC-GMM framework, the aggregated eigenvalues a, b are represented weighted averages of unrestricted component-wise eigenvalues $\{\lambda_{g1}, \ldots, \lambda_{gp}\}$ $(g = 1, \ldots, G)$. For example, the pooled eigenvalues a_g, a, a_j and b are

written below.

$$a_j = \frac{1}{d}\sum_{j=1}^{d}\bar{\lambda}_j, \quad a = \frac{1}{\sum_{g=1}^{G}\pi_g d_g}\sum_{g=1}^{G}\pi_g\sum_{j=1}^{d_g}\lambda_{gj}, \quad b = \frac{1}{\sum_{g=1}^{G}\pi_g d_g}\sum_{g=1}^{G}\pi_g\sum_{j=d_g+1}^{p}\lambda_{gj},$$

where d denotes the common intrinsic dimension across the components, and $\bar{\lambda}_1, \ldots, \bar{\lambda}_p$ denote the eigenvalues of the weighted sum of component-wise covariance matrices $\pi_1\boldsymbol{\Sigma}_1 + \cdots + \pi_G\boldsymbol{\Sigma}_G$, as the $[a_j \cdots]$ submodels require a common d. The above formulae reveals that higher d_g implies more involvement from component g in computing a, and lower d_g and b hold a similar relationship. Therefore, the components with more involvement are given higher weights. The component-wise fractional degrees of freedom for a and b are given respectively by

$$df_g^{(a)} = \underbrace{\left(\frac{\pi_g d_g}{\sum_{g=1}^{G}\pi_g d_g}\right)}_{\text{weight}} \times \underbrace{1}_{\text{df for } a}, \quad df_g^{(b)} = \underbrace{\left(\frac{\pi_g (p-d_g)}{\sum_{g=1}^{G}\pi_g (p-d_g)}\right)}_{\text{weight}} \times \underbrace{1}_{\text{df for } b}. \tag{5.7}$$

For a_j and $\boldsymbol{\Gamma}$, the component-wise fractional degrees of freedom are given in equation 5.8 below. For $df_g^{(\Gamma)}$, d_g is understood as d for $[\cdots d]$ submodels.

$$df_g^{(a_j)} = \left(\frac{\pi_g d}{\sum_{g=1}^{G}\pi_g d}\right) \times d = \pi_g d, \quad df_g^{(\Gamma)} = \left(\frac{\pi_g d_g}{\sum_{g=1}^{G}\pi_g d_g}\right) \times \left(d_g p - \frac{d_g(d_g+1)}{2}\right). \tag{5.8}$$

The discussion in the section is summarized in table 5.1, which contains the component-wise degrees of freedom for each submodel in the SC-GMM.

Submodel	a	b	Fractional	$df_g(a, b, \mathbf{\Gamma})$
$[a_{gj} b_g \mathbf{\Gamma}_g d_g]$	$\sum_g d_g$	G	$-$	$df^{(\Gamma)} + d_g + 1$
$[a_{gj} b \mathbf{\Gamma}_g d_g]$	$\sum_g d_g$	1	b	$df^{(\Gamma)} + d_g + df_g^{(b)}$
$[a_g b_g \mathbf{\Gamma}_g d_g]$	G	G	$-$	$df^{(\Gamma)} + 2$
$[a b_g \mathbf{\Gamma}_g d_g]$	1	G	a	$df^{(\Gamma)} + df_g^{(a)} + 1$
$[a_g b \mathbf{\Gamma}_g d_g]$	G	1	b	$df^{(\Gamma)} + 1 + df_g^{(b)}$
$[a b \mathbf{\Gamma}_g d_g]$	1	1	a, b	$df^{(\Gamma)} + df_g^{(a)} + df_g^{(b)}$
$[a_{gj} b_g \mathbf{\Gamma}_g d]$	Gd	G	$-$	$df^{(\Gamma)} + d + 1$
$[a_{gj} b \mathbf{\Gamma}_g d]$	Gd	1	b	$df^{(\Gamma)} + d + df_g^{(b)}$
$[a_g b_g \mathbf{\Gamma}_g d]$	G	G	$-$	$df^{(\Gamma)} + 2$
$[a b_g \mathbf{\Gamma}_g d]$	1	G	a	$df^{(\Gamma)} + df_g^{(a)} + 1$
$[a_g b \mathbf{\Gamma}_g d]$	G	1	b	$df^{(\Gamma)} + 1 + df_g^{(b)}$
$[a b \mathbf{\Gamma}_g d]$	1	1	a, b	$df^{(\Gamma)} + df_g^{(a)} + df_g^{(b)}$
$[a_j b_g \mathbf{\Gamma}_g d]$	d	G	a	$df^{(\Gamma)} + df_g^{(a_j)} + 1$
$[a_j b \mathbf{\Gamma}_g d]$	d	1	a	$df^{(\Gamma)} + df_g^{(a_j)} + df_g^{(b)}$

Table 5.1: Table of SC-GMM submodels and their degrees of freedom for the eigenvalues and orientation matrices. The a and b columns record the number of free parameters for the distinguishable and indistinguishable eigenvalues respectively. The 'Fractional' column records which eigenvalues are given fractional degrees of freedom. The $df_g(a, b, \mathbf{\Gamma})$ column records the component-wise number of free parameters from the eigenvalues and the orientation matrix. $df^{(\Gamma)} = d_g p - d_g(d_g + 1)/2$ and d_g is understood as d for $[\cdots d]$ submodels. For the common-orientation ($\mathbf{\Gamma}_g = \mathbf{\Gamma}$) submodels, $df^{(\Gamma)}$ is replaced with $df_g^{(\Gamma)}$ from equation 5.8.

5.3 Numerical Experiments

In this section, we study the performance of the ART, and compare it to some of the existing intrinsic dimension estimation methods, using simulated and real data sets. The following estimation methods are considered.

- **ART**: Anderson Relaxation Test parametrized by α_A.
- **Scree**: Scree test parametrized by α_S.

- **BIC**: Bayesian Information Criterion-based estimation as described in chapter 2.1.

- **LPCA**: Intrinsic dimension estimation via Local Principal Component Analysis by Fan et al. (2010). The data set is partitioned into K subsets via a nearest neighbour algorithm. Then, for each $k = 1, \ldots, K$, the eigenvalues $\lambda_{k1}, \ldots, \lambda_{kp}$ from the sample covariance of the k^{th} subset are computed. Let $\lambda_j = \sum_{k=1}^{K} \lambda_{kj}$. For pre-determined threshold values $s > 1$ and $t \in (0, 1)$, two intrinsic dimension estimates $\hat{d}_s, \hat{d}_t$ are obtained as below. The final estimate $\hat{d}$ is equal to $\min\{\hat{d}_s, \hat{d}_t\}$.

$$\hat{d}_s = \underbrace{\underset{d=1,\ldots,p}{\operatorname{argmin}} \left\{ \frac{\min\limits_{i=1,\ldots,d} \lambda_i}{\max\limits_{j=d+1,\ldots,p} \lambda_j} > s \right\}}_{\text{Comparing the first } d \text{ against the rest}}, \quad \hat{d}_t = \underbrace{\underset{d=1,\ldots,p}{\operatorname{argmin}} \left\{ \frac{\min\limits_{i=1,\ldots,d} \lambda_i}{\max\limits_{j=1,\ldots,p} \lambda_j} > t \right\}}_{\text{Comparing the first } d \text{ against all}}.$$

- **ESS**: Intrinsic dimension estimation from the Expected Simplex Skewness by Johnsson et al. (2014). Given a pre-determined parameter $t \in \{1, \ldots, n\}$, the set of all simplices with $t + 2$ vertices (1 vertex at the centroid of the data, and the other $t + 1$ observations as remaining vertices) are obtained, and a weighted average m of their volumes is computed. The d-dimensional Expected Simplex Skewness (ESS(d)) a theoretical value of m under the uniform distribution assumption of the observations over a d-dimensional unit ball. The simplex skewness measure is then defined as $m/\text{ESS}(d)$. The intrinsic dimension estimate $\hat{d}$ is d such that $m/\text{ESS}(d)$ is approximately 1.

- **OTPM**: Optimally Topology-preserving Maps by Bruske and Sommer (1998). Voronoi cells on the observations are used to construct a graph on the the data set. Then, Local PCA is applied on the graph to obtain the intrinsic dimension estimate at each observation. In this paper, the intrinsic dimension estimate $\hat{d}$ for the data set is defined as the (rounded) median of the point-wise dimension estimates.

- **kNN**: Weighted Average k-Nearest Neighbour Distances by Carter et al. (2009). Several bootstrap samples are obtained from the data set, and for each sample, the total edge length of its kNN graph is computed. Denote the resultant vector

of total kNN edge lengths as $\boldsymbol{L} = (L_1, \ldots, L_B)$. The objective is the least-square minimisation between $\boldsymbol{L}$ and its asymptotic form $\boldsymbol{L}(d)$, parametrized by an integer d. $\hat{d}$ is the minimizer of the said least-square objective with respect to d.

The list of experiments and their study objectives are given below.

- **Simulated data from 1-component GMM**: The ART and the scree test are compared at various levels of α_A and α_S. The change in estimation behaviour based on the threshold is studied.
- **Simulated data from 2-component GMM**: Selected submodels of various flexibility from the ART are compared. The relationship between the submodel and the intrinsic dimension estimates is studied.
- **Simulated data from 3-component GMM**: Intrinsic dimension recovery and computational speed from all methods are compared at increasing sample sizes.
- **Real data illustration: Bankruptcy**: A real-data illustration of the SC-GMM paired with (ART, Scree, BIC) is presented using the Company Bankruptcy data from Kaggle (Liang and Tsai, 2016).

The LPCA, ESS, OTPM and kNN estimation methods are implemented in the R package *intrinsicDimension* (Johnsson and University, 2019). For the four methods, the default parameter values were used whenever they exist. Exceptions are: OTPM (number of graph nodes $N = 100$; no default), and kNN (number of bootstrap samples for each sample size $M = 5$; default value is too slow). The computational speed of each method was measured on an Intel Xeon Gold 6150 processor, clocked at 2.70GHz.

5.3.1 Simulation: 1-component GMM

In this experiment, the ART and the scree test are compared, as one of ART's aims is to improve upon the scree test, and they are relatively easy for a direct comparison due to their parametrization. Samples of size $n = 100, 200, \cdots, 1000$ are generated

from a 50-dimensional Gaussian distribution with zero mean and a diagonal covariance matrix whose diagonal entries are designed to have the true intrinsic dimension of 10: $(seq(5, 4, 10), \underbrace{0.5, \cdots, 0.5}_{40 \text{ copies}})$, where $seq(a, b, c)$ denotes an equi-distant sequence of length c, from a to b. For each generated sample, the intrinsic dimension is estimated using the ART and the scree test. The considered submodels for the ART are $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$ and $[a_g b_g \mathbf{\Gamma}_g d_g]$. The ordered threshold values α_A and α_S are given below, where the method's affinity toward higher dimensions increases from left to right.

$$\alpha_A = 0.001, 0.01, 0.01, 0.05, 0.1, \quad \alpha_S = 0.2, 0.1, 0.05, 0.01, 0.001.$$

Overall, for each simulated sample, 15 intrinsic dimension estimates are computed. Finally, for each n, the above process is replicated 500 times, with a newly-generated sample for each replication.

$n =$	100	200	300	400	500	600	700	800	900	1000
ART$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$										
$\alpha_A = 0.0001$	20	9	8	8	8	8	8	8	8	8
	(10)	(1)								(1)
$\alpha_A = 0.001$	24	9	8	8	8	8	8	8	8	8
	(11)									
$\alpha_A = 0.01$	30	9	8	8	8	8	8	8	8	8
	(11)									
$\alpha_A = 0.05$	35	9	8	8	8	8	8	8	8	8
	(11)		(1)							
$\alpha_A = 0.1$	37	9	9	8	8	8	8	8	8	8
	(10)		(1)							
ART$[a_g b_g\mathbf{\Gamma}_g d_g]$										
$\alpha_A = 0.0001$	49	9	8	8	8	8	8	8	8	8
		(1)						(1)	(1)	(1)
$\alpha_A = 0.001$	49	9	8	8	8	8	8	8	8	8
$\alpha_A = 0.01$	49	9	8	8	8	8	8	8	8	8
$\alpha_A = 0.05$	49	9	8	8	8	8	8	8	8	8
		(1)								
$\alpha_A = 0.1$	49	9	9	8	8	8	8	8	8	8
		(1)								
Scree										
α_S =0.2	10	10	10	10	10	10	10	10	10	10
α_S =0.1	10	10	10	10	10	10	10	10	10	10
	(1)									
α_S =0.05	14	10	10	10	10	10	10	10	10	10
	(5)							(1)	(1)	(1)
α_S =0.01	48	48	46	35	22	15	12	11	11	10
	(2)	(3)	(12)	(26)	(32)	(10)	(5)	(3)	(2)	(1)
α_S =0.001	49	49	49	49	49	49	49	49	49	49

Table 5.2: Table of median (and inter-quartile range (IQR)) intrinsic dimension estimates for the ART$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$, ART$[a_g b_g\mathbf{\Gamma}_g d_g]$ and scree test. If the IQR for a given setting is non-zero, then it is written underneath the median, within brackets. The column names denote the sample size at which the data was generated, and the row names within each sub-table denote the threshold level used. For example, the first row of the first sub-table records the results obtained from ART$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$ with threshold $\alpha_A = 0.0001$.

Table 5.2 presents the median and the inter-quartile range (IQR) of the intrinsic dimension estimates under each method, across all considered sample sizes. There are three sub-tables separated by horizontal blank spaces: the top is for the ART$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$, the middle is for the ART$[a_g b_g \mathbf{\Gamma}_g d_g]$ and the bottom is for the scree test. In each sub-table, the row labels (leftmost column) denotes the threshold value used for the estimates. The threshold values are arranged (from top to bottom) in the order of increasing affinity toward higher dimensions.

Both ART submodels' estimates converged downward to $\hat{d} = 8$, which is lower than the true value of 10, where as that of the scree test converged downward to $\hat{d} = 10$ for all threshold values except $\alpha_S = 0.001$. The direction of convergence suggests that both the ART and the scree test tend to over-estimate the dimensions in small sample sizes, but gradually decrease as more information is obtained. Finally, the IQR values indicate that the ART's estimates are more consistent than that of the scree test. Thus, in this experiment, the ART offers an increased robustness in threshold selection at the cost of a mild under-estimation of intrinsic dimension. This implies a practical advantage, since the risk associated with sub-optimal threshold selection is virtually one-sided for the ART (under-estimation), in contrast to that of the scree test, which is two-sided (over or under-estimation). For instance, as long as the threshold value is sufficiently small (≈ 0.001), the chance of over-estimation would diminish. Moreover, if model parsimony is prioritized, under-estimation may be preferred to over-estimation. Overall, this experiment indicates that the ART could be a convenient alternative to the scree test.

5.3.2 Simulation: 2-component GMM

In this experiment, four submodels of the ART ($[a_{gj}b_g\mathbf{\Gamma}_g d_g]$, $[a_g b_g \mathbf{\Gamma}_g d_g]$, $[ab_g\mathbf{\Gamma}_g d_g]$ and $[ab\mathbf{\Gamma}_g d_g]$) are studied to examine the intrinsic dimension recovery under model mismatch. The data set is generated by a 50-dimensional 2-component GMM with zero mean for both components, under which there are two scenarios. The list below describes each scenario.

1. Covariance parameters follow $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$ submodel ($d_1 = 5$, $d_2 = 15$). The data-

generating covariance eigenvalues are given below.

$$(a_{1,1}, \cdots, a_{1,5}) = seq(5, 4, 5), \quad (b_{1,6}, \cdots, b_{1,50}) = (0.5, \cdots, 0.5),$$
$$(a_{1,1}, \cdots, a_{1,15}) = seq(10, 8, 15), \quad (b_{1,16}, \cdots, b_{1,50}) = (1, \cdots, 1).$$

2. Covariance parameters follow $[a_{gj} b_g \mathbf{\Gamma}_g d]$ submodel ($d = 10$). The data-generating covariance eigenvalues are given below.

$$(a_{1,1}, \cdots, a_{1,10}) = seq(5, 4, 10), \quad (b_{1,11}, \cdots, b_{1,50}) = (0.5, \cdots, 0.5),$$
$$(a_{1,1}, \cdots, a_{1,10}) = seq(10, 8, 10), \quad (b_{1,11}, \cdots, b_{1,50}) = (1, \cdots, 1).$$

The un-truncated component-wise orientation matrices $\boldsymbol{P}_1, \boldsymbol{P}_2$ are generated using the R package *pracma* (Borchers, 2021). The diagonal matrix with component-wise eigenvalues $\boldsymbol{D}_g$ is combined with $\boldsymbol{P}_g$ to create the data-generating component-wise covariance matrices $\boldsymbol{\Sigma} = \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g'$ for this experiment. For each generated data set under a fixed scenario, the component-wise intrinsic dimensions are estimated via the four ART submodels. This process is replicated 500 times for each scenario, and the two sets of 500 replications (for both scenarios) are obtained twice; once with component-wise sample size $n_1 = n_2 = 100$ (hence the sample size is 200), and once more with $n_1 = n_2 = 500$ (hence the sample size is 1000). Since the goal is intrinsic dimension estimation, not clustering, the true labels are used for parameter estimation.

Overall, there are 4 combinations of data-generating scenarios: (covariance, n_g) $\in \{(1, 100), (1, 500), (2, 100), (2, 500)\}$. For each (covariance, n_g) pair, 500 samples are generated, and four ART submodels are applied to each sample to estimate (d_1, d_2). α_A is set at 0.0001 for all submodels, based on the evidence presented in section 5.3.1.

$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$	$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$	$[a_g b_g\mathbf{\Gamma}_g d_g]$	$[ab_g\mathbf{\Gamma}_g d_g]$	$[ab\mathbf{\Gamma}_g d_g]$
$n_g = 100$	(21, 26)	(49, 49)	(49, 49)	(49, 49)
$n_g = 500$	(**5**, **15**)	(**5**, 49)	(49, 49)	(49, 49)
$[a_{gj}b_g\mathbf{\Gamma}_g d]$				
$n_g = 100$	(23, 23)	(49, 49)	(49, 49)	(49, 49)
$n_g = 500$	(**10**, **10**)	(**10**, **10**)	(49, 49)	(49, 49)

Table 5.3: Table of median $(\hat{d}_1, \hat{d}_2)$ estimates. It consists of two sub-tables separated by a blank horizontal line. The top sub-table records results for covariance scenario 1 $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$, and the bottom sub-table records those for covariance scenarios 2 $[a_{gj}b_g\mathbf{\Gamma}_g d]$. The row labels denote the the component-wise sample size used for the results in the same row. The median values corresponding to the true intrinsic dimension are bolded. The IQR is omitted because it was zero for all submodels except for $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$.

Table 5.3 shows that the intrinsic dimension estimates increase along with the rigidity of the considered submodel. At $n_g = 500$, the $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$ submodel succeeded in correctly identifying the component-wise intrinsic dimensions. The $[a_g b_g\mathbf{\Gamma}_g d_g]$ submodel estimated partially correctly the intrinsic dimensions. That is likely because the true distinguishable eigenvalues decrease slowly. Contrarily, the remaining two submodels produced maximal intrinsic dimensions. This indicates that excess rigidity in submodel can lead to over-estimation. This could be interpreted as a compensating behaviour for the restrictive parameter structure. Overall, this experiment suggests that, in the absence of relevant information, the $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$ submodel would be the best choice for the intrinsic dimension estimation.

5.3.3 Simulation: 3-component GMM

In this experiment, we compare the intrinsic dimension estimates and the computational costs of all seven methods. The samples are generated from a 100-dimensional 3-component

GMM with zero mean and the following covariance parameters $\{\boldsymbol{P}_g, \boldsymbol{D}_g\}_{g=1,2,3}$:

$$\begin{aligned}
&\boldsymbol{D}_1 = \text{diag}(seq(10, 9, 5), 1, \ldots, 1),\\
&\boldsymbol{D}_2 = \text{diag}(seq(5, 3, 15), 0.5, \ldots, 0.5),\\
&\boldsymbol{D}_2 = \text{diag}(seq(15, 13, 10), 5, \ldots, 5),\\
&\boldsymbol{P}_1, \boldsymbol{P}_2, \boldsymbol{P}_3 : \text{ randomly generated from R package } \textit{pracma}.
\end{aligned}$$

The considered component-wise sample sizes are $n_g = 200, 300, \ldots, 1000$ (equal sample size across components), and for each n_g, 500 samples are generated. To remove the potential confounding from incorrect labels, a 3-component GMM with true labels is fitted to each sample. Then, ART$[a_{gj} b_g \boldsymbol{\Gamma}_g d_g](\alpha_A = 0.0001)$, Scree$(\alpha_S = 0.2)$, BIC, LPCA, OTPM, ESS and kNN are applied on the covariance estimates to obtain the intrinsic dimensions $(\hat{d}_1, \hat{d}_2, \hat{d}_3)$. The elapsed time excludes the time taken to fit the GMM.

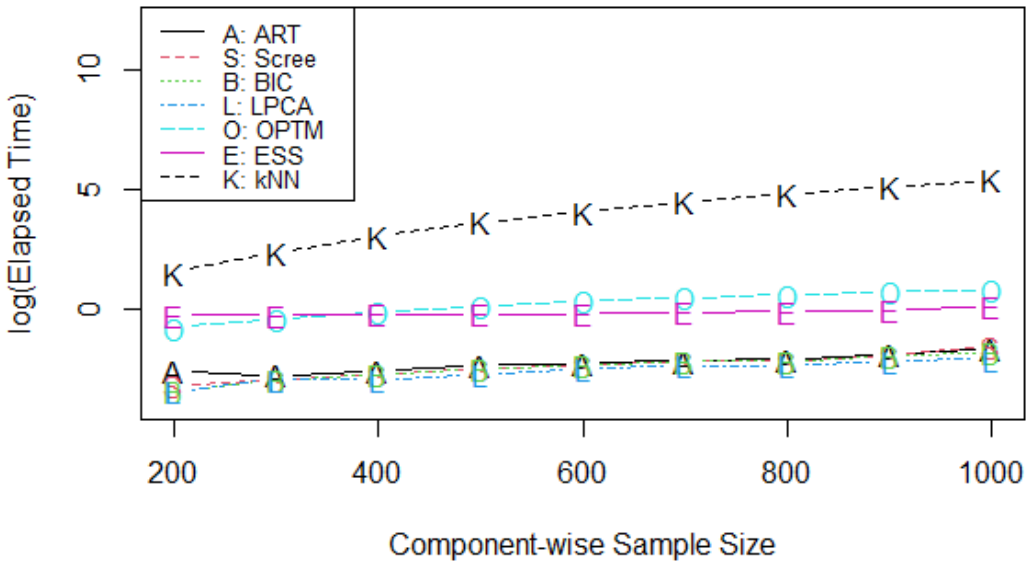

Figure 5.1: Line graph of logged median elapsed time against component-wise sample size for each method tested. The logging was necessary to examine all methods on a similar scale. The lines are letter-coded by the first letter of the methods' name.

$n_g \div 100$	2	3	4	5	6	7	8	9	10
ART	0.07	0.06	0.07	0.09	0.10	0.11	0.12	0.15	0.19
	(0.02)	(0.01)	(0.02)	(0.02)	(0.01)	(0.01)	(0.01)	(0.02)	(0.03)
Scree	0.04	0.05	0.06	0.08	0.09	0.11	0.11	0.14	0.20
	(0.02)	(0.01)	(0.01)		(0.02)		(0.01)	(0.01)	(0.03)
BIC	0.03	0.05	0.06	0.08	0.09	0.11	0.11	0.14	0.16
	(0.01)	(0.01)	(0.01)		(0.01)	(0.01)	(0.01)	(0.01)	(0.03)
LPCA	0.03	0.05	0.05	0.06	0.08	0.09	0.09	0.11	0.12
		(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)		(0.02)
OTPM	0.44	0.65	0.86	1.08	1.34	1.55	1.80	2.00	2.21
	(0.03)	(0.03)	(0.05)	(0.05)	(0.08)	(0.07)	(0.08)	(0.08)	(0.09)
ESS	0.75	0.76	0.78	0.78	0.80	0.85	0.89	0.94	1.05
	(0.05)	(0.05)	(0.06)	(0.06)	(0.06)	(0.07)	(0.06)	(0.05)	(0.06)
kNN	4.47	10.64	20.95	36.89	57.61	86.72	122.22	166.62	219.92
	(0.14)	(0.24)	(0.44)	(0.60)	(0.80)	(1.09)	(1.05)	(1.48)	(1.80)

Table 5.4: Table of median (and IQR in brackets underneath) elapsed time for each intrinsic dimension estimation method, rounded to two decimal places. If the rounded IQR is zero, then it is left as blank. The column labels denote $n_g \div 100$, and the row labels denote the estimation method.

Figure 5.1 visualizes the logged median elapsed time for estimation in each of seven methods against n_g. Three tiers of computational cost can be identified: low (ART, Scree, BIC, LPCA), medium (ESS, OTPM) and high (kNN). In particular, the LPCA appears to cost the least in most cases. In terms of growth against sample size, the medium group is the slowest, following narrowly by the low group and then by the high group, though the growth gap between the high group and the rest is the largest. This gap is expected because the kNN needs bootstrapping, though its IQR-to-median ratio is smaller compared to that of the other methods. More importantly, this figure shows that the ART is similarly fast as

the scree test and the BIC, implying its viability speed-wise as an alternative estimation method.

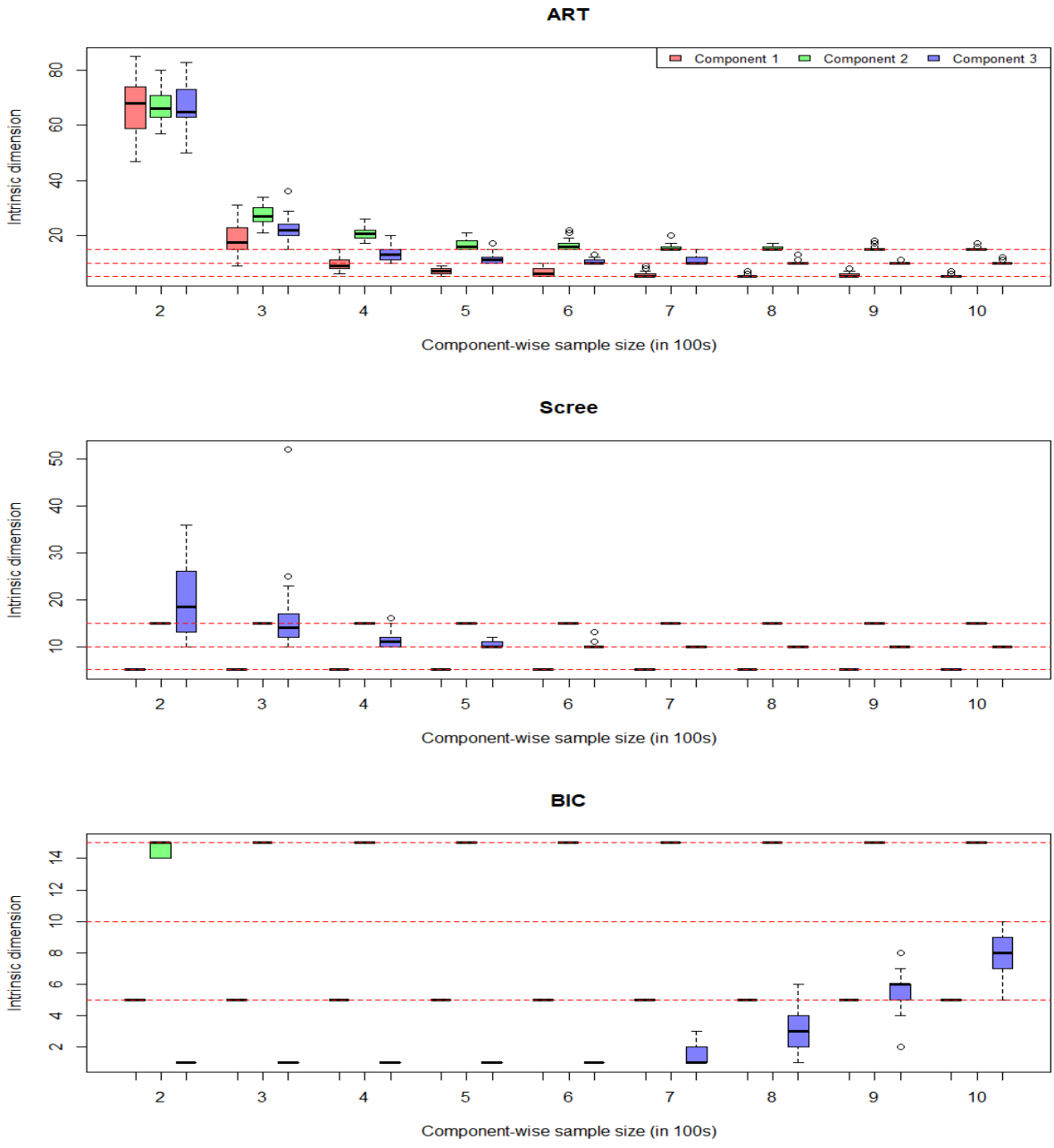

Figure 5.2: Grouped boxplot of median $\hat{d}_1, \hat{d}_2, \hat{d}_3$ values for the ART (top), scree test (middle) and BIC (bottom). In each plot, the horizontal axis denotes $n_g \div 100$, and the vertical axis denotes median($\hat{d}$). The dotted horizontal red lines mark the true component-wise intrinsic dimensions (5, 15, 10). The boxes are colour-coded by component.

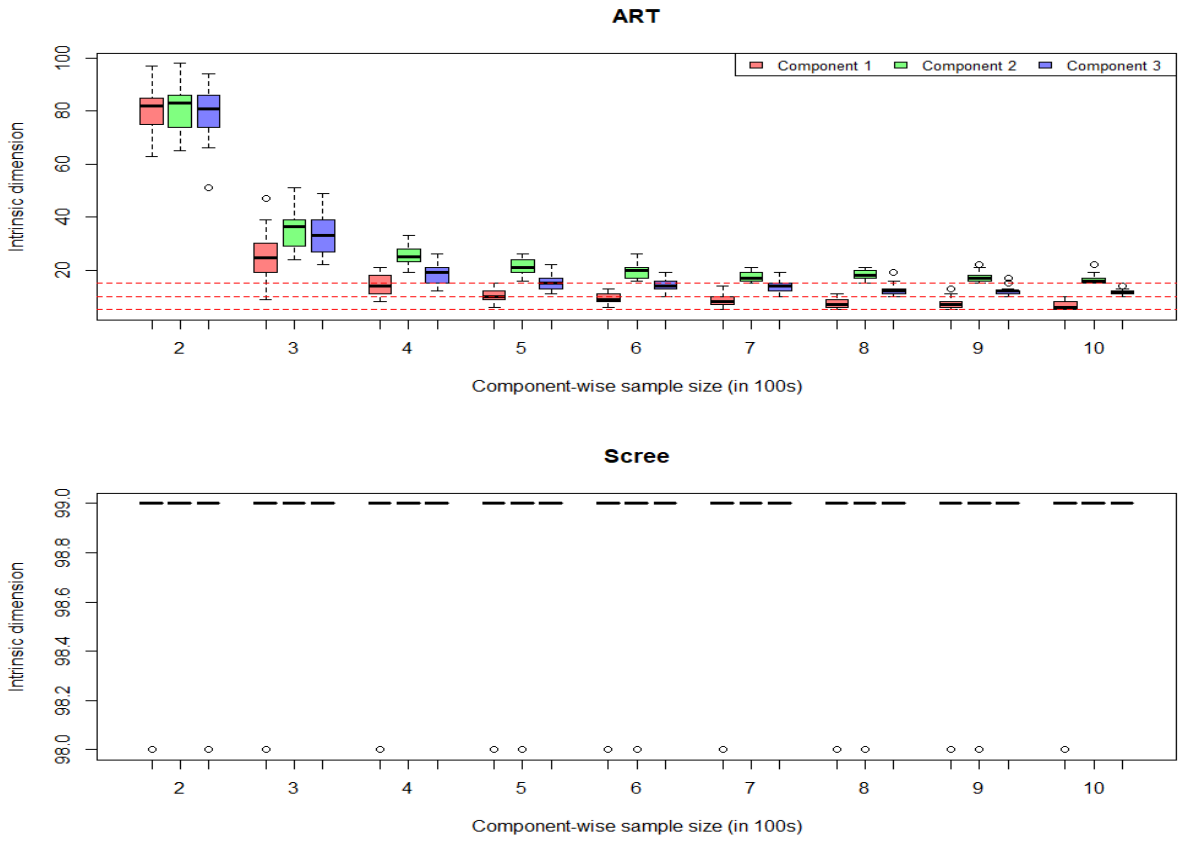

Figure 5.3: Grouped boxplot of median $\hat{d}_1, \hat{d}_2, \hat{d}_3$ values for the ART ($\alpha_A = 0.1$) (top) and scree test ($\alpha_S = 0.001$) (bottom). In each plot, the horizontal axis denotes $n_g \div 100$, and the vertical axis denotes median($\hat{d}$). The dotted horizontal red lines mark the true component-wise intrinsic dimensions (5, 15, 10). The boxes are colour-coded by component.

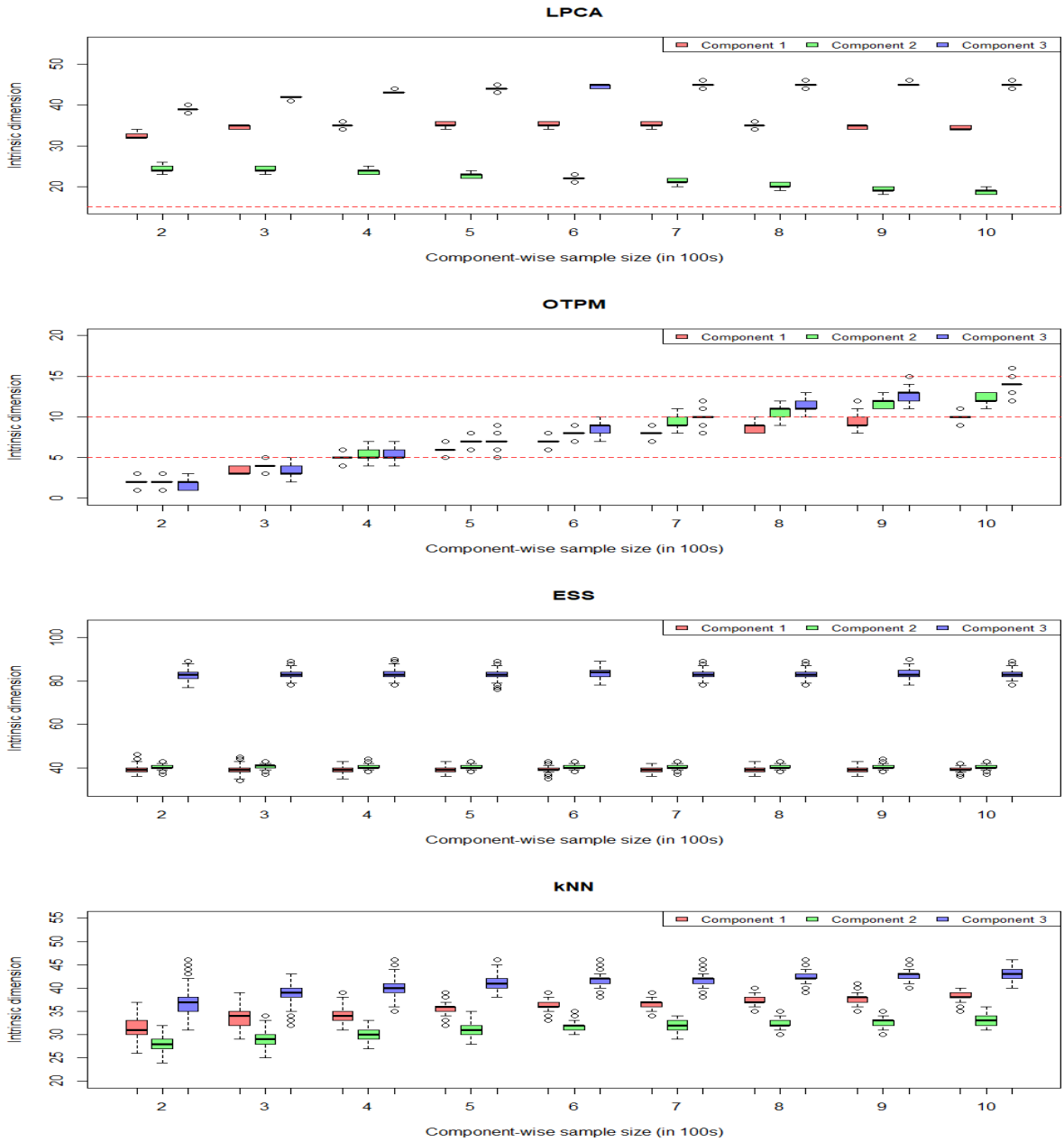

Figure 5.4: Grouped boxplot of median $\hat{d}_1, \hat{d}_2, \hat{d}_3$ values for the LPCA, OTPM, ESS and kNN (from top to bottom). In each plot, the horizontal axis denotes $n_g \div 100$, and the vertical axis denotes median($\hat{d}$). The dotted horizontal red lines mark the true component-wise intrinsic dimensions (5, 15, 10). The boxes are colour-coded by component.

Figures 5.2, 5.3 and 5.4 plot the boxplots of component-wise intrinsic dimension estimates against n_g for each tested method. The ART demonstrates a similar converging behaviour to section 5.3.1, where the estimates of all components arrive eventually at the true values marked by red dotted lines. The scree test and the BIC behave similarly, where the 5 and 15 intrinsic-dimensional components are estimated quite precisely, and a converging trend toward the true value is observed for the remaining component. However, the BIC seems to approach the true values slower than the ART, and section 5.3.1 tells us that the scree test's asymptotic behaviour is quite sensitive to the threshold α_S. Indeed, figure 5.3 shows that, when the threshold swings to a more conservative value ($\alpha_S = 0.001$), the scree test misses the marks completely. In contrast, the ART converges to the correct intrinsic dimension values even after a drastic change in threshold ($\alpha_A = 0.1$) similar to that of the scree test, reiterating its robustness in threshold selection. Out of the remaining methods, the ESS appears the most stable, though the estimates are quite far from the true values. This could be a parameter selection issue or the ESS could be chasing after a different form of intrinsic dimension. The OTPM and the kNN exhibit increasing trends in dimension estimates, though the OTPM hovers closer to the true intrinsic dimension values set in this experiment. the LPCA's estimates are fanning out as n_g increases, and their intended destinations are not clear even at $n_g = 1000$, which is a considerably large value. Overall, the ART is competitive speed-wise, and demonstrates asymptotic trends that are robust to changes in the threshold parameter.

5.3.4 Real Data Illustration: Bankruptcy

Here, we deploy the SC-GMM with the ART, scree test and BIC in a real-data setting. The Bankruptcy data set (Liang and Tsai, 2016) consists of 95 numeric-valued markers of financial health of a company, and there are 6819 companies present within. There are two classes present: bankrupt or not. Four intrinsic dimension selection methods are used: ART($\alpha_A = 0.0001$), Scree($\alpha_S = 0.2$), Scree($\alpha_S = 0.001$) and BIC. The number of components considered are $G = 1, 2, \ldots, 6$, and for each selection method, the SC-GMM is fitted based on 100 different k-means initialisations. The considered submodels

are $[a_{gj}b_g\mathbf{\Gamma}_g d_g]$, $[a_g b_g \mathbf{\Gamma}_g d_g]$, $[ab_g\mathbf{\Gamma}_g d_g]$, $[a_{gj}b\mathbf{\Gamma}_g d_g]$, $[a_g b\mathbf{\Gamma}_g d_g]$ and $[ab\mathbf{\Gamma}_g d_g]$, as they are the most flexible among the submodels. The selected submodel, component count, component-wise intrinsic dimensions, BIC of the fitted model, and the ARI are compared.

Method	Submodel	G	d_g	BIC	ARI
ART($\alpha_A = 0.0001$)	$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$	4	(50, 75, 69, 81)	**1883301**	**0.061**
Scree($\alpha_S = 0.2$)	$[ab_g\mathbf{\Gamma}_g d_g]$	6	(1, 3, 3, 1, 2, 1)	-942868	0.006
Scree($\alpha_S = 0.001$)	$[a_{gj}b\mathbf{\Gamma}_g d_g]$	5	(73, 1, 45, 1, 49)	583444	0.044
BIC	$[a_{gj}b_g\mathbf{\Gamma}_g d_g]$	5	(5, 56, 57, 30, 47)	1215835	0.036

Table 5.5: Table of model summaries for ART($\alpha_A = 0.0001$), Scree($\alpha_S = 0.2$), Scree($\alpha_S = 0.001$) and BIC (arranged by row). From the second column on the left, the selected submodel, estimated component count, component-wise intrinsic dimensions, BIC of the fitted model (rounded to the nearest unit) and the Adjusted Rand Index (ARI) (rounded to 3 decimal places) are presented. For the BIC and the ARI, the best values are bolded.

Table 5.5 shows that the ART produced a better fit than the rest in terms of the model BIC and ARI. The rounded average d_g for the four methods are 65, 2, 34 and 39 (in the same order as in table 5.5), indicating an approximately inverse relationship between G and d_g. This is a reasonable behaviour, since fewer components mean each component needs to capture more features of the data set, leading to a higher intrinsic dimension. Submodel-wise, the Scree($\alpha_S = 0.2$) selected the most restrictive one. This is consistent with the relatively large G and low d_g, indicating that each component contains less information about the data set compared to other methods. Thus, a smaller, more restrictive model would suffice. Indeed, a balance between parsimony and expressiveness of the model is desirable, and in some cases, parsimony at the cost of a worse fit may be preferred, wherein Scree($\alpha_S = 0.2$) could be appropriate. However, in general, a better-fitting model is more likely to be preferred over an excessively restrictive one (based on a given model selection criterion). In that sense, the ART appears to strike the best balance among the methods tested, based on the moderate amount of dimension reduction and a superior fit.

5.4 Discussion

In this chapter, we introduced a novel method of intrinsic dimension estimation for the subspace clustering framework on the Gaussian finite mixture model. Our contribution is intended to be a middle ground between the soundness in principle and computational viability. The numerical experiments showed that the ART is a competent (performance and computational cost-wise) alternative to the existing methods. The principle behind the ART could potentially be extended to non-Gaussian finite mixtures or other linear projection-based dimensionality reduction methods. Other directions could include the development of an efficient resampling-based approximation of the asymptotic distribution of the test statistic, in cases where the assumptions for the χ^2 convergence may not hold.

Chapter 6

Flexible Mixture Regression with the Generalized Hyperbolic Distribution

6.1 Introduction

In a regression problem, one may find heterogeneous response-covariate relationships based on latent groups. Several approaches have been proposed to model the cluster-dependent regression, among which are the mixture regression, mixture of experts (MoE) and cluster-weighted model (CWM).

The mixture regression, initially introduced by De Veaux (1989) using a Gaussian finite mixture, modelled the conditional distribution of the response Y given a p-dimensional covariate $\boldsymbol{x}$ (including the intercept term) as a GMM. The mixture regression is perhaps one of the more straightforward ways of introducing clustering to the regression setting, as the component-wise equation is identical to that of the ordinary least squares equation, and the mixing proportion is treated as a model parameter. With the rise in need to look beyond Gaussianity, the mixture regression model has been extended to several non-Gaussian finite mixtures such as the t Yao et al. (2014), skew-normal Liu and Lin (2014), Laplace Song et al. (2014) and the mean-shift normal Yu et al. (2017). In the semi and non-parametric

realm, recent contributions include Hunter and Young (2012); Hu et al. (2017); Ma et al. (2021). For an overview of robust mixture regression models, refer to Yu et al. (2020).

The MoE model, initially introduced by Jacobs et al. (1991), extends the mixture regression by allowing the mixing proportion to be modelled as a function of another covariate $\boldsymbol{r}$ (known as the gating function) in addition to modelling the response Y as a function of $\boldsymbol{x}$ (known as the expert). This approach grants a greater degree of flexibility in cluster-dependent response-covariate modelling by incorporating covariate into the mixing proportion, instead of letting it be estimated as a byproduct of component densities. Yet, it is worth noting that the MoE might need more care from the user, as the covariates need to be partitioned into the $\boldsymbol{r}$ and the $\boldsymbol{x}$ portion. Like the mixture regression model, robust variants of the MoE have been proposed recently, including t (Chamroukhi, 2016), skew-t (Chamroukhi, 2017) and annealing-based MoE (Rao et al., 1997).

The CWM, initially introduced by Gershenfeld (1997), generalizes the mixture regression model in a different way, where distributional assumptions were placed on the covariate $\boldsymbol{x}$. This enables joint modelling of $(Y, \boldsymbol{X})$ by considering the conditional distribution of Y given $\boldsymbol{X}$ and the marginal distribution of $\boldsymbol{X}$. This setup allows various combinations for response and covariate distributions, leading to a quite general model, though the model selection process may be more time-consuming. Recent advances in the CWM include Ingrassia et al. (2012, 2014); Subedi et al. (2013); Punzo and McNicholas (2017); García-Escudero et al. (2017).

The model of interest in this paper is the mixture regression model, due to the relatively straightforward formulation and the widespread use of the fixed-covariate paradigm. The finite mixture of generalized hyperbolic distribution (GHMM) introduced by Browne and McNicholas (2015) is a highly flexible mixture model that includes several robust distributions such as the hyperbolic, normal-inverse Gaussian, variance-gamma and t distributions. It has found its way into numerous modelling frameworks including Tortora et al. (2016); Kim and Browne (2019); Sharp and Browne (2021) thanks to its robustness, and it can improve the performance of a mixture regression model in the presence of a skewed error distribution. Therefore, we introduce a mixture regression model with the GHMM,

and develop a procedure for simplifying the fitted mixture regression by combining similar components into a single component. This procedure is intended for scenarios where a more macroscopic view on the data is desired. Hence, our contribution in this paper is two-fold: a novel mixture regression model and a procedure to aid its interpretability. In the rest of this chapter, we outline the key concepts in the remainder of the introduction. Then, we present the methodology in section 6.2, and simulated experiments and real data illustrations in section 6.3. We conclude with a brief discussion in section 6.4.

6.1.1 Gaussian Mixture Regression

De Veaux (1989) introduced the Gaussian mixture regression model to accommodate heterogeneous response-covariate relationships. Consider a G-component GMM and a set of response-covariate pairs $(y_i, \boldsymbol{x}_i)$. Given the component membership indicators $\boldsymbol{z}_i$ per chapter 2.1, the distribution of y_i is modelled by

$$(Y_i|Z_{ig} = 1) = \boldsymbol{x}_i'\boldsymbol{\gamma}_g + \epsilon_{ig},$$

where $\boldsymbol{\gamma}_g$ are the p-dimensional component-wise regression coefficient vectors, $\epsilon_{ig} \sim N(0, \sigma_g^2)$ represents the component-wise random error following Gaussian distribution with mean 0 and variance σ_g^2, and each ϵ_{ig} is independent of each other Then, the marginal density of Y_i at y_i can be written as that of a GMM

$$f(y_i; \boldsymbol{\Theta}) = \sum_{g=1}^{G} \pi_g \phi(y_i; \boldsymbol{x}_i'\boldsymbol{\gamma}_g, \sigma_g^2).$$

A drawback of the Gaussian component distribution is that it cannot accommodate for heavy tails, skewness and potential outliers. To that end, several robust mixture regression models have been developed, including the mixture regression using the Student-t (Yao et al., 2014), skew-normal (Liu and Lin, 2014) and Laplace (Song et al., 2014) distributions. Other alternatives include penalized mixture regression (Yu et al., 2017), mixture of experts (Rao et al., 1997; Chamroukhi, 2016) and cluster-weighted models (Punzo and McNicholas,

2017; Ingrassia et al., 2014). As the aforementioned works have shown, incorporating highly flexible distributions can further strengthen the robustness of the mixture regression model. To that end, we contribute to the literature by introducing a regression model using the finite mixture of generalized hyperbolic distributions (abbreviated by GHMM in this paper) by Browne and McNicholas (2015).

6.1.2 Generalized Hyperbolic Distribution

An identifiable finite mixture of generalized hyperbolic distributions was introduced by Browne and McNicholas (2015), thus in this paper, we follow their parametrization. We first need to define a generalized inverse Gaussian (GIG) random variable. A random variable $W > 0$ following the GIG distribution with a scale parameter $\eta > 0$, a concentration parameter $\omega > 0$ and an index parameter $\lambda \in \mathbb{R}$ has the density function

$$h(w; \omega, \eta, \lambda) = \frac{(w/\eta)^{\lambda-1}}{2\eta K_\lambda(\omega)} \exp\left\{\frac{-\omega}{2}\left(\frac{w}{\eta} + \frac{\eta}{w}\right)\right\},$$

where $K_\lambda(\omega)$ is the modified Bessel function of second kind with index λ evaluated at $\omega > 0$. We denote this relationship by $W \sim GIG(\omega, \eta, \lambda)$. A random variable $Y \in \mathbb{R}$ following the generalized hyperbolic distribution is defined as a Gaussian variance-mean mixture

$$Y = \mu + W\beta + \sqrt{W}U,$$

where $\mu, \beta \in \mathbb{R}$ are location and skewness parameters, $W \sim GIG(\omega, \eta, \lambda)$ with $\eta = 1$ and $U \sim N(0, \sigma^2)$ such that W and U are independent. This relationship is denoted as $Y \sim GH(\mu, \beta, \sigma^2, \omega, \lambda)$. The density function for this distribution is

$$f_{GH}(y; \mu, \beta, \sigma^2, \omega, \lambda) = \left(\frac{A}{B}\right)^{\frac{\lambda-1/2}{2}} \frac{K_{\lambda-1/2}(\sqrt{AB})}{(2\pi)^{\frac{1}{2}}\sigma K_\lambda(\omega)\exp\left((y-\mu)\beta/\sigma^2\right)}, \tag{6.1}$$

where $A = \omega + (y-\mu)^2/\sigma^2$ and $B = \omega + \beta^2/\sigma^2$. The GH distribution contains a wide range of distributions as special cases such as the hyperbolic, normal-inverse Gaussian,

variance-gamma distribution, Student-t, and Gaussian distributions; see McNeil et al. (2015); Barndorff-Nielsen (1978); Kotz et al. (2012) for details.

6.1.3 Simplifying the Model for Interpretability

As Hennig (2010) explained, not every component in a finite mixture may correspond to a cluster (whose definition is context-dependent), and the number of components may be over-estimated. Too many components could mean that the partitioning of observations may be too granular, thus interpreting the model can be complicated. When seeking a simpler explanation from a model, two potential remedies are component merging and combining. Given a finite mixture model, component merging unifies the label of selected components without refitting the model. Besides avoiding a refit, depending on the method, component merging can help with detecting multi-modal clusters that no single component can produce. Thus, component merging also helps finite mixtures with rigid component distributions bootstrap their way to more flexible shapes. For an overview and recent developments in this area, refer to Hennig (2010); Baudry et al. (2010); Melnykov (2016); Chacón (2019); Menardi (2016); Kim and Browne (2021a). Combining components is another flavour of the remedy, and it is the method of interest in this paper. Scott and Szewczyk (2001) introduced the Iterative Pairwise Replacement Algorithm (IPRA) that combines iteratively pairs of similar mixture components in a finite Gaussian mixture. The IPRA combined into a single component only the pair of interest through local refitting, thereby avoiding the computation on all components. This approach is useful when the one-to-one correspondence between a component and a cluster is reasonable, or when the user seeks a simpler interpretation of model parameters. For example, figure 6.1 plots two instances of clustering with GMM on the Old Faithful data (R Core Team, 2020), where the left-side fits three components and the right-side fits two components. An argument could be made for combining the black and green-coloured components in the left-side plot, given their proximity and the small difference in BIC (-822.693 versus -828.569 for three and two components respectively), as well as the mild change in density contours before and after combining. After combining, we arrive at the two-component mixture on

the right side.

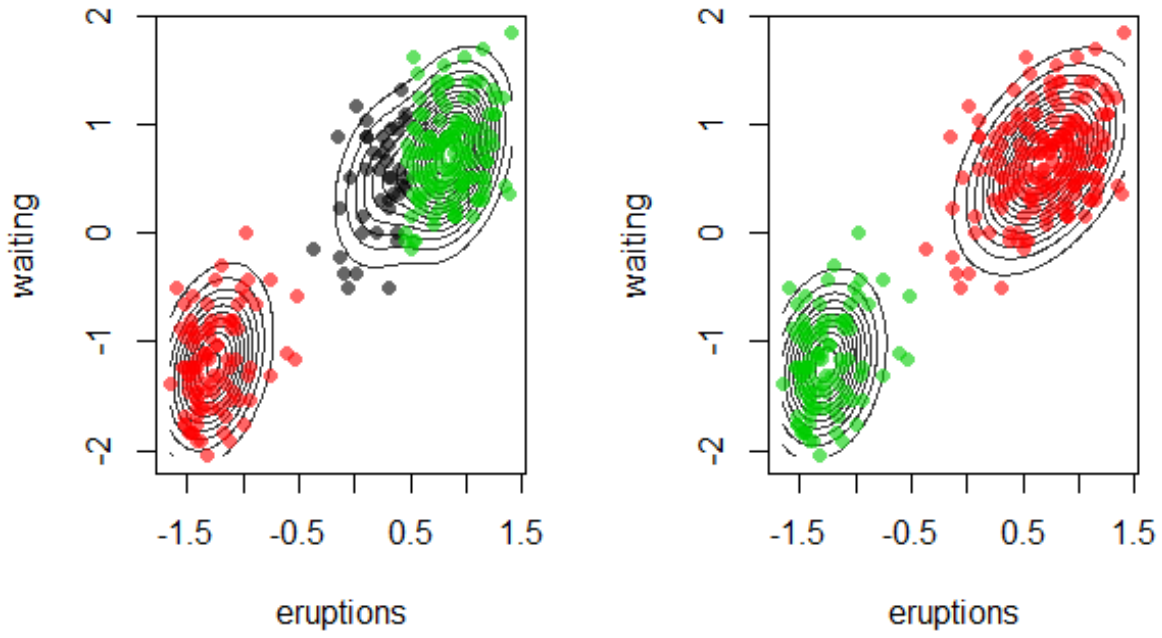

Figure 6.1: Scatterplot, with density contours, of two instances of GMM fitted to the scaled Old Faithful data. Left and right-side plots fit three and two components each.

In the mixture regression context, combining components can adjust the granularity of the response-covariate relationship. As an illustration, consider figure 6.2, which shows two instances of Gaussian linear regression fitted to the Cars data (R Core Team, 2020). The left-side partitions the data into two components via GMM first, and a regression line is fitted onto each component. On the right side, a single regression line is fitted to the whole data. If one is interested only in the general trend in the data, an argument can be made for a single-component regression. Moreover, merging components without refitting, instead of combining, can lead to ambiguity in interpretation. If the two Gaussian components

in the Cars data were merged without refitting, there would be two response-covariate relationships representing a single cluster. Combining them into a single component, thus regression line, can remove this ambiguity.

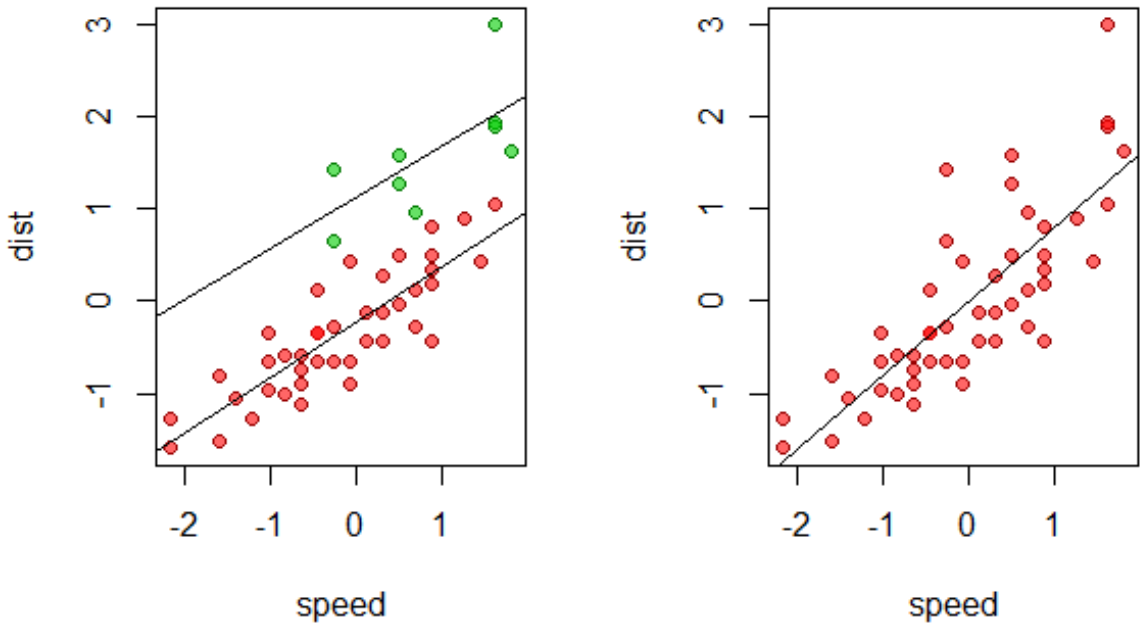

Figure 6.2: Scatterplot, with component-wise regression lines, of two instances of GMM fitted to the scaled Cars data. Left and right-side plots fit two and one components each.

6.2 Methodology

In this section, we introduce the generalized hyperbolic mixture regression (GHMR) model. We then present the parameter estimates, including the component-wise regression coefficients. Further, we present a component-combining procedure.

6.2.1 Generalized Hyperbolic Mixture Regression Model

Consider firstly the single component case. Given a set of n many response-covariate pairs (including the intercept) $(Y_i, \boldsymbol{x}_i)$ where Y_i is random and $\boldsymbol{x}_i$ is deterministic and p-dimensional, we assume the following relationship. We capitalize Y_i to contrast it with its observed value analog y_i.

$$Y_i = \boldsymbol{x}_i'\boldsymbol{\gamma} + \epsilon_i, \tag{6.2}$$

where $\boldsymbol{\gamma}$ is the regression coefficient vector and $\epsilon_i =\sim GH(0, \beta, \sigma^2, \omega, \lambda)$ for $i = 1, 2, \ldots n$ are the mutually independent generalized hyperbolic noise random variables. This means that we can rewrite Y_i as $Y_i = \boldsymbol{x}_i'\boldsymbol{\gamma} + W_i\beta + \sqrt{W_i}U_i$ where $W_i \sim GIG(\omega, 1, \lambda)$ and $U_i \sim N(0, \sigma^2)$ such that they form a GH random variable. This means that $U_i|w_i$ follows Gaussian distribution with mean $\boldsymbol{x}_i'\boldsymbol{\gamma}+w_i\beta$ and variance $w_i\sigma^2$. Furthermore, Browne and McNicholas (2015) showed that the conditional distribution of $W_i|y_i$ is $GIG(\omega + \beta^2/\sigma^2, \omega + (y_i - \boldsymbol{x}_i'\boldsymbol{\gamma})^2/\sigma^2, \lambda - 1/2)$. We can extend this model to a G-component finite mixture. Conditional on $Z_{ig} = 1$, the functional form of Y_i is

$$(Y_i|Z_{ig} = 1) = \boldsymbol{x}_i'\boldsymbol{\gamma}_g + \epsilon_{ig},$$

where $\boldsymbol{\gamma}_g$ is the regression coefficient vector for component g and $\epsilon_{ig} \sim GH(0, \beta_g, \sigma_g^2, \omega_g, \lambda_g)$ where the subscript g is understood as the component label. Then, the marginal density of y_i can be written as that of the univariate generalized hyperbolic mixture model (GHMM)

$$f(y_i; \boldsymbol{\Theta}) = \sum_{g=1}^{G} \pi_g f_{GH}(y_i; \boldsymbol{x}_i'\boldsymbol{\gamma}_g, \beta_g, \sigma_g^2, \omega_g, \lambda_g),$$

where $\boldsymbol{\Theta} = \{\{\boldsymbol{\gamma}_g, \beta_g, \sigma_g^2, \omega_g, \lambda_g, \pi_g\}\}_{g=1,\ldots,G}$ denotes the set of all model parameters. There are $Gp + 4G + (G - 1)$ free parameters in total. By treating Z_{ig} and W_{ig} as latent variables,

we can obtain the complete-data log-likelihood function $l_c(\boldsymbol{\Theta})$

$$
\begin{aligned}
l_c(\boldsymbol{\Theta}) &= \sum_{i=1}^{n}\sum_{g=1}^{G} Z_{ig}\left[\log\pi_g + \log\phi(y_i;\boldsymbol{x}_i'\boldsymbol{\gamma}_g + W_{ig}\beta_g, W_{ig}\sigma_g^2) + \log h(W_{ig};\omega_g,1,\lambda_g)\right] \\
&= \sum_{i=1}^{n}\sum_{g=1}^{G} Z_{ig}\Bigg\{\log\pi_g - \frac{1}{2}\left[\log W_{ig} + \log\sigma_g^2 + \frac{1}{W_{ig}\sigma_g^2}(y_i - \boldsymbol{x}_i'\boldsymbol{\gamma}_g - W_{ig}\beta_g)^2\right] \\
&\quad + \left[(\lambda_g - 1)\log W_{ig} - 2\log K_{\lambda_g}(\omega_g) - \frac{\omega_g}{2}(W_{ig} + 1/W_{ig})\right]\Bigg\} + \text{const}, \qquad (6.3)
\end{aligned}
$$

where 'const' is a collection of constants with respect to the parameters.

6.2.2 Parameter Estimation

We follow the EM algorithm-based parameter estimation by Browne and McNicholas (2015). The following expected values are needed for the E-step. Here, the parameter estimates at iteration t are denoted by a super-script (t). For example, $\boldsymbol{\Theta}^{(t)}$, $\boldsymbol{\Theta}_g^{(t)}$ and $\boldsymbol{\gamma}_g^{(t)}$ denote the set of estimates for all parameters, the set of estimates for parameters in component g and the estimate for $\boldsymbol{\gamma}_g$, at iteration t.

$$
\begin{aligned}
z_{ig}^{(t)} &:= \mathrm{E}\left[Z_{ig}|y_i,\boldsymbol{x}_i,\boldsymbol{\Theta}^{(t)}\right] = \pi_g^{(t)} f_{GH}\left(y_i;\boldsymbol{\Theta}_g^{(t)}\right) / \sum_{j=1}^{G}\pi_j^{(t)} f_{GH}\left(y_i;\boldsymbol{\Theta}_j^{(t)}\right) \\
a_{ig}^{(t)} &:= \mathrm{E}\left[W_{ig}|Z_{ig}=1,y_i,\boldsymbol{x}_i,\boldsymbol{\Theta}^{(t)}\right] = \sqrt{\frac{A_{ig}^{(t)}}{B_g^{(t)}}} \times R_{v_g^{(t)}}\left(\sqrt{A_{ig}^{(t)}B_g^{(t)}}\right) \\
b_{ig}^{(t)} &:= \mathrm{E}\left[1/W_{ig}|Z_{ig}=1,y_i,\boldsymbol{x}_i,\boldsymbol{\Theta}^{(t)}\right] = \frac{-2v_g^{(t)}}{A_{ig}^{(t)}} + \sqrt{\frac{B_g^{(t)}}{A_{ig}^{(t)}}} \times R_{v_g^{(t)}}\left(\sqrt{A_{ig}^{(t)}B_g^{(t)}}\right) \\
c_{ig}^{(t)} &:= \mathrm{E}\left[\log W_{ig}|Z_{ig}=1,y_i,\boldsymbol{x}_i,\boldsymbol{\Theta}^{(t)}\right] = \log\sqrt{\frac{A_{ig}^{(t)}}{B_g^{(t)}}} + \frac{\partial}{\partial s}\log\left\{K_s\left(\sqrt{A_{ig}^{(t)}B_g^{(t)}}\right)\right\}\Big|_{s=v_g^{(t)}},
\end{aligned}
$$

where $R_s(\cdot) = K_{s+1}(\cdot)/K_s(\cdot)$, $A_{ig}^{(t)} = \omega_g^{(t)} + \left(y_i - \boldsymbol{x}_i'\boldsymbol{\gamma}_g^{(t)}\right)^2/\sigma_g^{(t)^2}$, $B_g^{(t)} = \omega_g^{(t)} + \beta_g^{(t)^2}/\sigma_g^{(t)^2}$, $v_g^{(t)} = \lambda_g^{(t)} - \frac{1}{2}$ and

$$n_g := \sum_{i=1}^{n} z_{ig}^{(t)}, \quad \bar{a}_g := \frac{1}{n_g}\sum_{i=1}^{n} z_{ig}^{(t)} a_{ig}^{(t)}, \quad \bar{b}_g := \frac{1}{n_g}\sum_{i=1}^{n} z_{ig}^{(t)} b_{ig}^{(t)}, \quad \bar{c}_g := \frac{1}{n_g}\sum_{i=1}^{n} z_{ig}^{(t)} c_{ig}^{(t)}.$$

For the M-step, we maximize the expected complete-data log-likelihood function $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) = \mathrm{E}[l_c(\Theta)|\boldsymbol{\Theta}^{(t)}]$ with respect to $\boldsymbol{\Theta}$. The following parameter estimates are obtained.

$$\pi_g^{(t+1)} = \frac{n_g}{n}, \quad \beta_g^{(t+1)} = \frac{\bar{b}_g \sum_{i=1}^{n} z_{ig}^{(t)}(y_i - \boldsymbol{x}_i'\boldsymbol{\gamma}_g^{(t)})}{\sum_{i=1}^{n} z_{ig}^{(t)} a_{ig}^{(t)}}$$

$$\sigma_g^{(t+1)^2} = \frac{1}{n_g}\sum_{i=1}^{n} z_{ig}^{(t)}\left(b_{ig}^{(t)}(y_i - \boldsymbol{x}_i'\boldsymbol{\gamma}_g^{(t)})^2 - 2\beta_g^{(t)}(y_i - \boldsymbol{x}_i'\boldsymbol{\gamma}_g^{(t)})\right) + \bar{a}_g\beta_g^{(t)^2}.$$

Now let $q_g(\omega_g, \lambda_g) := -\log K_{\lambda_g}(\omega_g) + (\lambda_g - 1)\bar{c}_g - \frac{\omega_g}{2}(\bar{a}_g + \bar{b}_g)$.

$$\lambda_g^{(t+1)} = \bar{c}_g\lambda_g^{(t)}\left\{\frac{\partial}{\partial s}\ \log K_s(\omega_g^{(t)})\ \Big|_{s=\lambda_g^{(t)}}\right\}^{-1},$$

$$\omega_g^{(t+1)} = \omega_g^{(t)} - \left\{\frac{\partial}{\partial s} q_g(s, \lambda_g^{(t+1)})\ \Big|_{s=\omega_g^{(t)}}\right\}\left\{\frac{\partial^2}{\partial s^2} q_g(s, \lambda_g^{(t+1)})\ \Big|_{s=\omega_g^{(t)}}\right\}^{-1}.$$

In addition to the updates given by Browne and McNicholas (2015), we derive newly the update for $\boldsymbol{\gamma}_g$. Notice that $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)})$ is a quadratic function of $\boldsymbol{\gamma}_g$. Thus, after differentiating with respect to $\boldsymbol{\gamma}_g$, we solve the following equation,

$$\sum_{i=1}^{n} z_{ig}^{(t)}(b_{ig}^{(t)} y_i + \beta_g^{(t)})\boldsymbol{x}_i = \sum_{i=1}^{n} z_{ig}^{(t)} b_{ig}^{(t)} \boldsymbol{x}_i'\boldsymbol{\gamma}_g \boldsymbol{x}_i.$$

By denoting $\boldsymbol{X} = [\boldsymbol{x}_1 \dots \boldsymbol{x}_n]'$, $\boldsymbol{M}_g^{(t)} = \mathrm{diag}(z_{ig}^{(t)} b_{ig}^{(t)})_{i=1,\dots,n}$, $\boldsymbol{y} = (y_1, \dots, y_n)'$ and $\boldsymbol{z}_g^{(t)} = (z_{1g}^{(t)}, \dots, z_{ng}^{(t)})'$, the above equation can be simplified to

$$(\boldsymbol{X}'\boldsymbol{M}_g^{(t)}\boldsymbol{X})\boldsymbol{\gamma}_g = \boldsymbol{X}'(\boldsymbol{M}_g^{(t)}\boldsymbol{y} + \beta_g^{(t)}\boldsymbol{z}_g^{(t)}).$$

Then the solution below follows immediately,

$$\boldsymbol{\gamma}_g^{(t+1)} = (\boldsymbol{X}'\boldsymbol{M}_g^{(t)}\boldsymbol{X})^{-1}\boldsymbol{X}'(\boldsymbol{M}_g^{(t)}\boldsymbol{y} + \beta_g^{(t)}\boldsymbol{z}_g^{(t)}). \tag{6.4}$$

The matrix $\boldsymbol{X}'\boldsymbol{M}_g^{(t)}\boldsymbol{X}$ is invertible as long as $\boldsymbol{X}$ is full rank and the number of non-zero entries in $\boldsymbol{M}_g^{(t)}$ is at least the rank of $\boldsymbol{X}$. Otherwise, a pseudo-inverse $(\boldsymbol{X}'\boldsymbol{M}_g^{(t)}\boldsymbol{X})^+$ could be used instead.

Finally, once the estimated model parameter set $\hat{\boldsymbol{\Theta}}$ is obtained, the observations are assigned to components via Maximum A Posteriori (MAP) approximation of Z_{ig}, denoted by $\hat{z}_{ig}$.

Identifiability

The identifiability of model parameters is essential for their consistent estimation. Hennig (2000) showed that the identifiability of a finite mixture does not guarantee that of its regression variant, and derived a sufficient condition for identifiability of the Gaussian finite mixture regression model with fixed p-dimensional covariates. Let $\mathcal{H}$ be the minimum number of $(p-1)$-dimensional hyperplanes needed to cover all covariates (excluding the value 1 reserved for the intercept) in the data set. For example, we need at least one line to cover two points in $\mathcal{R}^2$, so $\mathcal{H} = 1$ in this case. To cover four vertices of a rectangle, we need at least two lines, so $\mathcal{H}$ would be 2. The proposed sufficient condition is $G < \mathcal{H}$. That means, as long as the number of components is suitably bounded, the Gaussian mixture regression model with fixed covariates would be identifiable. Furthermore, its proof shows that the mixture regression model is identifiable if the underlying mixture model is identifiable. Therefore, since Browne and McNicholas (2015) showed the identifiability of the GHMM, the GHMR is also identifiable.

Proposition 1. *If $G < \mathcal{H}$, then the GHMR is identifiable.*

Although the theoretical identifiability is established, its application can be challenging, because two covariates in a data set are unlikely to be an exact linear combination of

each other. This implies that a naïve application of the identifiability condition would often yield $\mathcal{H} = p$ in practice. Moreover, Hennig (2000) noted the steep computational complexity of NP-complete in computing $\mathcal{H}$ exactly. Hence, we propose instead to count the number of important principal axes of $\boldsymbol{X}^\star$, which denotes $\boldsymbol{X}$ without the column of 1s reserved for the intercept. Each principal axis represents a 1-dimensional subspace in $\mathcal{R}^p$, thus for each axis, there exist at least one $(p-1)$-dimensional hyperplane that contain the subspace represented by that axis. Therefore, though crude, counting the number of important principal axes can serve as a computationally feasible approximation of $\mathcal{H}$. This procedure can guide the investigator in determining the maximum number of components for consideration during model-fitting. The important principal axes are counted via the eigenvalues of the sample covariance matrix of $\boldsymbol{X}^\star$, similar to the scree test by Cattell (1966). Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$ be the said eigenvalues. We compute the first-order sequential differences $d_i = \lambda_i - \lambda_{i+1}$ for $i = 1, 2, \ldots, p-1$. Then, given a pre-determined threshold $c > 0$, the smallest i such that $d_i < c$ is chosen as an estimate of $\mathcal{H}$. In this paper, we set $c = 0.01$.

Combining GHMR components

Given a model with G components, suppose one wishes to simplify some components. As discussed in section 6.1.3, combining selected components into a single component is the method of interest. We propose a two-step procedure, where the first step identifies pairs with potential to merge, and the second step selects a single pair among those from first step based on the Integrated Completed Likelihood (ICL) by Biernacki et al. (2000), which is outlined in chapter 2.1. We describe the combining procedure below.

1. Measuring the difference between two regression coefficient vectors $\boldsymbol{\gamma}$ and $\boldsymbol{\gamma}^\star$: Let $d(\boldsymbol{\gamma}, \boldsymbol{\gamma}^\star)$ be defined as

$$d(\boldsymbol{\gamma}, \boldsymbol{\gamma}^\star) = a\frac{|\boldsymbol{\gamma}'\boldsymbol{\gamma}^\star|}{||\boldsymbol{\gamma}||_2||\boldsymbol{\gamma}^\star||_2} + (1-a)||\boldsymbol{\gamma} - \boldsymbol{\gamma}^\star||_2.$$

   $d(\boldsymbol{\gamma}, \boldsymbol{\gamma}^\star)$ is a non-negative measure of the difference between two vectors as a weighted

sum of their absolute cosine and the 2-norm distance, where we set $a = 1/3$ to prioritize the distance between the coefficient vectors. We compute the d value for every pair of coefficient vectors in the mixture model, resulting in $\binom{G}{2}$ values. Let $d_{\min}$ denote the smallest value among the $\binom{G}{2}$. In case of a tie, we prioritize the pair whose sum of component indices is lower. If there is still a tie, we choose the pair whose minimum of the component indices is lower. For example, if the d value of pairs $(2, 3)$ and $(1, 4)$ are equally minimal, then $d_{\min}$ equals to that of pair $(1, 4)$ since it has a lower minimum of component index. If there are only two components in the mixture, the BIC value of the two-component and one-component models are compared, and the component count with higher BIC is chosen.

2. Refit the pair of components corresponding to $d_{\min}$ as a single component. If the merged model improves in BIC, then the merged model is kept, and the process is repeated from step 1 using the merged model. Otherwise, the merging process is halted, and the model before the most recent merging is kept as the final one.

While the above procedure determines automatically the number of components post-merging (denoted by K), K could be pre-determined as well. In that case, the procedure would be repeated until K components remain, irrespective of the change in the BIC value. The refitting of the selected components is done via the EM algorithm shown in section 6.2.2, where the observations from selected components are grouped into a single component. Without loss of generality, let components 1 and 2 be merged. For notational brevity, component index will be omitted for the parameters and conditional expectations belonging to the merged component, but the symbols themselves will not change. For example, the scale and index parameters of the merged components will be denoted by σ^2 and λ respectively. Moreover, the conditional expectations $a_{ig}^{(t)}$, $b_{ig}^{(t)}$ and $c_{ig}^{(t)}$ computed with merged parameters will be written as $a_i^{(t)}$, $b_i^{(t)}$ and $c_i^{(t)}$ respectively.

First, the membership probabilities at the time of convergence are combined: $z_i = z_{i1}^{(t)} + z_{i2}^{(t)}$. Then we can compute $\bar{n} = \sum_{i=1}^{n} z_i$ and $\pi = \bar{n}/n$. With the scaled weight of component 1, $u = \sum_{i=1}^{n} z_{i1}^{(t)}/\bar{n}$, the model parameters are initialized as the weighted

average of parameters from each component

$$\boldsymbol{\gamma}^{(t)} = u\boldsymbol{\gamma}_1^{(t)} + (1-u)\boldsymbol{\gamma}_2^{(t)}, \quad \beta^{(t)} = u\beta_1^{(t)} + (1-u)\beta_2^{(t)},$$
$$\sigma^{(t)^2} = u\sigma_1^{(t)^2} + (1-u)\sigma_2^{(t)^2}, \quad \omega^{(t)} = u\omega_1^{(t)} + (1-u)\omega_2^{(t)},$$
$$\lambda^{(t)} = u\lambda_1^{(t)} + (1-u)\lambda_2^{(t)}.$$

Thereafter, the conditional expectations $a_i^{(t)}$, $b_i^{(t)}$ and $c_i^{(t)}$, and the updated model parameters $\boldsymbol{\gamma}^{(t+1)}$, $\beta^{(t+1)}$, $\sigma^{(t+1)^2}$, $\omega^{(t+1)}$ and $\lambda^{(t+1)}$ are computed using the formulae analogous to those in section 6.2.2. The update is iterated thereafter until convergence; the convergence criterion is outlined in section 6.3.1.

6.3 Numerical Experiments

In this section, we illustrate the GHMR using simulated and real data sets. Below is the brief description of each setting.

- Simulated data 1: A p-dimensional ($p = 10, 20$), 2-component mixture regression data with generalized hyperbolic error is considered with component-wise sample size $n_g = 100, 200$ for $p = 10$ and $n_g = 200, 400$ for $p = 20$. The goal is to compare the performance of various mixture regression models.

- Simulated data 2: A 2-dimensional, 1-component mixture regression data with generalized hyperbolic error is considered with sample size $n = 100, 200, 300, 400$. We illustrate the gain in performance from combining GHMR components.

- Real data 1: The Fish Market data from Pyae (2019) is used. We focus on the relationship between the Height and Width variables of Bream and Roach species.

- Real data 2: The Italian Tourism data from ISTAT is used. We focus on the informativeness of components estimated by the GHMR.

The list of compared models are given below. For a pre-determined range of component count G, each model is fitted until convergence, and the one yielding the best model selection criterion value is chosen. In this manuscript, the BIC is used as the model selection criterion. For example, if the range of G is $1, 2, \ldots, 6$, then for each $G = 1, 2, \ldots, 6$, a GHMR model is fitted until convergence, and the one with the best BIC value is chosen (likewise for the other models). All other model-specific hyperparameters from GMR, RGMR and TLE are chosen as the default values set in their respective software packages.

- Generalized Hyperbolic Mixture Regression (GHMR).
- Gaussian Mixture Regression (GMR) implemented in the R package *flexmix* (Grün and Leisch, 2008). It is a mixture regression model where the error follows a finite mixture of Gaussian distributions.
- Robust Gaussian Mixture Regression (RGMR) from the R package *mixtools* (Benaglia et al., 2009). It is based on the GMR model but allocates an extra component to collect the observations classified as noise.
- Trimmed Likelihood Estimation for GMR (TLE) from R package *RobMixReg* (Cao et al., 2020). It deploys a trimmed likelihood estimation method from Neykov et al. (2007).

6.3.1 Computational Aspects

The mixture model parameters were initialized via a preliminary component assignment of each observation, followed by the calculation of component-wise model parameters. For the GMR, RGMR and TLE, the corresponding R packages' default component assignment method was used. Specifically, GMR (from *flexmix*) and RGMR (from *mixtools*) and TLE (from *RobMixReg*) used uniform random assignment, and the GHMR used k-means assignment. With regards to algorithm convergence, Aitken's Acceleration was used on the GHMR to determine the convergence of the EM algorithm (outlined in chapter

2.1). In this work, the stopping threshold is set to 0.01. For the other methods, the default convergence rule for the corresponding packages were used. Specifically, the GMR checks whether $l(\boldsymbol{\Theta}^{(t+1)}) - (\boldsymbol{\Theta}^{(t)}) < c$ for $c = 10^{-8}$. The RGMR and TLE check whether $|l(\boldsymbol{\Theta}^{(t+1)}) - (\boldsymbol{\Theta}^{(t)})|/|l(\boldsymbol{\Theta}^{(t+1)})| < c$, where $c = 0.01$ for RGMR and $c = 10^{-8}$ for TLE. Model performance was measured by the BIC, ARI and the distance between the true and estimated component-wise regression coefficient vectors. To circumvent the issue of unbounded likelihood arising from a degenerate model, the scale parameter σ_g^2 is constrained to be larger than a pre-set threshold so that the variance does not vanish. The threshold is set as 10^{-8} in this paper. If a fitted model violates this constraint, it is discarded and the model is re-fitted with a new initialization.

To assess the closeness between the estimated and true regression coefficient vectors, the 2-norm between each (estimated coefficient, true coefficient) pair is computed. For example, suppose the estimated model has 3 components but the true model has 2. Then there are 6 (estimated, true) vector pairs: $\{(1,1),(1,2),(2,1),(2,2),(3,1),(3,2)\}$. The 2-norm between the coefficient vectors for each pair is computed, and the pairwise norms are summed. To formalize, given a model with G-components and another with $\hat{G}$ components, we define the discrepancy measure 'Dist' between the coefficient vector sets from the two models as

$$\text{Dist}(\{\boldsymbol{\gamma}_1,\ldots,\boldsymbol{\gamma}_G\},\{\hat{\boldsymbol{\gamma}}_1,\ldots,\hat{\boldsymbol{\gamma}}_{\hat{G}}\}) = \sum_{\text{unique } (i,j) \text{ pairs}} ||\boldsymbol{\gamma}_i - \hat{\boldsymbol{\gamma}}_j||_2.$$

Given a true model and a set of candidate models, Dist chooses the candidate that minimizes its value when compared against the true model. In general, if $G_0 = \min\{G,\hat{G}\}$ and $G_1 = \max\{G,\hat{G}\}$, the number of summands in our discrepancy measure is $G_0G_1 - G_0(G_0-1)/2$. We can expect Dist to favour parsimony over verbosity, per the following illustration. Suppose that the true model has 4 components, and that two estimated models are available: candidate 1 has 2 components, and candidate 2 has 6 components. Further suppose that the 2-norm between every (true, estimated) coefficient vector pair is 1, and that each pairwise 2-norm within the true model is also 1. In other words, let $\{\boldsymbol{\gamma}_1,\ldots,\boldsymbol{\gamma}_4\}$, $\{\hat{\boldsymbol{\gamma}}_1,\hat{\boldsymbol{\gamma}}_2\}$ and $\{\tilde{\boldsymbol{\gamma}}_1,\ldots,\tilde{\boldsymbol{\gamma}}_6\}$ denote the coefficient vector set of the true model, candidate 1

and candidate 2 respectively. Then, we assume that

$$\begin{aligned}
&||\boldsymbol{\gamma}_i - \boldsymbol{\gamma}_j||_2 = 1 \quad \text{for } i \neq j, \\
&||\boldsymbol{\gamma}_i - \hat{\boldsymbol{\gamma}}_j||_2 = 1 \quad \text{for } i = 1, \ldots, 4, \quad j = 1, 2, \text{ and} \\
&||\boldsymbol{\gamma}_i - \tilde{\boldsymbol{\gamma}}_j||_2 = 1 \quad \text{for } i = 1, \ldots, 4, \quad j = 1, \ldots, 6.
\end{aligned}$$

Then, based on the summand count formula, we have

$$\begin{aligned}
&\text{Dist}(\{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_4\}, \{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_4\}) = 4 \times 4 - 4(4-1)/2 = 10, \\
&\text{Dist}(\{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_4\}, \{\hat{\boldsymbol{\gamma}}_1, \hat{\boldsymbol{\gamma}}_2\}) = 2 \times 4 - 2(2-1)/2 = 7, \text{ and} \\
&\text{Dist}(\{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_4\}, \{\tilde{\boldsymbol{\gamma}}_1, \ldots, \tilde{\boldsymbol{\gamma}}_6\}) = 4 \times 6 - 4(4-1)/2 = 18.
\end{aligned}$$

Hence, in terms of Dist, model candidate 1 would be preferred, if both candidates produce coefficient vectors that are equally distant from that of the true model.

6.3.2 Simulated Data 1

A p-dimensional, 2-component, mixture regression data with generalized hyperbolic errors is generated. The data-generating model is specified as follows. Let $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ follow independently the p-dimensional Gaussian distribution with mean equal to $(1, 1, \ldots, 1)'$ and covariance equal to the identity matrix, and denote the component-wise regression coefficients by

$$\boldsymbol{\gamma}_1 = (\underbrace{-3 \cdots, -3}_{p \text{ copies}})', \quad \boldsymbol{\gamma}_2 = (\underbrace{3, \cdots, 3}_{p \text{ copies}})'.$$

The responses Y_i are generated from

$$Y_i = \begin{cases} -2 + \boldsymbol{\gamma}_1' \boldsymbol{x}_{i1} + GH((\mu_1, \sigma_1^2, \beta_1, \omega_1, \lambda_1) = (0, 1.5, 3, 0.5, 0.7)) & \pi_1 = 0.5, \\ 2 + \boldsymbol{\gamma}_2' \boldsymbol{x}_{i2} + GH((\mu_2, \sigma_2^2, \beta_2, \omega_2, \lambda_2) = (0, 1, -1, 2, 1)) & \pi_2 = 0.5, \end{cases} \tag{6.5}$$

and the generated response variable was scaled by its sample standard deviation. The

(dimension, component-wise sample size) = (p, n_g) pairs are selected so that the ratio n_g/p is 10 or 20. Specifically, the following values are considered: $(p, n_g) \in \{(10, 100), (10, 200), (20, 200), (20, 400)\}$. The GHMR, GMR, RGMR and TLE models are fitted to the generated data set for performance comparison, and this is replicated 500 times, with a newly-generated data set each time. For GHMR, RGMR and TLE models, the considered component counts are $G = 1, 2, 3, 4, 5, 6$, and that for GMR are $G = 2, 3, 4, 5, 6$, since the GMR software does not support $G = 1$.

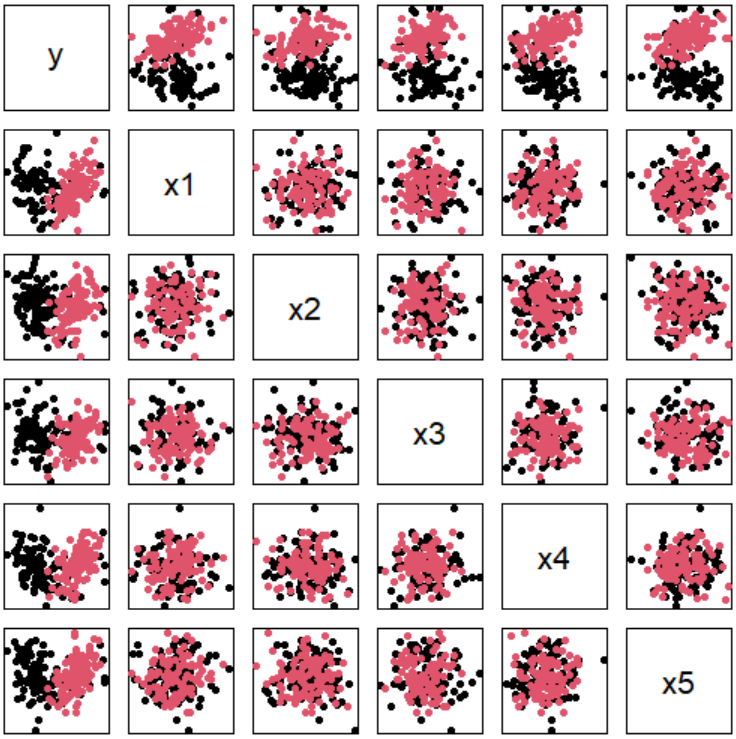

Figure 6.3: A pair plot of a 5-dimensional instance of data set simulated from 6.5, where the observations are coloured by component.

(10, 100)	BIC	G	ARI	Dist
GHMR	**-67**	2	0.781	**38**
	(1630)	(1)	(0.500)	(19)
GMR	-92	5	0.542	95
	(72)	(2)	(0.164)	(38)
RGMR	-148	3	0.810	57
	(54)	(1)	(0.146)	19
TLE	-244	2	**0.835**	38
	(86)	(2)	(0.215)	(38)

(10, 200)	BIC	G	ARI	Dist
GHMR	**3585**	3	0.815	57
	(4601)	(2)	(0.224)	(38)
GMR	-180	5	0.652	95
	(78)	(2)	(0.134)	(38)
RGMR	-212	3	0.831	57
	(62)	(0)	(0.098)	(0)
TLE	-430	2	**0.876**	**38**
	(116)	(0)	(0.039)	(0)

(20, 200)				
GHMR	-293	2	0.680	54
	(1375)	(2)	(0.963)	(54)
GMR	**334**	5	0.587	134
	(138)	(1)	(0.173)	(54)
RGMR	193	3	0.865	81
	(81)	(1)	(0.169)	(27)
TLE	10	2	**0.903**	**54**
	(163)	(1)	(0.110)	(27)

(20, 400)				
GHMR	32	2	0.669	54
	(7164)	(3)	(0.856)	(80)
GMR	**708**	5	0.666	134
	(134)	(2)	(0.130)	(54)
RGMR	585	3	0.810	80
	(103)	(0)	(0.079)	(0)
TLE	139	2	**0.905**	**54**
	(194)	(0)	(0.004)	(0)

Table 6.1: Table of median BIC, component count G, Dist (rounded to nearest digit) and ARI (rounded to three decimal places) over 500 replications of model-fitting on the data sets generated from (6.5). The (p, n_g) pair for each table is specified in the top-left corner. Inter-quartile ranges (IQR) are written in brackets underneath each median value, and the best median BIC, ARI and Dist are bolded.

Table 6.1 indicates that the GHMR achieved the highest median BIC when $(p, n_g) = (10, 100), (10, 200)$. In contrast, the TLE obtained the highest median ARI in all settings. In addition, the TLE obtained the median G of 2 (the true number of components) in all settings, followed by GHMR (3 out of 4 settings). Both GMR and RGMR tended to over-estimate the component count, but RGMR was closer to 2, which is likely attributed to its

BIC	$(10, 100)$	$(10, 200)$	$(20, 200)$	$(20, 400)$
GHMR	**252**	**408**	74	193
GMR	222	71	**406**	**303**
RGMR	24	21	15	4
TLE	2	0	5	0

Table 6.2: Table recording the number of replications where each model achieved the highest performance measurement based on 500 replications. The in-class best values are bolded.

robust formulation in comparison to the GMR. Interestingly, when compared replication-wise, the GHMR obtained the highest BIC most often when $(p, n_g) = (10, 100), (10, 200)$, whereas the GMR performed well when $(p, n_g) = (20, 200), (20, 400)$, as shown in table 6.2. The tabulated results suggest that the GHMR is capable of producing a significantly better fit compared to the other models.

6.3.3 Simulated Data 2

A 2-dimensional, 1-component mixture regression data with generalized hyperbolic errors is generated. The covariates x_i $(i = 1, \ldots, n)$ are generated independently from a univariate Gaussian distribution with mean and variance equal to 1, and the slope parameter is $\gamma = 1$. The error distribution and response equation are

$$\epsilon_i \sim GH((\mu, \sigma^2, \beta, \omega, \lambda) = (0, 2, 0, 2, 0.05)),$$
$$Y_i = 0 + \gamma x_i + \epsilon_i.$$

This experiment is intended to showcase the performance improvement achieved from GHMR component combining under over-estimated component counts. For each generated data set, a G-component GHMR model is fitted over $G = 2, 3, 4, 5, 6$. Then, the fitted model's components are combined via the procedure outlined in section 6.2.2. The considered sample sizes are $n = 100, 200, 300, 400$. For n value, the experiment is replicated 500 times. The results are summarized in table 6.3.

Table 6.3 shows that combining components resulted in a significant improvement in

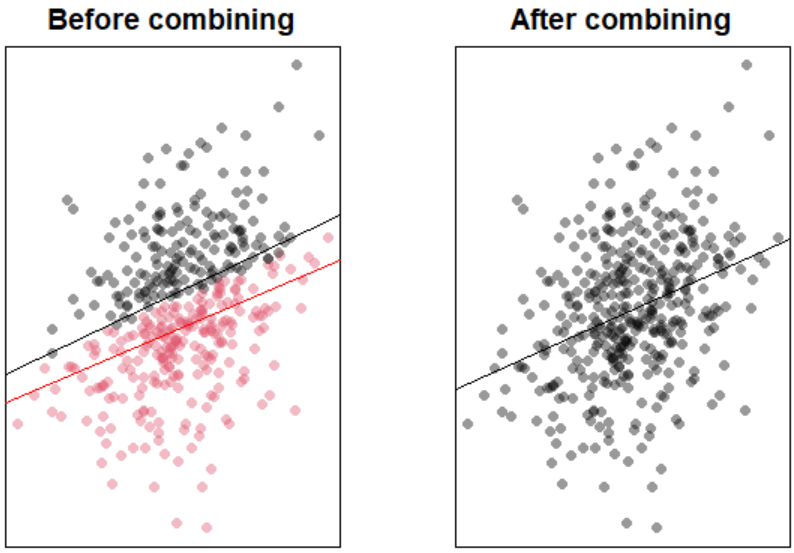

Figure 6.4: An instance of the experiment before and after component combining. On the left, a scatterplot of observations coloured by components and overlaid with regression lines is shown. On the right, a combined GHMR model with overlaid regression line is shown.

median ICL values across all considered sample sizes. In addition, component-combining resulted in 1 component (true component count) in at least 50% of replications, except for $n = 400$ whence near 50% rate is observed. Furthermore, the scatterplots in figure 6.5 show that component-combining resulted in reduced component counts across all n values. In particular, the combining procedure was most effective when the initial component count is 2, as shown by dot sizes. Even at higher initial component counts, the combining procedure managed to reduce G to various degrees. This study demonstrates that component-combining, coupled with a model selection criterion that promotes cluster detection, can compliment the GHMR model.

6.3.4 Real Data Illustration 1: Fish Market

Seafood consumption is a major source of expenditure globally, with the 2018 aquaculture production value reaching USD 263.6 billion, according to the UN Food and Agriculture

Organization (FAO) (UN). In such a large industry, correctly classifying the seafood for sale is important for both consumers and sellers. Unfortunately, fish fraud also appears to be a large industry. Askew (2020) reported that 'the overall economic impact related to the diversion of fish from the legitimate trade system is costing us $26 billion to $50 billion globally'. An example of fish fraud is intentional mis-labelling. For instance, Warner et al. (2013) revealed that less than 1% of the seafood consumed in the United States is checked for fraud, and that 59% of tested fish types were mis-labelled. In particular, 44% of tested grocery stores, restaurants and sushi venues mis-labelled seafood. Thus, the detection of mis-labelled fish could leverage advanced statistical tools like the mixture regression model. To that end, we consider a subset of the Fish Market data from Pyae (2019), which consists of the Height (response variable) and the Width (covariate) of 55 fish. Two types of fish are present: Bream and Roach, representing 64% and 36% of the data set respectively.

Figure 6.6: Images of the common Bream (left) and the common Roach (right) fish. Sources: https://en.wikipedia.org/wiki/Common_bream and https://en.wikipedia.org/wiki/Common_roach.

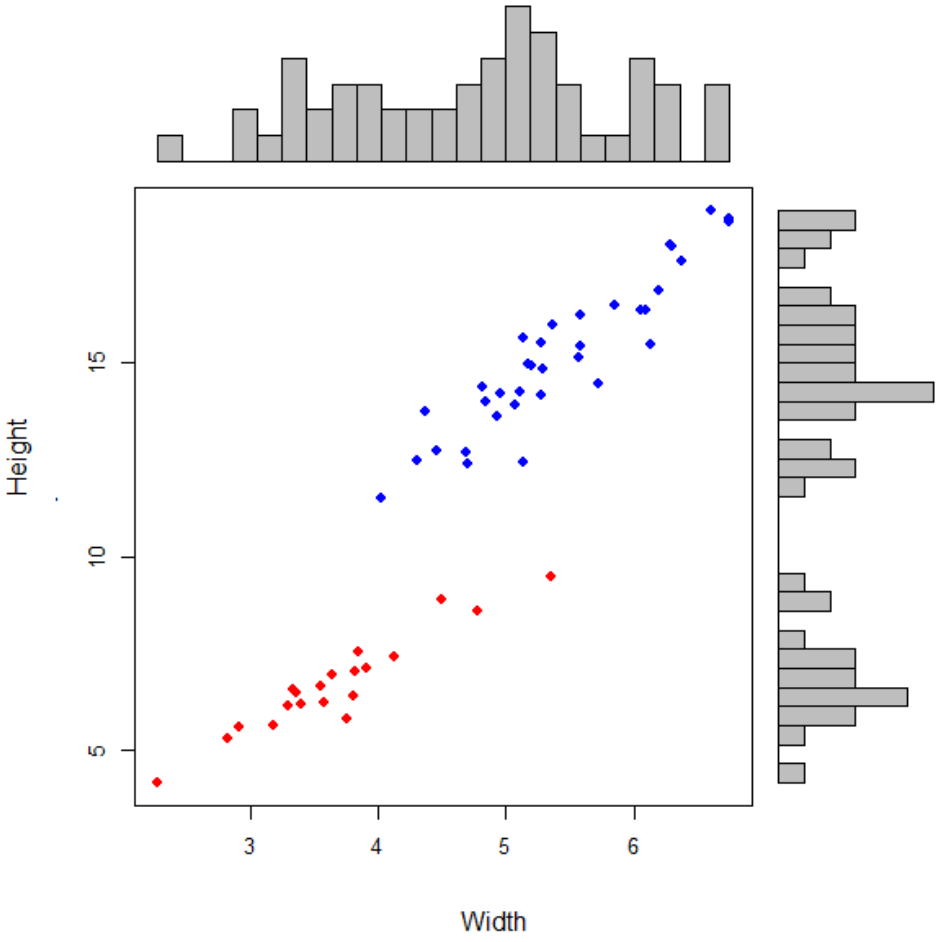

Figure 6.7: Scatterplot, with marginal histograms, of the Fish data. Breams are marked with blue dots and Roaches are marked with red dots.

From figure 6.6, Breams appear to be longer and more slender than Roaches. Figure 6.7 shows a clear bimodality in Height variable, with Breams scattered in a steeper slope than Roaches. The two species show a clear separation in their joint distribution. The GHMR, GMR, RGMR and TLE models are fitted over $G = 1, 2, \ldots, 6$ (except for GMR which starts with $G = 2$ since $G = 1$ is not supported software-wise), and we report on the best fits BIC-wise, based on 100 different initializations.

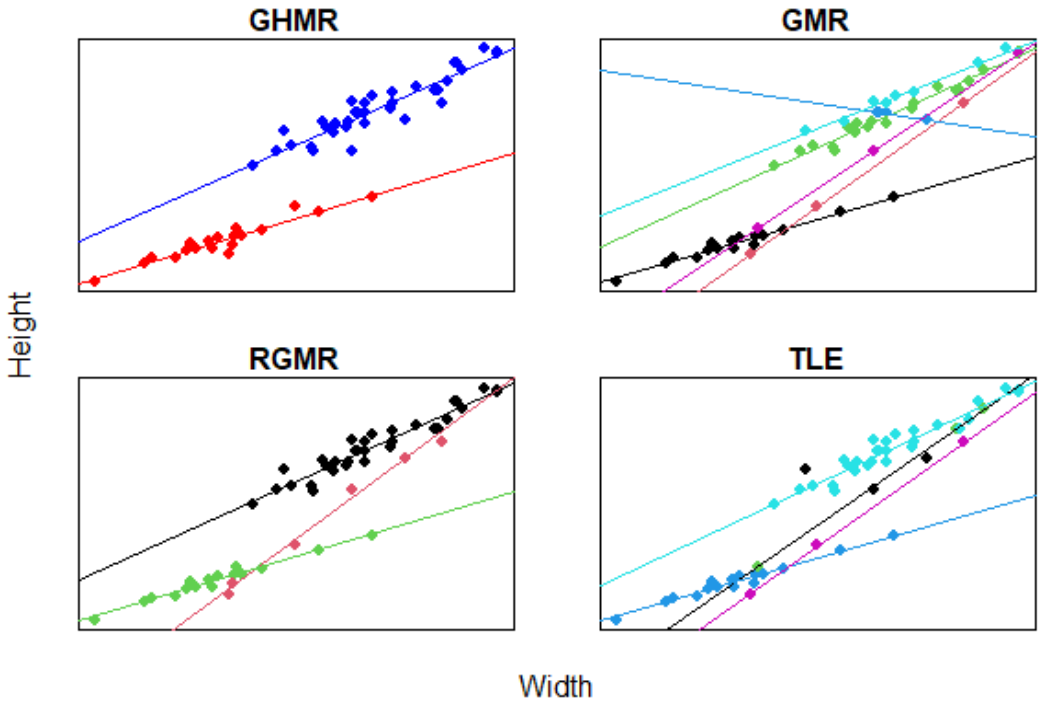

Figure 6.8: Colour-coded scatterplots of the Fish data with component-wise regression lines estimated by the mixture regression models. Each plot's heading indicates the line-generating model.

Table 6.4 and figure 6.8 summarize the the fit of the four models. Although the GMR achieved the highest BIC, its component count is the highest ($G = 6$) and its clustering result is the worst (ARI = 0.43). This suggests an overfit from the GMR, since the robust variants (RGMR and TLE) estimated reduced component counts with superior ARI. In contrast, the GHMR identified the fish groups perfectly. The trade-off is a lower BIC value, but this is likely due to a larger parameter set arising from the generalized hyperbolic distribution. Figure 6.7 shows a further evidence of overfitting, to varying degrees, by the GMR, RGMR and TLE. In particular, the GMR fitted three lines to the Breams (blue cluster in the GHMR plot), where one was sufficient for the GHMR. This is a case where a flexible distribution like the generalized hyperbolic distribution can help avoid spurious response-covariate relationships.

The non-negligible values β_g, λ_g, ω_g shown in table 6.5 suggest the non-normality of the cluster-wise distribution of Height variable. When the non-normality is not accounted for, the mixture regression model may overfit, as was the case here. The regression coefficients imply numerically that Breams are indeed longer and more slender than Roaches, though we may also observe more deviation in the estimated Height-Width relationship from Breams, as suggested by its larger σ_g^2 estimate. Overall, the Fish data shows that the GHMR model can be effective in identifying heterogeneous response-covariate relationships while accounting for a departure from normality.

6.3.5 Real Data Illustration 2: Italian Tourism

Tourism contributes significantly to the Italian economy. According to OECD, in 2017, tourism accounted for approximately 13% of Italy's Gross Domestic Product (GDP) and 14.7% of its workforce. As such, a deep understanding of tourists' behaviour would be of interest to Italy. The Italian tourism data (ISTAT) contains the national monthly visitor figures in Italy from January 1996 to December 2007. It is of 180 rows and 2 variables - overnight tourist count (Overn) and the visitor count to state museums, monuments and museum networks (MonMus). Beside the timestamp, the data set contains no ground truth labels. We want to study the type of heterogeneity in the association between Overn (covariate) and MonMus (response). Figure 6.9 suggests roughly three clusters - the right-side one consisting of July and August records, the middle one consisting of June and September records, and the left-side one consisting of the remaining months. This month-based separation is consistent with the known seasonality in Italian tourism. For instance, according to Travel and Leisure, the Summer months (May to September) comprise the peak tourist season. The data set visually suggests a steeper slope of MonMus against Overn across records from January to May and October and December (left-side cluster).

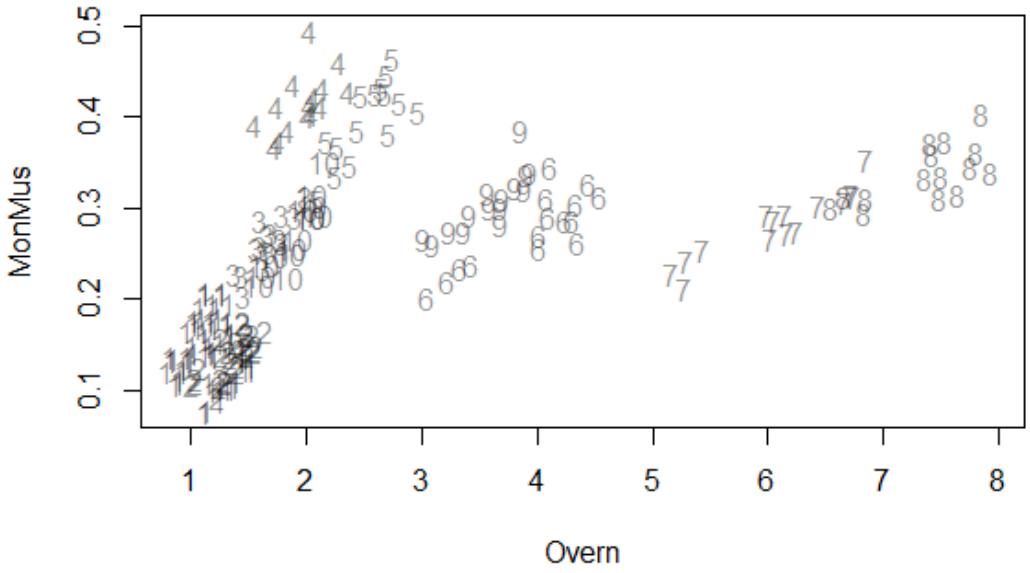

Figure 6.9: Scatterplot of the Tourism data where each point is numbered by month. For example, 1s denote observations from January, 2s denote observations from February, etc.. The variables' unit is ten million.

Similar to the Fish data analysis, the four models are fitted over $G = 1, \ldots, 6$ (except for GMR which starts with $G = 2$), and the BIC-wise best fit over 100 initializations is reported. To account for the scale of values (in tens of millions), the observations were divided by 10 million before model-fitting. However, as the TLE model diverged in all instances, we used an alternative scaling (dividing the original data by variable-wise standard deviation) before fitting a TLE model.

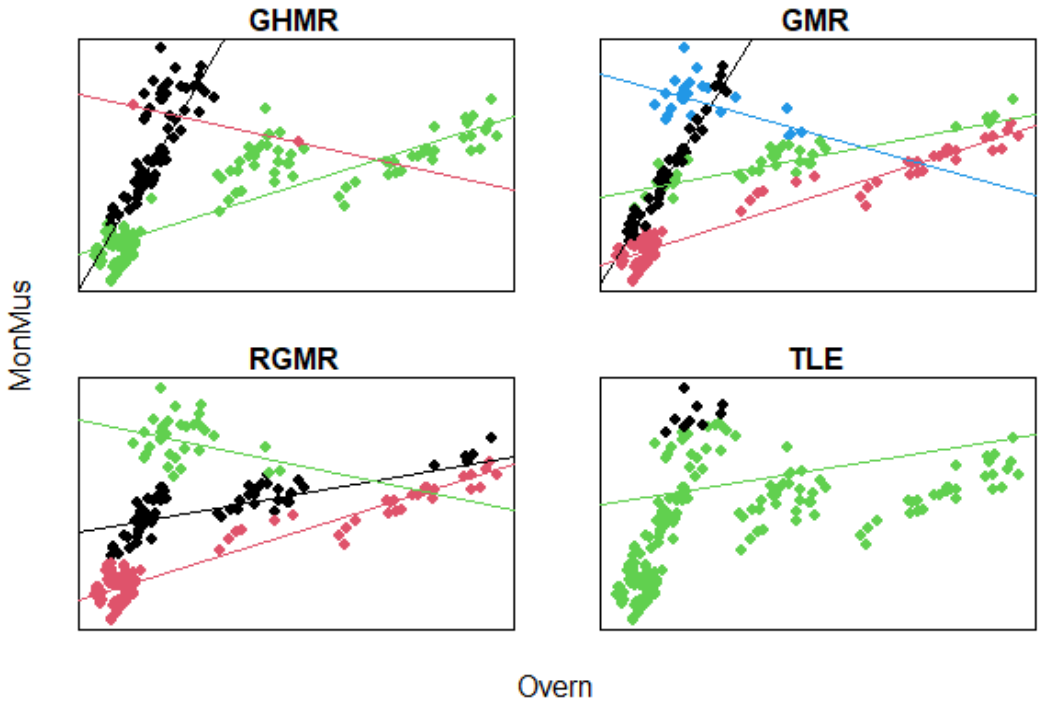

Figure 6.10: Colour-coded scatterplots of the Tourism data with component-wise regression lines estimated by the mixture regression models. Each plot's heading indicates the line-generating model. For TLE, the model did not estimate a regression vector for the black dots as it deemed them as outliers.

Figure 6.10 shows that the GHMR obtained the cleanest separation by month among the compared models. Its two main components, black and green, consist mostly of {March, April, May, October} and {January, February, June, July, August, September, November, December} respectively (shown in table 6.7). The RGMR identified similar, but more consolidated, association structures than the GMR. The TLE did not identify any meaningful heterogeneity from the data set, though it did capture the positive association between Overn and MonMus at a very high level. Thus, the GHMR seems to have captured the middle ground between GMR/RGMR (granular) and TLE (coarse) in detecting heterogeneous association.

We now focus on the GHMR. Consider the numeber of visitors to state museums, mon-

uments, archaeological site and museum complexes in Italy during 2019, obtained from Statistica (2019 figures were the only public and readily-accessible ones, to our best knowledge). The bottom plot in figure 6.11 shows that the GHMR produced an approximate division of months by the number of visitors to monuments and museums. When the regression slopes are incorporated, we can deduce that the peak monument and museum months exhibit a stronger positive association between the visitor count and the number of overnight tourists. Interestingly, the overall peak tourist months like July, August and September are grouped into a different component, which could be explained by the hot Summer weather that tends to favour outdoor activities. In terms of model parameter estimates, table 6.8 of component-wise distribution parameters suggests a significant departure from normality, which further substantiates the benefit of distributional flexibility in mixture regression modelling. Overall, the GHMR has demonstrated its value in analyzing a data set without an apparent ground truth label via a superior fit (as measured by BIC) and an informative starting point for further investigation.

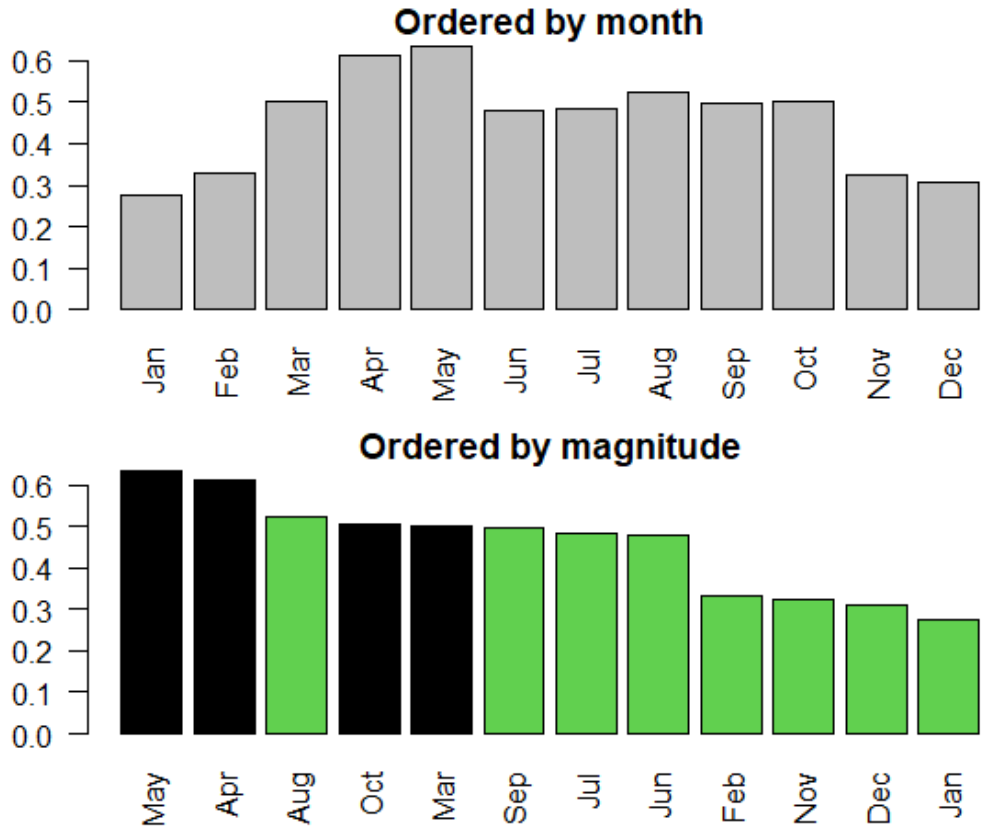

Figure 6.11: Bar plots of monthly visitors (in ten millions) to state museums, monuments, archaeological site and museum complexes in Italy during 2019. The top plot is ordered by month, and the bottom plot is ordered by magnitude, and colour-coded by components to which a majority of observations belong to. Data sourced from Statistica.

6.4 Discussion

In this chapter, a flexible mixture regression model with the generalized hyperbolic distribution was introduced, as well as an iterative component combining procedure. Simulated and real data sets have shown that the GHMR model can provide an edge against the ex-

isting models, and that it can be deployed as a flexible tool for regression analysis. Avenues for future work include the study of distributional properties of the regression coefficient estimator, extensions to parsimonious variants and the investigation of the model under the inclusion of categorical variables.

ICL	$n = 100$	$n = 200$	$n = 300$	$n = 400$
Before	-451	-1047	-1472	75
	(1344)	(2946)	(4239)	(6059)
After	**-28**	**-32**	**-34**	**964**
	(986)	(2357)	(3344)	(5151)

G	$n = 100$	$n = 200$	$n = 300$	$n = 400$
Before	2	2	2	2
	(1)	(1)	(1)	(2)
After	2	**1**	**1**	2
	(1)	(1)	(2)	(2)

	$n = 100$	$n = 200$	$n = 300$	$n = 400$
count$(G = 1)$	250	253	258	236

Table 6.3: Table of summary statistics obtained from the experiment conducted in section 6.3.2. The column labels denote the sample size under which the experiment was conducted. The top table records the median (and IQR in brackets) ICL values of the fitted GHMR model before and after combining. The middle table records the median (and IQR in brackets) G values of the fitted GHMR model before and after combining. The bottom table records the number of replications in which the component-combined GHMR model estimated 1 component. The ICL and G values are rounded to zero decimal places, and the best in-class values are bolded.

	GHMR	GMR	RGMR	TLE
BIC	-217.66	**-178.27**	-195.65	-192.14
G	**2**	6	3	4
ARI	**1**	0.43	0.79	0.68

Table 6.4: Table of the BIC, estimated component count (G) and ARI obtained by the GHMR, GMR, RGMR and TLE models based on 100 different initializations. The best in-class value is bolded.

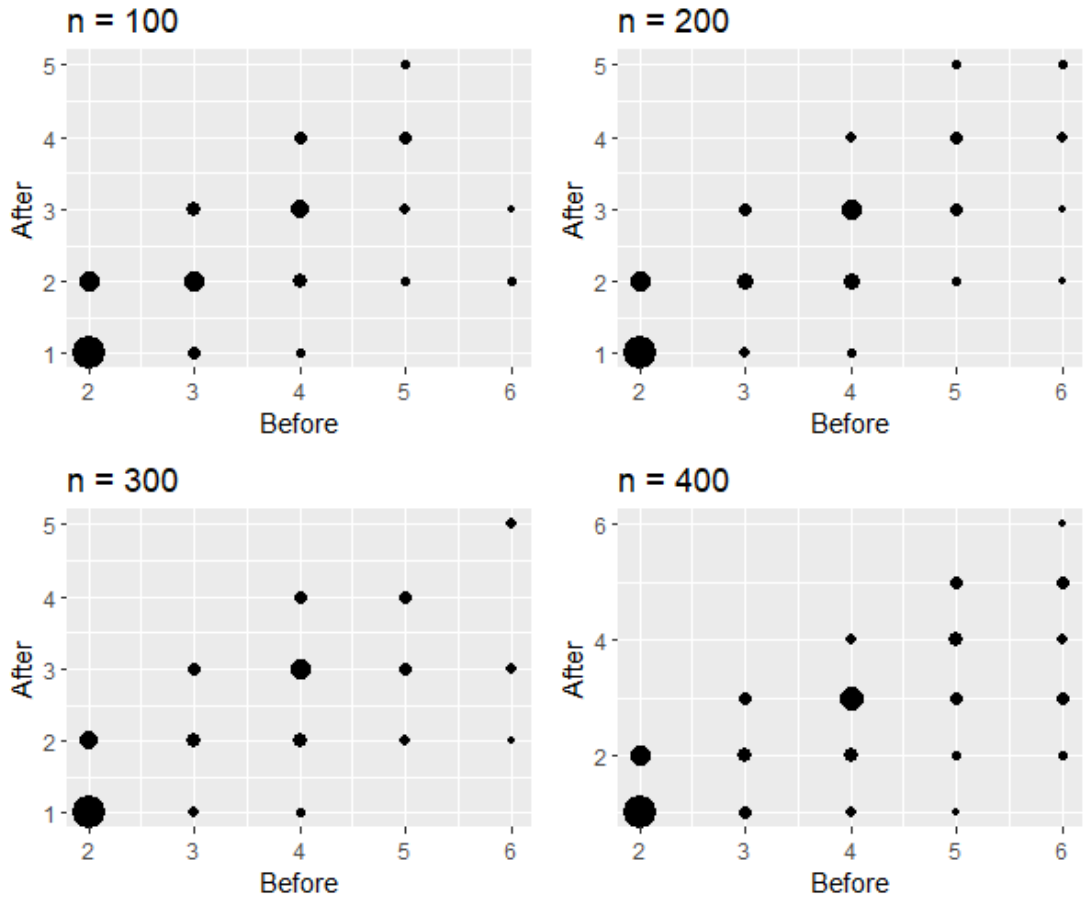

Figure 6.5: Scatterplots of number of components estimated before and after component-combining. Each plot's heading indicated the sample size under which the plot was generated. The dots' sizes are scaled by their frequency of occurrence. For instance, in the top-left plot, when the GHMR model initially estimated 2 components (leftmost horizontal axis value), the majority of replications resulted in 1 component after combining.

	Intercept	Width	σ_g^2	β_g	λ_g	ω_g	π_g
Bream	13.56	2.56	0.61	-0.03	-0.80	2.02	0.64
Roach	8.63	1.71	0.14	-0.01	-0.69	2.09	0.36

Table 6.5: Component-wise parameter estimates from the GHMR.

	GHMR	GMR	RGMR	TLE
BIC	**1905.88**	436.17	406.76	-523.81
G	3	4	3	1

Table 6.6: Table of the BIC and estimated component count (G) obtained by the GHMR, GMR, RGMR and TLE models based on 100 different initializations. The best BIC value is bolded.

Component	Jan	Feb	Mar	Apr	May	Jun
1 (black)			15	14	15	
2 (red)				1		1
3 (green)	15	15				14

	Jul	Aug	Sep	Oct	Nov	Dec
1 (black)				14	5	
2 (red)						
3 (green)	15	15	15	1	10	15

Table 6.7: Table of component-wise month distribution generated by the GHMR. Empty cell indicates zero observation.

	Intercept	MonMus	σ_g^2	β_g	ω_g	λ_g	π_g
Comp 1	-0.04	0.18	12.31	-0.04	0.0003	-1.39	0.39
Comp 3	0.11	0.03	377.25	-0.08	1.00×10^{-5}	-1.42	0.60

Table 6.8: Distribution parameter estimates for the major components from the GHMR, rounded to two decimal places.

Chapter 7

Mode Merging for a Finite Mixture of t-distributions

7.1 Introduction

Finite mixture models can be interpreted as a model representing heterogeneous subpopulations within the whole population. However, as discussed in Hennig (2010), more care is needed when associating a mixture component with a cluster. Consider a finite mixture where each component is unimodal (this includes the popular Gaussian finite mixture). By interpreting each component as a cluster, an implicit assumption of unimodality on the shape of each cluster is imposed, along with other features of the component distribution, such as the spread, skewness and the heaviness of tails. Although this example does not mean the component-cluster association is always incorrect, it opens up the possibility of associating a cluster with a union of components.

7.1.1 The Ridgeline Function and the Mean-shift

A method for merging mixture components is modal clustering via the mean-shift algorithm (Comaniciu and Meer, 2002). It is a fixed-point algorithm that seeks out the nearest

mode (local maximizer) of a density function. For a G-component Gaussian finite mixture model (GMM) with component mixing, location and covariance parameters π_g, $\boldsymbol{\mu}_g$ and $\boldsymbol{\Sigma}_g$ respectively, the mean shift formula is given by Chacón (2019):

$$\boldsymbol{m}^{(t+1)} = \left[\sum_{g=1}^{G} z_g^{(t)} \boldsymbol{\Sigma}_g^{-1}\right]^{-1} \left[\sum_{g=1}^{G} z_g^{(t)} \boldsymbol{\Sigma}_g^{-1} \boldsymbol{\mu}_g\right], \tag{7.1}$$

where $\boldsymbol{m}^{(t)}$ is the solution of the t^{th} iteration of the algorithm,

$$z_g^{(t)} = \pi_g \phi(\boldsymbol{m}^{(t)}; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g) / \sum_{k=1}^{G} \pi_g \phi(\boldsymbol{m}^{(t)}; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g),$$

and $\phi(\cdot; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$ denotes the single-component Gaussian density. The algorithm is run until convergence with initialization $\boldsymbol{m}^{(0)} = \boldsymbol{\mu}_g$ for $g = 1, 2, \ldots, G$. The resulting mode estimate $\hat{\boldsymbol{m}}$ can be seen as a maximum likelihood estimate of the mode parameter $\boldsymbol{m}$, as Carreira-Perpinan (2007) showed that the mean-shift algorithm for GMM is an EM algorithm. In fact, depending on the initialization $\boldsymbol{m}^{(0)}$, it is possible to discover all modes of the finite mixture density through mean shift. Ray and Lindsay (2005) defined the ridgeline function for a finite Gaussian mixture as

$$\boldsymbol{m}(\boldsymbol{\alpha}) = \boldsymbol{m}(\alpha_1, \ldots, \alpha_G) = \left[\sum_{g=1}^{G} \alpha_g \boldsymbol{\Sigma}_g^{-1}\right]^{-1} \left[\sum_{g=1}^{G} \alpha_g \boldsymbol{\Sigma}_g^{-1} \boldsymbol{\mu}_g\right], \tag{7.2}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_G)$ is a point on a $(G-1)$-dimensional unit simplex defined as

$$\mathcal{S}_G = \left\{ \boldsymbol{\alpha} \in [0,1]^G : \sum_{g=1}^{G} \alpha_g = 1 \right\}.$$

Then, Ray and Lindsay (2005) showed that the ridgeline function (7.2) defined on $\mathcal{S}_G$ contains all the critical values of the GMM density, including all modes. This property is useful because the number of modes of finite Gaussian mixture model is not always obvious. In fact, such a model can have more than G modes, as shown by Ray and Ren (2012).

Since each $z_g^{(t)}$ is non-negative and $z_1^{(t)} + \cdots + z_G^{(t)} = 1$, it is clear that a mean-shift estimate $\boldsymbol{m}^{(t)}$ at any (t) corresponds to a point on the ridgeline function surface over $\mathcal{S}_G$. Therefore, regardless of the number of variables p in the data set, the mean shift algorithm for GMM can be interpreted a search for the density modes over a constrained $(G-1)$-dimensional space. If $p > G-1$, then the mean shift can also be seen as a dimension reduction tool.

7.1.2 Finite Mixture of t-distributions

The model of interest in this paper is a finite mixture of t-distributions by Peel and McLachlan (2000). Let a random vector $\boldsymbol{U}$ follow a p-dimensional Gaussian distribution with zero mean and covariance $\boldsymbol{\Sigma}$, $N_p(\boldsymbol{0}, \boldsymbol{\Sigma})$. Also, suppose $Y \sim Gamma(\nu/2, \nu/2)$ (shape-rate parametrization) to be independent of $\boldsymbol{U}$. Then, if a random vector $\boldsymbol{X}$ follows a p-dimensional t-distribution with location vector $\boldsymbol{\mu}$, scale matrix $\boldsymbol{\Sigma}$ and degrees of freedom parameter $\nu > 0$, its stochastic relationship can be written as

$$\boldsymbol{X} = \boldsymbol{\mu} + \frac{\boldsymbol{U}}{\sqrt{Y}}.$$

Let $f_t(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu)$ denote the density function of the above p-dimensional t-distribution, $t_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu)$. Then, the marginal density function of a G-component mixture of t-distributions, abbreviated as tMM, is written as

$$f(\boldsymbol{x}; \boldsymbol{\Theta}) = \sum_{g=1}^{G} \pi_g f_t(\boldsymbol{x}; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \nu_g), \tag{7.3}$$

where $\pi_g > 0$, $\boldsymbol{\mu}_g$, $\boldsymbol{\Sigma}_g$ and ν_g denote the component-wise proportion, location, scale and degrees of freedom parameters respectively. We will use $\boldsymbol{\Theta}$ to denote the set of model parameters for the finite mixture. Now suppose we have n many independent observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ from the above tMM. With the incorporation of latent component membership indicators $\boldsymbol{Z}_i = (Z_{i1}, \ldots, Z_{iG})'$ (outlined in chapter 2.1), the conditional distributions

pertaining to $\boldsymbol{X}_i$ and Y_{ig} are given as follows.

$$\begin{aligned}
&\boldsymbol{X}_i|Z_{ig}=1 \sim t_p(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \nu_g),\\
&Y_{ig}|Z_{ig}=1 \sim Gamma\left(\frac{\nu_g}{2}, \frac{\nu_g}{2}\right),\\
&\boldsymbol{X}_i|Z_{ig}=1, y_{ig} \sim N_p\left(\boldsymbol{\mu}_g, \frac{1}{y_{ig}}\boldsymbol{\Sigma}_g\right),\\
&Y_{ig}|Z_{ig}=1, \boldsymbol{x}_i \sim Gamma\left(\frac{\nu_g+p}{2}, \frac{\nu_g+\delta(\boldsymbol{x}_i, \boldsymbol{\mu}_g; \boldsymbol{\Sigma}_g)}{2}\right),
\end{aligned}$$

where $\delta(\boldsymbol{x}_i, \boldsymbol{\mu}_g; \boldsymbol{\Sigma}_g) = (\boldsymbol{x}_i - \boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_g)$. Andrews and Mcnicholas (2012) introduced a framework for fitting parsimonious variants of the tMM, implemented as a R package *teigen* (Andrews et al., 2018). The number of free parameters in the model is reduced through the following eigen-decomposition of component-wise scale matrices: $\boldsymbol{\Sigma}_g = \lambda_g \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g'$, where $\boldsymbol{P}_g$ is the matrix of eigenvectors, $\boldsymbol{D}_g$ is the diagonal matrix whose diagonal entries are proportional to the eigenvalues and λ_g is the proportionality constant for the eigenvalues. Various levels of parameter reduction is obtained by combining some or all of the following constraints: $\boldsymbol{P}_g = \boldsymbol{P}$, $\boldsymbol{P}_g = \boldsymbol{I}$, $\boldsymbol{D}_g = \boldsymbol{D}$, $\boldsymbol{D}_g = \boldsymbol{I}$ and $\lambda_g = \lambda$. This strategy can be beneficial when modelling high-dimensional data, because the number of free parameters in a unconstrained scale matrix is $p(p+1)/2$ (p is the data's dimension), which increases quadratically with respect to p.

Parameter estimation for tMM is done by the EM algorithm, as shown in Peel and McLachlan (2000). Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ denote the set of n independent observations from a G-component tMM. Then, with the latent variables $\boldsymbol{z}_i$ and $(y_{i1}, \ldots, y_{iG})'$ from section 7.1.2, we obtain the following complete-data log-likelihood l_c.

$$l_c(\boldsymbol{\Theta}) = \sum_{i=1}^{n}\sum_{g=1}^{G} z_{ig}\left[\log(\pi_g) + \log\phi\left(\boldsymbol{x}_i; \boldsymbol{\mu}_g, \frac{\nu_g}{y_{ig}}\boldsymbol{\Sigma}_g\right) + \log h(y_{ig}; \nu_g)\right], \tag{7.4}$$

where $\boldsymbol{\Theta}$ denotes the set of model parameters, and $h(\cdot; \nu)$ denotes the pdf of $Gamma(\nu/2, \nu/2)$. At iteration t of the EM algorithm, we calculate the conditional expectation of $l_c(\boldsymbol{\Theta})$ with respect to the latent variables given $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and the current parameter estimates $\boldsymbol{\Theta}^{(t)}$,

denoted by $Q\left(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}\right)$.

$$Q\left(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}\right) = \sum_{i=1}^{n}\sum_{g=1}^{G} z_{ig}^{(t)}\Big[\log(\pi_g) - \frac{1}{2}\log(|2\pi\boldsymbol{\Sigma}_g|) - \frac{u_{ig}^{(t)}}{2}\delta(\boldsymbol{x}_i, \boldsymbol{\mu}_g; \boldsymbol{\Sigma}_g) - \log\Gamma\left(\frac{\nu_g}{2}\right) + \frac{\nu_g}{2}\log\left(\frac{\nu_g}{2}\right) + \frac{\nu_g}{2}\left(w_{ig}^{(t)} - u_{ig}^{(t)}\right) - w_{ig}^{(t)}\Big], \quad (7.5)$$

where

$$z_{ig}^{(t)} = \mathrm{E}[Z_{ig}|\boldsymbol{x}_i, \boldsymbol{\Theta}^{(t)}] = \frac{\pi_g^{(t)} f_t\left(\boldsymbol{x}; \boldsymbol{\mu}_g^{(t)}, \boldsymbol{\Sigma}_g^{(t)}, \nu_g^{(t)}\right)}{\sum_{k=1}^{G}\pi_k^{(t)} f_t\left(\boldsymbol{x}; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}, \nu_k^{(t)}\right)},$$

$$u_{ig}^{(t)} = \mathrm{E}[Y_{ig}|Z_{ig} = 1, \boldsymbol{x}_i, \boldsymbol{\Theta}^{(t)}] = \frac{\nu_g^{(t)} + p}{\nu_g^{(t)} + \delta\left(\boldsymbol{x}_i, \boldsymbol{\mu}_g^{(t)}; \boldsymbol{\Sigma}_g^{(t)}\right)}$$

$$w_{ig}^{(t)} = \mathrm{E}[\log(Y_{ig})|Z_{ig} = 1, \boldsymbol{x}_i, \boldsymbol{\Theta}^{(t)}] = \log\left(u_{ig}^{(t)}\right) + \psi\left(\frac{\nu_g^{(t)} + p}{2}\right) - \log\left(\frac{\nu_g^{(t)} + p}{2}\right),$$

and $\psi(\cdot)$ is the digamma function. Then, by differentiating with respect to each parameter, the new component-wise estimates $\pi_g^{(t+1)}$, $\boldsymbol{\mu}_g^{(t+1)}$, $\boldsymbol{\Sigma}_g^{(t+1)}$ and $\nu_g^{(t+1)}$ are obtained.

7.2 Methodology

In this section, we present the main contribution of this paper. We derive the mean-shift algorithm for tMM and its parsimonious variants. The goal of the algorithm is to estimate the modes of the density function of the tMM.

t Mean-shift

Using the parameter estimates $\hat{\boldsymbol{\Theta}}$ obtained by EM algorithm in section 7.1.2, we want to find the modes of the tMM density f. The t-distribution's density is not as well-behaved as that of Gaussian distribution, so a direct differentiation is difficult. Instead, we construct

an EM algorithm to find the modes. Similar to the complete-data log-likelihood in (7.4), we introduce the latent variables Y_g and Z_g representing the Gamma distribution and component membership associated with a mode vector $\boldsymbol{m}$. Given the current mode estimate $\boldsymbol{m}^{(t)}$, the expected complete-data log-likelihood with respect to the mode $\boldsymbol{m}$ depends only on the conditional Gaussian densities.

$$\mathcal{K}(\boldsymbol{m}|\hat{\boldsymbol{\Theta}}, \boldsymbol{m}^{(t)}) = \sum_{g=1}^{G} z_g^{(t)} \left(-\frac{u_g^{(t)}}{2} (\boldsymbol{m} - \hat{\boldsymbol{\mu}}_g)' \hat{\boldsymbol{\Sigma}}_g^{-1} (\boldsymbol{m} - \hat{\boldsymbol{\mu}}_g) \right) + \text{const}, \tag{7.6}$$

where 'const' is a collection of additive constants,

$$z_g^{(t)} = \frac{\hat{\pi}_g f_t \left(\boldsymbol{m}^{(t)}; \hat{\boldsymbol{\mu}}_g, \hat{\boldsymbol{\Sigma}}_g, \hat{\nu}_g \right)}{\sum_{k=1}^{G} \hat{\pi}_k f_t \left(\boldsymbol{m}^{(t)}; \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k, \hat{\nu}_k \right)}, \quad \text{and} \quad u_g^{(t)} = \frac{\hat{\nu}_g + p}{\hat{\nu}_g + \delta \left(\boldsymbol{m}^{(t)}, \hat{\boldsymbol{\mu}}_g; \hat{\boldsymbol{\Sigma}}_g \right)}.$$

The new mode estimate $\boldsymbol{m}^{(t+1)}$ is obtained by differentiating $\mathcal{K}$ with respect to $\boldsymbol{m}$:

$$\boldsymbol{m}^{(t)} = \left[\sum_{g=1}^{G} z_g^{(t)} u_g^{(t)} \hat{\boldsymbol{\Sigma}}_g^{-1} \right]^{-1} \left[\sum_{g=1}^{G} z_g^{(t)} u_g^{(t)} \hat{\boldsymbol{\Sigma}}_g^{-1} \hat{\boldsymbol{\mu}}_g \right]. \tag{7.7}$$

The resultant merging rule is as follows. Firstly, run the t mean-shift until convergence G times using each of the component mean vectors $\hat{\boldsymbol{\mu}}_g$ as initial values. Then, the mixture components whose mode estimates are equal are assigned to a single cluster. However, exact equality between two modes would be computationally too strict. Thus, we can relax this rule to merge two components if the corresponding modes $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ are such that $||\boldsymbol{m}_1 - \boldsymbol{m}_2||_2 < c$, for a pre-determined threshold $c > 0$. In this paper, c is set at 0.01.

Connection to the Ridgeline Function

The update equation in (7.7) shares similar properties the Gaussian ridgeline function in (7.1). Recall that the Gaussian mean shift can be seen as a search algorithm for critical points of the Gaussian ridgeline function. Similarly, the mean shift algorithm for tMM

can be interpreted as a search algorithm for critical points of the conditional ridgeline function of tMM, which is defined as the right-hand side expression in equation (7.7). Indeed, because the complete-data expectation $\mathcal{K}(\boldsymbol{m}|\hat{\boldsymbol{\Theta}}, \boldsymbol{m}^{(t)})$ is continuous with respect to both $\boldsymbol{m}$ and $\boldsymbol{m}^{(t)}$, Theorem 2 of Wu (1983) guarantees the convergence of the EM estimate sequence $\{\boldsymbol{m}^{(t)}\}_{t=1}^{\infty}$ to a critical point of the tMM density function. Note that, if all $u_g^{(t)}$ is set to be 1 and the densities involved in $z_g^{(t)}$ are changed to those from GMM, then the Gaussian mean shift is recovered.

Parsimonious Variants

The mean shift solution in equation (7.7) can be simplified depending on the parsimonious model fitted from the tMM family. This is highly convenient, because all component-wise scale matrices must be inverted otherwise. Among all models in this family, the most significant simplifications come from those with constraints on the scale matrices $\boldsymbol{\Sigma}_g = \lambda_g \boldsymbol{P}_g \boldsymbol{D}_g \boldsymbol{P}_g'$. In this section, we present some of the most simplified variants, in a decreasing order of parsimony.

- If $(\lambda_g, \boldsymbol{P}_g, \boldsymbol{D}_g) = (\lambda, \boldsymbol{P}, \boldsymbol{D})$ for all g, then all components share a common scale matrix. Hence, the component-wise scale matrices can be factored out, resulting in the most simplified form below,

$$\boldsymbol{m}^{(t+1)} = \frac{\sum_{g=1}^{G} z_g^{(t)} u_g^{(t)} \hat{\boldsymbol{\mu}}_g}{\sum_{g=1}^{G} z_g^{(t)} u_g^{(t)}}.$$

- If $(\lambda_g, \boldsymbol{P}_g, \boldsymbol{D}_g) = (\lambda_g, \boldsymbol{P}, \boldsymbol{D})$ for all g, then each component is entitled to its own scaling factor for the scale matrix. Let $\hat{\lambda}_g$ denote the EM estimate of λ_g. Then, we can still obtain a simple weighted sum,

$$\boldsymbol{m}^{(t+1)} = \frac{\sum_{g=1}^{G} \left(z_g^{(t)} u_g^{(t)} / \hat{\lambda}_g \right) \hat{\boldsymbol{\mu}}_g}{\sum_{g=1}^{G} z_g^{(t)} u_g^{(t)} / \hat{\lambda}_g}.$$

- If $(\lambda_g, \boldsymbol{P}_g, \boldsymbol{D}_g) = (\lambda_g, \boldsymbol{P}, \boldsymbol{D}_g)$ for all g, then the update formula can be simplified still, though not as much as the earlier cases. Let $\widehat{\boldsymbol{D}}_g$ and $\hat{\boldsymbol{P}}$ denote the EM estimates of $\boldsymbol{D}_g$ and $\boldsymbol{P}$ respectively. Because inversion is required only on the diagonal matrix, we have

$$\boldsymbol{m}^{(t+1)} = \hat{\boldsymbol{P}} \left[\sum_{g=1}^{G} \frac{z_g^{(t)} u_g^{(t)}}{\hat{\lambda}_g} \widehat{\boldsymbol{D}}_g^{-1} \right]^{-1} \left[\sum_{g=1}^{G} \frac{z_g^{(t)} u_g^{(t)}}{\hat{\lambda}_g} \widehat{\boldsymbol{D}}_g^{-1} \hat{\boldsymbol{P}}' \hat{\boldsymbol{\mu}}_g \right].$$

The other models in the *t*eigen family can simplify the formula as well, but the extent of reduction is less significant compared to the ones presented here. In general, we can see that the t mean-shift algorithm can inherit the parsimony constraints imposed on the mixture.

7.3 Numerical Experiments

In this section, we study the performance of *t*MM mode merging through simulated and real data experiments. Specifically, three 2-dimensional and one 6-dimensional simulated data sets at various sample sizes will be clustered, as well as the Old Faithful (Härdle et al., 1991) and the Chronic Kidney Disease (Dua and Graff, 2017) data sets. The following methods are deployed in all experiments herein. Each deployed method will be referred to using their corresponding abbreviations hereafter.

- Parsimonious finite mixture of Gaussian distributions implemented in the R package *mclust* (Scrucca et al., 2016) (GMM)
- Parsimonious finite mixture of t distributions implemented in the R package *teigen* (Andrews et al., 2018) (*t*EIGEN)
- GMM with overlap-based component merging 'DEMP+' by Melnykov (2016) (G-DEMP+): DEMP+ is a hierarchical component merging method that uses a measure called DEMP+ that indicates the degree of overlap between a pair of components.

At each stage of merging, a pair of clusters with the largest DEMP+ value is chosen for merging. If no pairs of clusters yield DEMP+ values beyond the threshold, the algorithm is terminated, and the resulting clusters are used. Note that a cluster may contain multiple components. The heuristic overlap threshold suggested by Melnykov (2016) is 0.1, which we use in this paper. The DEMP+ method is outlined in chapter 2.3.

- tEIGEN with DEMP+ (t-DEMP+)
- Mode merging for GMM by Chacón (2019) (G-Mode)
- t-Mean Shift (t-Mode)

Computational Aspects

All deployed methods are run until convergence. In addition, for all experiments, the set of the number of mixture components considered is $\{1, 2, \ldots, 8\}$. For a fair comparison with GMM-based methods, the simplified formulae presented in section 7.2 are not considered in the experiments. For G-Mode and t-Mode, the tolerance threshold for convergence is set at $\epsilon = 10^{-6}$. This means that the mode merging was deemed converged when $||\boldsymbol{m}^{(t+1)} - \boldsymbol{m}^{(t)}||_2 < \epsilon$. The experiments are run on a 2-socket, 24-core cluster of Intel Xeon E5-2690 v3 CPUs, clocked at 2.60 GHz.

As pointed out by Hennig (2010), the mixture component merging problem is not identifiable in terms of the model likelihood, because it does not re-fit the model based on the merged components. Thus, model-free performance measurements are needed, so the following metrics are used.

- The number of components estimated by each method (G).
- Adjusted Rand Index (ARI) as outlined in chapter 2.1.
- Additive Margin (AM) as outlined in chapter 2.1.
- Elapsed time measured in seconds (Time).

Simulated Data

In this experiment, we simulate data sets from four different finite mixture distributions. For each data-generating model, n_g observations are generated for all components. Several n_g values are considered: $n_g = 50, 100, 150, 200$. Below is the list of models and their parameters. The contour plots of 2-dimensional models are shown in figure 7.1.

(D1) A 3-component mixture of 2-dimensional skew-normal distributions. Each component is well-separated, resulting in three clusters. The parametrization of skew-normal distribution follows Lin et al. (2016), where $\boldsymbol{\mu}_g$, $\boldsymbol{\Sigma}_g$ and $\boldsymbol{\lambda}_g$ denote the component-wise location vector, scale matrix and skewness vector. The model parameters of the mixture are as follows,

$$\pi_1 = 0.2,\ \boldsymbol{\mu}_1 = (2, -4)',\ \boldsymbol{\lambda}_1 = (3, 3)',\ \boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & -0.1 \\ -0.1 & 1 \end{bmatrix},$$
$$\pi_2 = 0.3,\ \boldsymbol{\mu}_2 = (3.5, 2.5)',\ \boldsymbol{\lambda}_2 = (1, 5)',\ \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \text{ and}$$
$$\pi_3 = 0.5,\ \boldsymbol{\mu}_3 = (0, 0)',\ \boldsymbol{\lambda}_3 = (-3, 1)',\ \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

(D2) A 3-component mixture of 2-dimensional t distributions. Two of the components overlap to form a X-shaped cluster, and the third one forms a circular stand-alone cluster. Denote by $\boldsymbol{\mu}_g$, $\boldsymbol{\Sigma}_g$ and ν_g the location vector, scale matrix and the degrees of

freedom respectively as before. The model parameters of this mixture is as follows.

$$\pi_1 = 1/3,\ \boldsymbol{\mu}_1 = (5,-1)',\ \nu_1 = 2,\ \boldsymbol{\Sigma}_1 = 2 \times \begin{bmatrix} 1 & 0.7 \\ 0.7 & 0.6 \end{bmatrix}$$
$$\pi_2 = 1/3,\ \boldsymbol{\mu}_2 = (5,-1)',\ \nu_2 = 2,\ \boldsymbol{\Sigma}_2 = 2 \times \begin{bmatrix} 0.6 & -0.7 \\ -0.7 & 1 \end{bmatrix}$$
$$\pi_3 = 1/3,\ \boldsymbol{\mu}_3 = (0,0)',\ \nu_3 = 2,\ \boldsymbol{\Sigma}_3 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

(D3) A 6-component mixture of 2-dimensional Gaussian distributions from Baudry et al. (2010). This is a 6-component Gaussian mixture with 4 modes. There are two pairs of components where each pair is overlapped into a cross-shaped cluster, resulting in four clusters. Denote by $\boldsymbol{\mu}_g$ and $\boldsymbol{\Sigma}_g$ the component-wise location vector and covariance matrix. The model parameters are defined as follows,

$$\boldsymbol{P} = 0.5 \times \begin{bmatrix} 1 & -\sqrt{3} \\ \sqrt{3} & 1 \end{bmatrix},\ \boldsymbol{D}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix},\ \boldsymbol{D}_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 1 \end{bmatrix},$$
$$\pi_1 = \pi_2 = \pi_3 = \pi_4 = 0.2,\ \pi_5 = \pi_6 = 0.1,$$
$$\boldsymbol{\mu}_1 = (0,0)',\ \boldsymbol{\mu}_2 = (8,5)',\ \boldsymbol{\mu}_3 = \boldsymbol{\mu}_4 = (1,5)',\ \boldsymbol{\mu}_5 = \boldsymbol{\mu}_6 = (8,0)',\ \text{and}$$
$$\boldsymbol{\Sigma}_1 = \boldsymbol{P}\boldsymbol{D}_1\boldsymbol{P}',\ \boldsymbol{\Sigma}_2 = \boldsymbol{P}'\boldsymbol{D}_1\boldsymbol{P},\ \boldsymbol{\Sigma}_3 = \boldsymbol{\Sigma}_5 = \boldsymbol{D}_1,\ \boldsymbol{\Sigma}_4 = \boldsymbol{\Sigma}_6 = \boldsymbol{D}_2.$$

(D4) A 3-component mixture of 6-dimensional skew-normal distributions. This is the 6-dimensional analogue of model (D1). Using the same parametrization as (D1), the model parameters are as follows,

$$\pi_1 = \pi_2 = \pi_3 = 1/3,\ \text{and}$$
$$\boldsymbol{\mu}_1 = \boldsymbol{\mu}_3 = (-1,1,-1,1,-1,1)',\ \boldsymbol{\mu}_2 = (5,5,5,5,5,5)'.$$

The component scale matrices are tri-diagonal matrices. For $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$, the diagonal

entries are 2, 2, 1.6, 1.6, 1.6, 1.6, and the super and sub-diagonal entries are 0.9. For $\mathbf{\Sigma}_3$, the diagonal entries are all 2, and the super and sub-diagonal entries are -1.

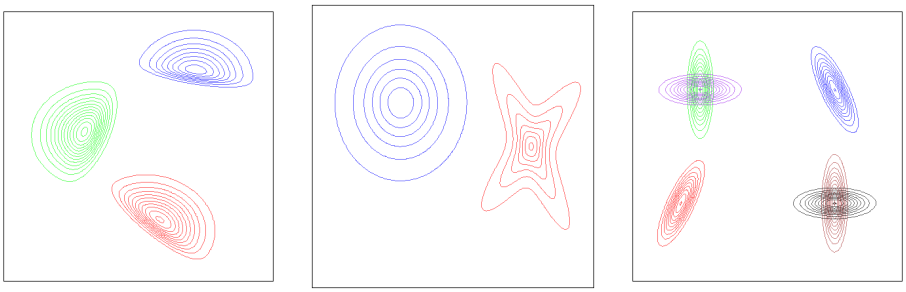

Figure 7.1: From left to right: Contour plots of the models (D1), (D2) and (D3). The contour plot for (D4) is omitted as it is more than 2-dimensional.

(D1), $n_g = 50$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	3	3	1	1
Range(G)	[3, 6]	[3, 5]	[3, 5]	[3, 5]	[1, 3]	[1, 3]
Median(AM)	1.795	1.797	1.809	1.808	0 (1.775)	0 (0.4)
Median(ARI)	**1**	**1**	**1**	**1**	0 (1)	0 (0.535)
Median(Time)	0.407	27.185	0.002	0.002	0.434	0.441

(D1), $n_g = 100$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	3	3	1	1
Range(G)	[3, 5]	[3, 5]	[3, 4]	[3, 5]	[1, 4]	[1, 4]
Median(AM)	1.770	1.767	1.797	1.790	0 (1.779)	0 (1.341)
Median(ARI)	**1**	**1**	**1**	**1**	0 (0.990)	0 (0.892)
Median(Time)	1.270	66.300	0.001	0.001	0.500	0.505

(D1), $n_g = 150$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	3	3	1	1
Range(G)	[3, 7]	[3, 6]	[3, 6]	[3, 4]	[1, 4]	[1, 4]
Median(AM)	1.720	1.727	1.788	1.781	0 (1.780)	0 (1.779)
Median(ARI)	0.993	0.993	**1**	**1**	0 (0.993)	0 (0.993)
Median(Time)	1.479	83.299	0.002	0.002	0.433	0.441

(D1), $n_g = 200$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	3	3	1	1
Range(G)	[3, 7]	[3, 7]	[3, 5]	[3, 5]	[1, 4]	[1, 4]
Median(AM)	1.689	1.639	1.782	1.782	0 (1.784)	0 (1.031)
Median(ARI)	0.989	0.984	**0.995**	**0.995**	0 (0.995)	0 (0.812)
Median(Time)	2.785	141.035	0.001	0.01	0.520	0.570

Table 7.1: Tables of summary statistics pertaining to the tested methods, where the data set is generated by models (D1). For example, the top table is constructed from data sets with $n_g = 50$. For median(AM) and median(ARI), the bracketed numbers for DEMP+ methods are the corresponding median values among replications that identified more than one cluster. The highest ARI values in each table are bolded.

(D2), $n_g = 50$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	4	2	2	2	1	1
Range(G)	[2, 8]	[1, 4]	[1, 7]	[1, 3]	[1, 3]	[1, 3]
Median(AM)	0.375	0.299	0.395	0.430	0 (0.581)	0 (0.508)
Median(ARI)	0.344	0.410	0.559	**0.653**	0 (0.488)	0 (0.656)
Median(Time)	0.610	4.595	0.015	0.001	1.225	0.135

(D2), $n_g = 100$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	5	3	2	2	1	1
Range(G)	[3, 7]	[1, 5]	[2, 5]	[1, 4]	[1, 1]	[1, 2]
Median(AM)	0.406	0.282	0.439	0.431	0	0 (0.497)
Median(ARI)	0.352	0.437	0.635	**0.743**	0	0 (0.717)
Median(Time)	1.330	24.635	0.020	0.020	2.255	0.475

(D2), $n_g = 150$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	5	3	2	2	1	1
Range(G)	[4, 7]	[1, 5]	[2, 4]	[1, 3]	[1, 1]	[1, 3]
Median(AM)	0.369	0.267	0.417	0.411	0	0 (0.458)
Median(ARI)	0.332	0.428	0.638	**0.725**	0	0 (0.695)
Median(Time)	2.115	16.735	0.020	0.001	2.355	0.495

(D2), $n_g = 200$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	5	3	2	2	1	1
Range(G)	[4, 8]	[1, 4]	[2, 6]	[1, 3]	[1, 1]	[1, 2]
Median(AM)	0.381	0.252	0.403	0.402	0	0 (0.495)
Median(ARI)	0.329	0.419	0.646	**0.723**	0	0 (0.674)
Median(Time)	2.755	17.805	0.020	0.001	2.410	0.490

Table 7.2: Tables of summary statistics pertaining to the tested methods, where the data set is generated by models (D2). For example, the top table is constructed from data sets with $n_g = 50$. For median(AM) and median(ARI), the bracketed numbers for DEMP+ methods are the corresponding median values among replications that identified more than one cluster. The highest ARI values in each table are bolded.

(D3), $n_g = 50$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	5	5	4	4	1	1
Range(G)	[4, 8]	[4, 8]	[4, 6]	[4, 6]	[1, 3]	[1, 4]
Median(AM)	1.670	1.912	2.633	2.615	0 (1.318)	0 (2.356)
Median(ARI)	0.638	0.667	**0.847**	**0.847**	0 (0.669)	0 (0.824)
Median(Time)	0.840	69.275	0.010	0.010	2.420	2.080

(D3), $n_g = 100$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	6	6	4	4	1	1
Range(G)	[4, 8]	[4, 8]	[4, 5]	[4, 6]	[1, 5]	[1, 4]
Median(AM)	1.388	1.346	2.669	2.669	0 (1.337)	0 (0.924)
Median(ARI)	0.636	0.627	**0.849**	**0.849**	0 (0.679)	0 (0.581)
Median(Time)	1.790	122.065	0.015	0.020	3.910	3.520

(D3), $n_g = 150$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	6	6	4	4	1	1
Range(G)	[5, 8]	[4, 8]	[4, 5]	[4, 6]	[1, 5]	[1, 5]
Median(AM)	1.294	1.270	2.641	2.632	0 (0.564)	0 (0.674)
Median(ARI)	0.632	0.624	**0.848**	**0.848**	0 (0.497)	0 (0.458)
Median(Time)	2.72	171.31	0.020	0.020	3.980	3.590

(D3), $n_g = 200$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	6	6	4	4	1	1
Range(G)	[5, 8]	[5, 8]	[4, 5]	[4, 7]	[1, 6]	[1, 5]
Median(AM)	1.338	1.236	2.656	2.645	0 (0.582)	0 (0.705)
Median(ARI)	0.634	0.627	**0.849**	**0.849**	0 (0.552)	0 (0.475)
Median(Time)	3.75	225.71	0.02	0.02	3.94	3.61

Table 7.3: Tables of summary statistics pertaining to the tested methods, where the data set is generated by models (D3). For example, the top table is constructed from data sets with $n_g = 50$. For median(AM) and median(ARI), the bracketed numbers for DEMP+ methods are the corresponding median values among replications that identified more than one cluster. The highest ARI values in each table are bolded.

(D4), $n_g = 50$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	4	2	2	2	1	2
Range(G)	[3, 7]	[2, 4]	[2, 4]	[2, 3]	[1, 4]	[1, 3]
Median(AM)	0.369	0.830	0.787	0.930	0 (0.492)	0.818 (0.879)
Median(ARI)	0.484	0.894	0.920	**0.973**	0 (0.571)	0.894 (0.92)
Median(Time)	1.410	2.665	0.020	0.010	1.280	0.140

(D4), $n_g = 100$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	4	2	2	2	1	2
Range(G)	[4, 6]	[2, 4]	[2, 4]	[2, 3]	[1, 1]	[1, 3]
Median(AM)	0.324	0.815	0.763	0.893	0	0.838
Median(ARI)	0.440	0.940	0.857	**0.973**	0	0.940
Median(Time)	2.400	6.000	0.010	0.001	1.330	0.140

(D4), $n_g = 150$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	4	3	2	2	1	1.5
Range(G)	[4, 7]	[2, 4]	[2, 4]	[2, 2]	[1, 4]	[1, 3]
Median(AM)	0.313	0.383	0.663	0.887	0	0.121
Median(ARI)	0.433	0.571	0.832	**0.991**	0	0.254
Median(Time)	3.305	12.940	0.010	0.001	1.340	0.500

(D4), $n_g = 200$	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	4	3	2	2	1	2
Range(G)	[4, 8]	[2, 3]	[2, 4]	[2, 2]	[1, 1]	[1, 3]
Median(AM)	0.318	0.383	0.689	0.901	0	0.357 (0.871)
Median(ARI)	0.449	0.571	0.824	**0.983**	0	0.558 (0.96)
Median(Time)	4.070	25.830	0.020	0.001	1.335	0.300

Table 7.4: Tables of summary statistics pertaining to the tested methods, where the data set is generated by models (D4). For example, the top table is constructed from data sets with $n_g = 50$. For median(AM) and median(ARI), the bracketed numbers for DEMP+ methods are the corresponding median values among replications that identified more than one cluster. The highest ARI values in each table are bolded.

Tables 7.1 to 7.4 summarize the experiment for (D1), (D2), (D3) and (D4) respectively. In general, all merging methods detected a decreased number of clusters as expected. However, at the threshold of 0.1, DEMP+ methods reduced all components into one cluster frequently, as indicated by the noticeable difference between the median values inside and outside the parentheses. Nonetheless, when the instances that identified only one cluster was excluded in median calculation, DEMP+ methods exhibited comparative levels of additive margin and ARI. This suggests cutoff threshold selection for DEMP+ is critical for its performance. To our knowledge, there is no specialized threshold tuning strategy for DEMP+, so a grid-based search would be the best option. However, this process could be slow, depending on the granularity of the search grid. Among all four, (D3) benefitted the most from the merging methods. In particular, G-Mode and t-mode methods led to a marked level of improvement in additive margin as well as ARI, where the median ARI values were in mid-to-high 80%'s across all sample sizes. In (D4), the t-based methods continued to produce better quality clusters, as indicated by AM and ARI. Overall, the mode merging methods exhibited low computation time compared to other methods. This experiment indicates that the mode merging method can be an effective tool for identifying clusters from mixture components at a relatively cheap computational cost.

Real Data: Old Faithful

The Old Faithful data set (Härdle et al., 1991) is an excellent example where a finite mixture model may identify more components than the number of perceived clusters. In figure 7.2, one can identify two most notable clusters; one on the bottom-left corner and the other on the top-right corner. The contour plots indicate that each cluster is unimodal, which makes this data set bimodal. However, for example, when a GMM is fitted, the model often identifies 3 components instead of 2. For this data set, the range of components fitted is $G = 1, 2, \ldots, 8$.

Table 7.5 shows the key summaries pertaining to the tested methods. The median number of fitted components is 3 for both GMM and tEIGEN, whereas median is 2 for the merging methods. However, the mode merging methods (G-Mode and t-Mode) was more

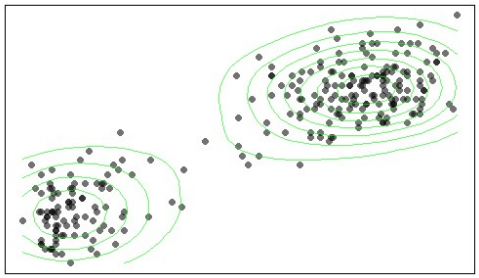

Figure 7.2: Scatterplot of the Old Faithful data set with contour lines. Horizontal axis measures the duration of eruptions and the vertical axis measures the waiting time between eruptions. Two clusters are easily noted; one on the bottom left corner and the other on the top right corner.

	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	2	2	2	2
Range(G)	[2, 3]	[2, 3]	[2, 2]	[2, 2]	[1, 3]	[1, 3]
Median(AM)	1.233	1.235	2.185	2.186	1.193	1.183
Median(Time)	1.75	69.69	0.01	0.01	0.47	0.46

Table 7.5: Table of summary statistics for all tested methods. Each column corresponds to a tested method. Starting from the second row from the top, the rows contain median and the range of number of fitted components, additive margins, and elapsed times (in seconds).

consistent than the DEMP+ methods in terms of the range. In addition, the mode merging methods showed a significant improvement in additive margin, whereas the DEMP+ methods showed a slight decrease from that of the initially-fitted mixture models. However, this is due to the instances where the DEMP+ methods estimated 1 cluster, which corresponds to the additive margin value of 0. Indeed, the median additive margin of G-DEMP+ and t-DEMP+ after removing the instances that estimated 1 cluster are 2.084 and 2.106 respectively. These adjusted median values are closer to that of mode merging methods. It's also interesting to see that tEIGEN's median additive margin is slightly higher than that of GMM, and t-Mode continues to have a slight advantage over G-Mode in terms of additive margin. This is somewhat expected because the quality of merged components depends on the initial set of fitted components. Another appeal of mode merging methods is the shorter

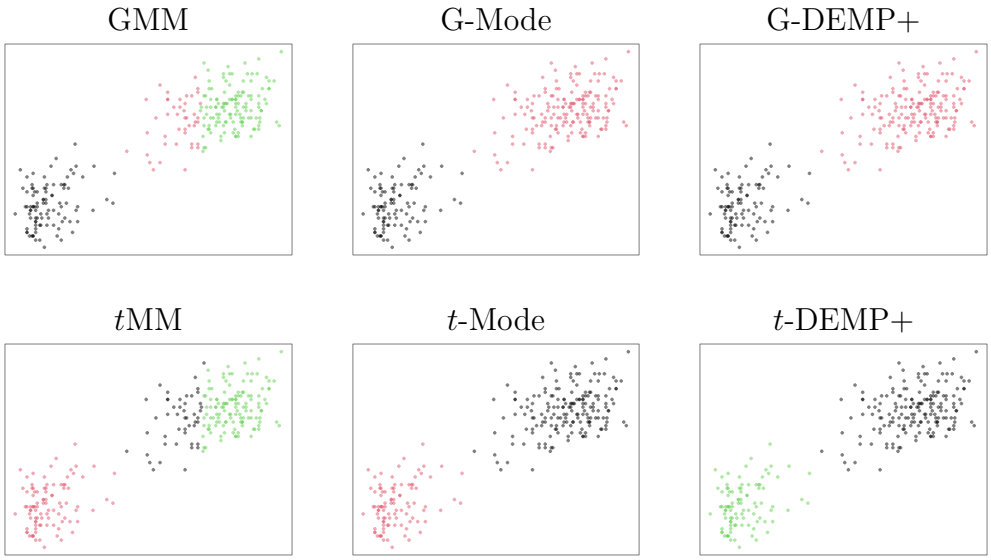

Figure 7.3: Scatterplot examples of the Old Faithful, colour-coded by estimated clusters from each tested method. The top and bottom rows correspond to GMM and tEIGEN-based methods. Reading from left to right, the first column is (GMM, tEIGEN), the second column is (G-Mode, t-Mode) and the third column is (G-DEMP+, t-DEMP+).

computation time. The equally low median elapsed time for G-Mode and t-Mode indicates that mode merging can be executed at a negligible computational cost in comparison to the initial model-fitting process. Figure 7.3 shows examples of the components identified by each tested method. The top row shows the fitted components from GMM (left) and tEIGEN (right), and both methods have fitted three. In particular, we see that the top right cluster is estimated by two components. Mode merging methods have merged the top two components in all replications, resulting in the two notable clusters as discussed earlier. The DEMP+ methods have identified the two notable clusters as well, but their results are more varied. Overall, this experiment shows that mode merging for tMM can be a computationally cheap component merging technique that inherits the advantages that tMM has over GMM.

	GMM	tEIGEN	G-Mode	t-Mode	G-DEMP+	t-DEMP+
Median(G)	3	3	3	3	1	1
Range(G)	[2, 7]	[2, 5]	[2, 7]	[2, 5]	[1, 4]	[1, 4]
Median(AM)	0.365	0.336	0.369	0.340	0 (0.376)	0 (0.348)
Median(ARI)	0.782	0.787	0.782	0.787	0 (0.803)	0 (0.801)
Median(Time)	2.42	16.35	0.001	0.001	0.58	0.52
Median(Dist)	4.082	3.732	4.082	3.732	0 (4.082)	0 (3.900)

Table 7.6: Table of summary statistics for all tested methods. Each column corresponds to a tested method. Starting from the second row from the top, the rows contain median and the range of number of fitted components, additive margins, and elapsed times (in seconds) and the median value of average distance between each pair of mixture component means denoted by Median(Dist). For Median(AM), Median(ARI) and Median(Dist), the bracketed numbers for DEMP+ methods are the corresponding median values among replications that identified more than one cluster.

Chronic Kidney Disease

In this section, we cluster the Chronic Kidney Disease data set from Dua and Graff (2017), where a cleaned version is available in the R package *teigen* (Andrews et al., 2018). Clustering various measurements taken from the patients and investigating the relationship between the identified clusters and the patients' disease status can help with the study of the disease's characteristics. When the disease status is binarily classified, fewer clusters (ideally two, perhaps) could be favoured over more clusters. Therefore, if a finite mixture model fits a large number of components, then component merging could be beneficial.

Table 7.6 shows that the merging method did not result in a significant change in AM or ARI. This is likely because each cluster is too far away from each other, as shown in the Medium(Dist) row. Indeed, if the component-wise modes are far apart (for GMM and tMM, component-wise mean and mode are equal), then, they are unlikely to merge to one mode. In addition, far-apart modes would result in corresponding densities having a low degree of overlap. All of this would imply that the mode merging and DEMP+ methods would not change the cluster assignments significantly. This is somewhat expected given that Median(Dist) value for GMM and tEIGEN are both large; The performance of the

tested merging methods dependent on the initially-fitted model.

7.4 Discussion

In this chapter, the t mean-shift, a novel mode merging method based on the EM algorithm for the tMM and its parsimonious variants, is introduced. The update equation is closely related to the Gaussian ridgeline function, which contains all critical points of the Gaussian finite mixture density function, including all modes. The performance of the method was demonstrated using both simulated and real data. In the numerical experiments, the introduced method was shown to identify mixture components whose modes are near each other and merge them into a single cluster, while maintaining a low computational cost. Directions for future work include extensions to other non-Gaussian finite mixtures, and a further analysis on the mean-shift and non-Gaussian ridgeline functions.

Chapter 8

StableMerge: A Generalized Mode Merging Framework

8.1 Introduction

Model-based clustering deploys a finite mixture model, whose probability mass or density function is a convex combination of probability mass or density functions usually from the same parametric family, to estimate clusters within a data set. Each observation is allocated to a single summand density function (also known as a component density) via component-wise membership probability, so the set of observations associated with each component is commonly interpreted as a cluster (McLachlan and Peel, 2004). The consequent component-cluster correspondence naturally demands a high degree of flexibility from component distributions, as the shape of each cluster, whose definition itself is context-dependent, is likely unbeknownst to the investigator a priori. To address this concern, numerous robust finite mixtures have been developed, including Peel and McLachlan (2000); Franczak et al. (2013); Lee and McLachlan (2013); Browne and McNicholas (2015); Dang et al. (2015); Punzo and McNicholas (2016). Comprehensive overviews on robust model-based clustering can be found in McLachlan and Peel (2004); McNicholas (2016).

Another flavour of cluster identification from finite mixtures is modal clustering. A suc-

cinct description of the key issue is given in Hennig (2010). Modal clustering is appropriate when the component-cluster correspondence assumption no longer holds. Such a relaxation is particularly relevant when the mixture model comprises of rigid component densities. For example, the Gaussian finite mixture is arguably the most popular one, but its component distribution, the Gaussian distribution, is unable to capture non-standard tail behaviours or any sort of skewness. A common consequence of this deficit in flexibility is the over-estimation of the number of mixture components, whereby multiple components are fitted to a single cluster. A bundle of such components can result in a collectively unimodal region, which modal clustering attempts to seek out. Another benefit of modal clustering is that a single mode-based cluster may assume a shape beyond the limit of a single component, thereby alleviating the burden of flexibility imposed on individual components. Recent developments on modal clustering include Carreira-Perpinan (2000); Comaniciu and Meer (2002); Yuan et al. (2010); Carreira-Perpinan (2007); Chacón (2019); Melnykov (2016); Kim and Browne (2021a). For an overview on modal clustering, refer to Menardi (2016); Chacón (2020). The mean-shift algorithm by Comaniciu and Meer (2002) is a gradient-based method of mode detection, where an initial point is iteratively updated in the direction of increasing density, until the nearest local mode is reached. It is intuitively straight-forward, and is easy to apply in many situations. Moreover, Carreira-Perpinan (2007) showed that the mean-shift algorithm for the Gaussian finite mixture model is an Expectation-Maximization (EM) algorithm (EM algorithm credited to Dempster et al. (1977)), which adds to the method's appeal. Chacón (2019) applied the mean-shift algorithm on a Gaussian finite mixture for modal clustering, and Kim and Browne (2021a) introduced a mean-shift-based mode merging algorithm for a finite mixture of multivariate t-distributions, and showed that the t-mean-shift is also an EM algorithm. However, to our best knowledge, a mode-merging algorithm is yet to be developed for other non-Gaussian finite mixtures besides t, and as listed earlier, there is a large body literature on various non-Gaussian finite mixture models. As they are not immune to the violation of component-cluster correspondence assumption, we develop a mean-shift-based mode merging algorithm for three classes of highly flexible non-Gaussian components: power-exponential, normal variance mixture, and normal variance-mean mixture. For each class,

we begin by introducing its general representation, and develop the mean-shift algorithm for its finite mixture. Then we present a unified merging rule for all three classes. Thereafter, we demonstrate the algorithms in simulated and real-data settings. We will then conclude with a brief discussion on our work and future directions.

8.2 Methodology

In this section, we generalize the mean-shift-based mode-merging algorithms to three broad classes of component distributions: power exponential (Gómez et al., 1998), normal variance mixture and normal variance-mean mixture (McNeil et al., 2015).

The power exponential (PE) distribution, also known as the generalized Gaussian distribution, extends the Gaussian distribution to allow leptokurtosis (heavier tails) and platykurtosis (lighter tails). A p-dimensional PE distribution is parametrized by a location vector $\boldsymbol{\mu} \in \mathbb{R}^p$, a $(p \times p)$-dimensional positive definite scale matrix $\boldsymbol{\Sigma}$ and a positive shape parameter $\beta > 0$. Per Gómez et al. (1998), its density function is given by

$$f_{PE}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \beta) = \frac{p\Gamma(p/2)}{\Gamma\left(1 + \frac{p}{2\beta}\right) 2^{1+\frac{p}{2\beta}}} |\pi\boldsymbol{\Sigma}|^{-1/2} \exp\left\{-\frac{1}{2}\left[(\boldsymbol{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]^{\beta}\right\}. \quad (8.1)$$

Special cases of the PE distribution include symmetric Laplace at $\beta = 1/2$ and Gaussian at $\beta = 1$, and its asymptotic distribution, as $\beta \to \infty$, is the uniform distribution between 0 and 1. Moreover, when $\beta \leq 1$, then it is a normal variance mixture distribution (also known as a scale mixture of normal distributions) (Gómez-Sánchez-Manzano et al., 2008). A finite mixture of power exponential distributions was studied by Zhang and Liang (2010); Dang et al. (2015).

Another class of generalization on tailedness is the normal variance mixture distribution. Let $W > 0$ be a (absolutely) continuous random variable with parameter set denoted by Ω, $\boldsymbol{\mu} \in \mathbb{R}^p$ a location vector, and $\boldsymbol{U}$ a p-dimensional Gaussian random vector with zero mean and covariance $\boldsymbol{\Sigma}$. Then, a p-dimensional random vector $\boldsymbol{X}$ following a normal variance

mixture (NVM) distribution with admits the following stochastic representation

$$\boldsymbol{X} = \boldsymbol{\mu} + \sqrt{W}\boldsymbol{U}. \tag{8.2}$$

In a G-component mixture setting, the NVM distribution yields the following hierarchy. Let $\boldsymbol{Z} = (Z_1, \ldots, Z_G)'$ denote indicator variables so that $Z_g = 1$ if $\boldsymbol{X}$ belongs to component g, and let W_g denote the latent NVM variable with distribution conditional on $Z_g = 1$ being H_g, parametrized by $\boldsymbol{\Omega}_g$. Then,

$$\begin{aligned} \boldsymbol{X}|Z_g = 1 &\sim NVM_p(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Omega}_g), \\ W_g|Z_g = 1 &\sim H_g(\boldsymbol{\Omega}_g), \\ \boldsymbol{X}|Z_g = 1, w_g &\sim N_p(\boldsymbol{\mu}_g, w_g\boldsymbol{\Sigma}_g). \end{aligned} \tag{8.3}$$

$$\tag{8.4}$$

This latent variable hierarchy makes NVM distributions well-suited for EM algorithms. The third, and perhaps the most flexible, generalization on the Gaussian distribution is the normal variance-mean mixture (NVMM) distribution. A NVM distribution accommodates non-Gaussian tail behaviours, but is still intolerant of skewness. The NVMM generalizes the NVM further by incorporating locational asymmetry. Let $\boldsymbol{U}$ follow a p-dimensional Gaussian distribution with zero mean and covariance $\boldsymbol{\Sigma}$ and $W > 0$ a positive random variable, independent from $\boldsymbol{U}$, following a distribution h with parameters denoted by $\boldsymbol{\Omega}$. Then, a p-dimensional NVMM random vector $\boldsymbol{X}$ with location parameter $\boldsymbol{\mu}$ and skewness parameter $\boldsymbol{\alpha}$ admits the following stochastic representation

$$\boldsymbol{X} = \boldsymbol{\mu} + W\boldsymbol{\alpha} + \sqrt{W}\boldsymbol{U}. \tag{8.5}$$

Similar to the NVM, in a G-component mixture setting, the NVMM distribution yields

the following hierarchy. Using an analogous notation as that from the NVM, we have

$$\begin{aligned}\boldsymbol{X}|Z_g = 1 &\sim NVMM_p(\boldsymbol{\mu}_g, \boldsymbol{\alpha}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Omega}_g),\\ \boldsymbol{X}|Z_g = 1, w_g &\sim N_p(\boldsymbol{\mu}_g + w_g\boldsymbol{\alpha}_g, w_g\boldsymbol{\Sigma}_g),\\ W_g|Z_g = 1 &\sim H_g(\boldsymbol{\Omega}_g).\end{aligned} \tag{8.6}$$

In our work, we will assume that H_g has a density function denoted by h_g. We develop a mean-shift algorithm for the finite mixture of EP, NVM and NVMM distributions respectively, thereby enabling mode identification on a truly broad range of distributions. Given a G-component finite mixture density f, we want to estimate its local maximizer $\boldsymbol{x}^\star$ via an EM algorithm. That is, $f(\boldsymbol{x})$ is subject to maximization with respect to $\boldsymbol{x}$. We assume that all parameters are known a priori. In practice, this means that their estimates would be computed before mean-shift. The form of $f(\boldsymbol{x})$ is often challenging to tackle directly, so we adopt the complete-data framework to apply the EM algorithm. The resultant complete-data tuple is $(\boldsymbol{x}', \boldsymbol{z}')' = (\boldsymbol{x}', z_1, \ldots, z_G)'$ for the PEMM, and $(\boldsymbol{x}', \boldsymbol{w}', \boldsymbol{z}')' = (\boldsymbol{x}', w_1, \ldots, w_G, z_1, \ldots, z_G)'$ for the NVM-MM and NVMM-MM.

8.2.1 Power Exponential Mean-shift

In a G-component PE mixture model (PEMM), the gth component's density function follows a PE distribution with component-wise parameter sets $\boldsymbol{\Theta}_g = \{\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \beta_g\}$ for $g = 1, \ldots, G$. That is,

$$f_{PE}(\boldsymbol{x}; \boldsymbol{\Theta}_g) = \frac{p\Gamma(p/2)}{\Gamma\left(1 + \frac{p}{2\beta_g}\right) 2^{1+\frac{p}{2\beta_g}}} |\pi\boldsymbol{\Sigma}_g|^{-1/2} \exp\left\{-\frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_g)\right]^{\beta_g}\right\}. \tag{8.7}$$

The PEMM's complete-data log-density is

$$\log f(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\Theta}) = \sum_{g=1}^{G} z_g \left\{-\frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{\mu}_g)\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_g)\right]^{\beta_g}\right\} + \text{const},$$

where $\boldsymbol{\Theta}$ denotes the set of all model parameters, and 'const' is an additive constant with respect to $\boldsymbol{x}$. Consequently, the conditional expectation given the mode estimate at iteration t is

$$Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}, \boldsymbol{\Theta}) = \mathrm{E}\left[\log f(\boldsymbol{x}, \boldsymbol{Z})|\boldsymbol{x}^{(t)}\right] = \sum_{g=1}^{G} -\frac{z_g^{(t)}}{2}\left[(\boldsymbol{x}-\boldsymbol{\mu}_g)\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)\right]^{\beta_g} + \text{const},$$

where $z_g^{(t)}$ is the conditional expectation of z_g at iteration t. Algebraically,

$$z_g^{(t)} = \frac{\pi_g f_g(\boldsymbol{x}^{(t)}; \boldsymbol{\Theta}_g)}{\sum_{k=1}^{G} \pi_k f_{PE}(\boldsymbol{x}^{(t)}; \boldsymbol{\Theta}_k)}. \tag{8.8}$$

Since $\boldsymbol{x}$ cannot be separated to form an explicit solution, an implicit optimization scheme is used. Consider the gradient and Hessian of $Q(\boldsymbol{x}|\boldsymbol{x}^{(t)})$

$$\nabla_{\boldsymbol{x}} Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = \sum_{g=1}^{G}(-z_g^{(t)}\beta_g)\left[(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)\right]^{\beta_g-1}\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g) \tag{8.9}$$

$$\begin{aligned}\nabla_{\boldsymbol{x}}^2 Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = \sum_{g=1}^{G}(-z_g^{(t)}\beta_g)\Big\{ &\left[(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)\right]^{\beta_g-1}\boldsymbol{\Sigma}_g^{-1} \\ &+ 2(\beta_g-1)\left[(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)\right]^{\beta_g-2}\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_g)(\boldsymbol{x}-\boldsymbol{\mu}_g)'\boldsymbol{\Sigma}_g^{-1}\Big\}.\end{aligned} \tag{8.10}$$

Each β_g must be at least 1 for its gradient to exist (and at least 2 for the Hessian), as that ensures continuity over all $\boldsymbol{x}$. Indeed, the component-wise log-likelihood is continuously differentiable only $\lfloor\beta_g\rfloor$ times. Thus, $Q(\boldsymbol{x}|\boldsymbol{x}^{(t)})$ has $\min\{\lfloor\beta_1\rfloor, \ldots, \lfloor\beta_G\rfloor\}$ continuous derivatives with respect to $\boldsymbol{x}$, implying that an optimization scheme requiring the dth derivative is appropriate only when $d \geq \min\{\lfloor\beta_1\rfloor, \ldots, \lfloor\beta_G\rfloor\}$. In general, a derivative-free method should be used. In this work, we use the Nelder-Mead method implemented R software's *optim* function (R Core Team, 2020).

8.2.2 Normal Variance Mixture Mean-shift

In a G-component NVM mixture model (NVM-MM), the component-wise density functions $f_{NVM}(\boldsymbol{x}; \boldsymbol{\Theta}_g)$ follow a NVM distribution parametrized by $\boldsymbol{\Theta}_g = \{\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Omega}_g\}$. Examples include t (McNeil et al., 2015), symmetric Laplace (Kotz et al., 2012) and symmetric generalized hyperbolic (McNeil et al., 2015) distributions. The general form of the NVM-MM complete-data log-density is

$$\begin{aligned}\log f(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}; \boldsymbol{\Theta}) &= \sum_{g=1}^{G} z_g \log f_g(\boldsymbol{x}, w_g; \boldsymbol{\Theta}_g) \\ &= \sum_{g=1}^{G} z_g \left[\log \phi(\boldsymbol{x}|w_g; \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g) + \log h(w_g; \boldsymbol{\Omega}_g)\right],\end{aligned}$$

where $\phi_g(\boldsymbol{x}|w_g)$ denotes the conditional Gaussian density of $\boldsymbol{x}$ given w_g, and $h_g(w_g)$ denotes the marginal density of w_g, both under gth component membership. Its conditional expectation is

$$Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = \sum_{g=1}^{G} -\frac{z_g^{(t)} \mathrm{E}[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g]}{2} (\boldsymbol{x} - \boldsymbol{\mu}_g)' \boldsymbol{\Sigma}_g^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_g) + \text{const},$$

where $z_g^{(t)}$ takes a similar form to (8.8) but with f_{NVM} instead of f_{PE}. Since $\log f(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}; \boldsymbol{\Theta})$ interacts with $\boldsymbol{x}$ only through the conditional Gaussian density ϕ, solving for $\boldsymbol{x}$ in the critical point equation $\nabla_{\boldsymbol{x}} Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = 0$ yields

$$\boldsymbol{x}^{(t+1)} = \left[\sum_{g=1}^{G} z_g^{(t)} \mathrm{E}\left[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g\right] \boldsymbol{\Sigma}_g^{-1}\right]^{-1} \sum_{g=1}^{G} z_g^{(t)} \mathrm{E}\left[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g\right] \boldsymbol{\Sigma}_g^{-1} \boldsymbol{\mu}_g. \tag{8.11}$$

The above iteration is repeated until a convergence criterion is met.

8.2.3 Normal Variance-mean Mixture Mean-shift

In a G-component NVMM mixture model (NVMM-MM), the component-wise density functions $f_{NVMM}(\boldsymbol{x})$ follow a NVMM distribution parametrized by $\boldsymbol{\Theta}_g = \{\boldsymbol{\mu}_g, \boldsymbol{\alpha}_g, \boldsymbol{\Sigma}_g, \boldsymbol{\Omega}_g\}$. Examples include skew-$t$ (Barndorff-Nielsen and Shephard, 2001), (shifted) asymmetric Laplace (Kotz et al., 2012; Franczak et al., 2013) and generalized hyperbolic (Browne and McNicholas, 2015) distributions. Like in the NVM case, the complete-data for a NVMM-MM extends to $(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$. The complete-data log-density function is

$$\log f(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}; \boldsymbol{\Theta}) = \sum_{g=1}^{G} z_g \left[\log \phi_g(\boldsymbol{x}|w_g; \boldsymbol{\mu}_g, \boldsymbol{\alpha}_g, \boldsymbol{\Sigma}_g) + \log h_g(w_g; \boldsymbol{\Omega}_g)\right],$$

where the gth component's conditional Gaussian density of $\boldsymbol{x}$ is governed by the mean vector $\boldsymbol{\mu}_g + w_g\boldsymbol{\alpha}_g$ and covariance matrix $w_g\boldsymbol{\Sigma}_g$. The conditional expectation of $\log f(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}; \boldsymbol{\Theta})$ is

$$Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = \sum_{g=1}^{G} -z_g^{(t)} \left\{ \frac{-\mathrm{E}[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g]}{2} (\boldsymbol{x} - \boldsymbol{\mu}_g)' \boldsymbol{\Sigma}_g^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_g) + \boldsymbol{\alpha}_g' \boldsymbol{\Sigma}_g^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_g) \right\} + \text{const},$$

where $z_g^{(t)}$ take a similar form as (8.8) but with f_{NVMM} instead. Solving for $\boldsymbol{x}$ in the critical point equation $\nabla_{\boldsymbol{x}} Q(\boldsymbol{x}|\boldsymbol{x}^{(t)}) = 0$ yields

$$\begin{aligned} \boldsymbol{x}^{(t+1)} = & \left[\sum_{g=1}^{G} z_g^{(t)} \mathrm{E}\left[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g\right] \boldsymbol{\Sigma}_g^{-1} \right]^{-1} \\ & \times \sum_{g=1}^{G} z_g^{(t)} \boldsymbol{\Sigma}_g^{-1} \left(\mathrm{E}\left[W_g^{-1}|Z_g = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_g\right] \boldsymbol{\mu}_g + \boldsymbol{\alpha}_g \right). \end{aligned} \tag{8.12}$$

The above iteration is repeated until a convergence criterion is met.

8.2.4 Monotonicity with Respect to Log-density

Carreira-Perpinan (2007) showed that the Gaussian mean-shift is an EM algorithm, thus the sequence of observed-data log-density $\{\log f(\boldsymbol{x}^{(0)}; \boldsymbol{\Theta}), \log f(\boldsymbol{x}^{(1)}; \boldsymbol{\Theta}), \ldots\}$ evaluated over the mean-shift solutions is monotonically increasing. We show that our non-Gaussian mean-shift algorithms preserves the said monotonicity.

Proposition 8. *Let $\{\boldsymbol{x}^{(t)}\}_{t=1,\ldots}$ denote a sequence of solutions computed from the mean-shift algorithm for PEMM, NVM-MM or NVMM-MM. Then, the corresponding observed-data log-density sequence $\{\log f(\boldsymbol{x}^{(t)}; \boldsymbol{\Theta})\}_{t=1,\ldots}$ is monotonically increasing.*

Proof. For notational brevity, let $f(\boldsymbol{x})$, $f(\boldsymbol{x}, w)$ and $h(w|\boldsymbol{x})$ denote the observed-data marginal, complete-data joint and latent-data conditional densities under a fixed set of parameters. Then, $\log f(\boldsymbol{x})$ can be decomposed as

$$\log f(\boldsymbol{x}) = \log f(\boldsymbol{x}, w) - \log h(w|\boldsymbol{x}).$$

Taking the expectation with respect to the latent data conditional on a different value $\hat{\boldsymbol{x}}$, we have

$$\log f(\boldsymbol{x}) = Q(\boldsymbol{x}|\hat{\boldsymbol{x}}) - \mathrm{E}[\log h(w|\boldsymbol{x})|\hat{\boldsymbol{x}}],$$

where $Q(\boldsymbol{x}|\hat{\boldsymbol{x}}) = \mathrm{E}[\log f(\boldsymbol{x}, w)|\hat{\boldsymbol{x}}]$. Hence, the difference between $\log f(\boldsymbol{x})$ and $\log f(\hat{\boldsymbol{x}})$ is

$$\log f(\boldsymbol{x}) - \log f(\hat{\boldsymbol{x}}) = Q(\boldsymbol{x}|\hat{\boldsymbol{x}}) - Q(\hat{\boldsymbol{x}}|\hat{\boldsymbol{x}}) - \mathrm{E}[\log h(w|\boldsymbol{x})|\hat{\boldsymbol{x}}] + \mathrm{E}[\log h(w|\hat{\boldsymbol{x}})|\hat{\boldsymbol{x}}].$$

By Jensen's inequality (Jensen, 1906), we know that $\mathrm{E}[\log h(w|\hat{\boldsymbol{x}})|\hat{\boldsymbol{x}}] \geq \mathrm{E}[\log h(w|\boldsymbol{x})|\hat{\boldsymbol{x}}]$, so we conclude

$$\log f(\boldsymbol{x}) - \log f(\hat{\boldsymbol{x}}) \geq Q(\boldsymbol{x}|\hat{\boldsymbol{x}}) - Q(\hat{\boldsymbol{x}}|\hat{\boldsymbol{x}}).$$

Hence, increasing $Q(\boldsymbol{x}|\hat{\boldsymbol{x}}) - Q(\hat{\boldsymbol{x}}|\hat{\boldsymbol{x}})$ translates to increasing $\log f(\boldsymbol{x}) - \log f(\hat{\boldsymbol{x}})$, which is the objective of the mean-shift algorithm. □

8.2.5 Threshold-free Component Merging

Based on the proposed mean-shift algorithms, we develop a procedure for detecting unimodal clusters via merging of component-wise modes. A high-level description of an existing method is as follows. Given a G-component mixture model and the modes of each component $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_G$, the corresponding mean-shift algorithm is applied to each $\boldsymbol{m}_g$ $(g = 1, \ldots, G)$ to obtain G local estimates of the mixture modes $\hat{\boldsymbol{m}}_1, \ldots, \hat{\boldsymbol{m}}_G$. Then, the labels of components whose mode estimates are close enough are combined into a single label. For instance, if $||\hat{\boldsymbol{m}}_1 - \hat{\boldsymbol{m}}_2||_2 < c$ for a pre-determined threshold $c > 0$, then the observations belonging to components 1 and 2 would be grouped into a single cluster than their label would be unified to 1. This procedure was initially applied to the GMM by Chacón (2019), and to the tMM by Kim and Browne (2021a).

The outcome of the aforementioned merging procedure is heavily dependent on the closeness threshold c. Specifically, higher values of c yields fewer clusters. Yet, a selection rule for this crucial constant is not apparent. Thus, we introduce a robust threshold-free merging procedure. Let

$$d_{gl} = ||\hat{\boldsymbol{m}}_g - \hat{\boldsymbol{m}}_l||_2$$

denote the Euclidean distance between the estimated mixture modes generated from components g and l, for $g < l$ and $g, l = 1, \ldots, G$. Then, there are $G^\star = \binom{G}{2} + 1$ distance values if we include a zero. We build a tree for a bottom-up hierarchical clustering (we use complete-linkage in this work), where two groups of observations $\mathcal{S}_i$ and $\mathcal{S}_j$ are combined into one if

$$d_{ij}^{\max} = \max\{||\boldsymbol{x} - \boldsymbol{y}||_2 : \boldsymbol{x} \in \mathcal{S}_i, \boldsymbol{y} \in \mathcal{S}_j\}$$

is the smallest among all pairs of observation groups $\mathcal{S}_1, \ldots, \mathcal{S}_K$. If groups i and j are combined, the distance between them is thus $d_{ij}^{\max}$. In our problem, the initial observation sets are $\mathcal{S}_g = \{\hat{\boldsymbol{m}}_g\}$ for $g = 1, \ldots, G$. Now consider the ordered pairwise distances of the mode estimates:

$$\{0\} \cup \{d_{gl}\}_{g<l=1,\ldots,G} \quad \longrightarrow \quad 0 = d_{(0)} \leq d_{(1)} \leq d_{(2)} \leq \cdots \leq d_{(G^\star)}.$$

Then we can see that, under the complete-linkage procedure, the maximum possible distance between groups is $d_{(G^\star)}$. If the tree is 'cut' at some $c \in [0, d_{(G^\star)}]$, then group pairs whose distance is greater than c are separated, forming different clusters. Consider the following example. Suppose there are four points in $\mathbb{R}^2$: $\boldsymbol{x}_1 = (0,0)'$, $\boldsymbol{x}_2 = (0,0.1)'$, $\boldsymbol{x}_3 = (5,0)'$, $\boldsymbol{x}_4 = (4,0.1)'$. There are six Euclidean distances between the four points:

$$||\boldsymbol{x}_1 - \boldsymbol{x}_2|| = 0.1, \quad ||\boldsymbol{x}_1 - \boldsymbol{x}_3|| = 5, \quad ||\boldsymbol{x}_1 - \boldsymbol{x}_4|| = \sqrt{16.01}$$
$$||\boldsymbol{x}_2 - \boldsymbol{x}_3|| = \sqrt{25.01}, \quad ||\boldsymbol{x}_2 - \boldsymbol{x}_4|| = 4, \quad ||\boldsymbol{x}_3 - \boldsymbol{x}_4|| = \sqrt{1.01}$$

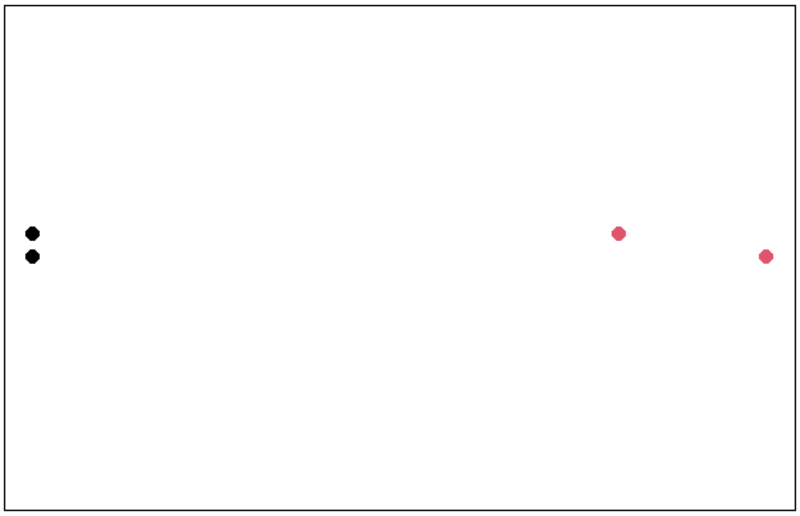

Figure 8.1: Coloured scatterplot of $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4$.

Based on the complete-linkage hierarchical clustering, the points are iteratively merged

in the following way

$$
\begin{aligned}
&\{\boldsymbol{x}_1\}, \{\boldsymbol{x}_2\}, \{\boldsymbol{x}_3\}, \{\boldsymbol{x}_4\} \\
&\longrightarrow \{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \{\boldsymbol{x}_3\}, \{\boldsymbol{x}_4\} \text{ smallest max distance is } 0.1, \text{ between } \boldsymbol{x}_1 \text{ and } \boldsymbol{x}_2. \\
&\longrightarrow \{\boldsymbol{x}_1, \boldsymbol{x}_2\}, \{\boldsymbol{x}_3, \boldsymbol{x}_4\} \text{ smallest max distance is } \sqrt{1.01}, \text{ between } \boldsymbol{x}_3 \text{ and } \boldsymbol{x}_4. \\
&\longrightarrow \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4\} \text{ smallest max distance is } 4, \text{ between } \boldsymbol{x}_2 \text{ and } \boldsymbol{x}_4.
\end{aligned}
$$

Although the tree has 3 key values (0.1, $\sqrt{1.01}$ and 4), it could be cut at any value between 0 and $\sqrt{25.01}$, resulting in different cluster counts. For example, if the tree is cut at an arbitrary value in the sub-interval $[0, 0.1)$, then 4 clusters would be identified. If cut in the sub-interval $[0.1, \sqrt{1.01})$, then 3 clusters would be identified. This observation suggests that the cluster count corresponding to the widest sub-interval could be seen as the most stable. We formalize this notion of cluster count stability as a definition.

Definition 1. *The stability of a cluster count k under complete-linkage hierarchical clustering is defined as*

$$
St(k) = \max_{a,b \in [0, d_{G^\star}]} \{b - a : \textit{cluster count if cut at } b = \textit{cluster count if cut at } a = K,\ b \geq a\}.
$$

The most stable cluster count is then defined as $k \in \{1, \ldots, G\}$ that maximizes $St(\cdot)$. In the case of a tie, the smallest cluster count is selected. However, in practical settings, identifying the boundary between two cluster counts can be challenging. Therefore, we approximate the boundaries by computing the cluster count at each $d_{(i)}$. For instance, in the 4-point example above, we have the following table of $d_{(i)}$ and corresponding cluster counts. The stability at cluster count k is approximated by calculating the difference between the smallest $d_{(i)}$ corresponding to k and the largest $d_{(i)}$ corresponding to $k+1$. In case of $k = 1$, $St(k)$ is approximated by computing the difference between the smallest and largest $d_{(i)}$ corresponding to $k = 1$.

$d_{(i)}$	0	0.1	$\sqrt{1.01}$	4	$\cdots$	$\sqrt{25.01}$
Cluster count	4	3	2	1	$\cdots$	1
$\hat{St}$	0.1	0.905	2.995	1.001	-	-

Table 8.1: Table of ordered pairwise distance between points and corresponding cluster counts and approximated stability.

The above example indicates that the most stable cluster count is 2, and this is consistent with the intuitive judgement, as the sets $\{\boldsymbol{x}_1, \boldsymbol{x}_2\}$ and $\{\boldsymbol{x}_3, \boldsymbol{x}_4\}$ are highly separated. The main benefit is that our stability-maximization procedure is a threshold-free way of estimating the number of clusters from a set of component-wise modes. The mean-shift and mode-merging procedures are outlined in algorithms 3 and 4 respectively.

Algorithm 3 PEMM/NVM-MM/NVMM-MM Mean-shift

initialize:
Set model parameters $\boldsymbol{\Theta}$.
Set convergence threshold $c > 0$.
Set convergence indicator: convergence($\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{\Theta}$, c).
for $g = 1, \ldots, G$ **do**
  $t = 0$
  $\boldsymbol{x}_g^{(t)} = \boldsymbol{\mu}_g$
  while converged = FALSE **do**
    $z_k^{(t)} \leftarrow \pi_k f_k(\boldsymbol{x}_g^{(t)}; \boldsymbol{\Theta}_k) / f(\boldsymbol{x}_g^{(t)}; \boldsymbol{\Theta})$
    if model is PEMM **then**
      $\boldsymbol{x}_g^{(t+1)} \leftarrow \underset{\boldsymbol{x}}{\operatorname{argmax}} \sum_{g=1}^{G} -\frac{z_g^{(t)}}{2} \left[(\boldsymbol{x} - \boldsymbol{\mu}_g)\boldsymbol{\Sigma}_g^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_g)\right]^{\beta_g}$
    else if model is NVM-MM **then**
      $w_k^{(t)} \leftarrow \mathrm{E}\left[W_k^{-1} | Z_k = 1, \boldsymbol{x}_g^{(t)}, \boldsymbol{\Theta}_k\right]$
      $\boldsymbol{x}_g^{(t+1)} \leftarrow \left[\sum_{k=1}^{G} z_k^{(t)} w_k^{(t)} \boldsymbol{\Sigma}_k^{-1}\right]^{-1} \sum_{k=1}^{G} z_k^{(t)} w_k^{(t)} \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k$
    else if model is NVMM-MM **then**
      $w_k^{(t)} \leftarrow \mathrm{E}\left[W_k^{-1} | Z_k = 1, \boldsymbol{x}_g^{(t)}, \boldsymbol{\Theta}_k\right]$
      $\boldsymbol{x}_g^{(t+1)} \leftarrow \left[\sum_{k=1}^{G} z_k^{(t)} w_k^{(t)} \boldsymbol{\Sigma}_k^{-1}\right]^{-1} \sum_{k=1}^{G} z_k^{(t)} \boldsymbol{\Sigma}_k^{-1} \left(w_k^{(t)} \boldsymbol{\mu}_k + \boldsymbol{\alpha}_k\right)$
    end if
    converged $\leftarrow$ convergence($\boldsymbol{x}_g^{(t+1)}$, $\boldsymbol{x}_g^{(t)}$, $\boldsymbol{\Theta}$, c)
    $t \leftarrow t + 1$
  end while
  $\hat{\boldsymbol{x}}_g \leftarrow \boldsymbol{x}_g^{(t)}$
end for
return $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_G$

Algorithm 4 Stability-based Mode Merging (StableMerge)

initialize:
Component-wise mode-label pairs $\{\hat{\boldsymbol{m}}_1, 1\}, \ldots, \{\hat{\boldsymbol{m}}_G, G\}$.
ClustStability $\leftarrow$ ().
Compute $d_{gl} = ||\hat{\boldsymbol{m}}_g - \hat{\boldsymbol{m}}_l||_2$ for $g < l$, $g, l = 1, \ldots, G$.
Sort in ascending order the set $\{0\} \cup \{d_{gl}\}$ to $0 = d_{(0)} \leq d_{(1)} \leq \cdots \leq d_{(G^\star)}$ where $G^\star = \binom{G}{2}$.
Construct a complete-linkage hierarchical tree using $\{\hat{\boldsymbol{m}}_1, \ldots, \hat{\boldsymbol{m}}_G\}$.
for $i = 0, \ldots, G^\star$ **do**
    Calculate the number of clusters produced when the tree is cut at $d_{(i)}$.
end for
return $\underset{k=1,\ldots,G}{\operatorname{argmax}}\ \hat{St}(k)$

8.3 Computational Aspects

In this section, we discuss three points of consideration regarding the implementation of the mean-shift and StableMerge procedures: mixture model parameter estimation, parsimonious variants and mode initialization.

Mixture Model Parameter Estimation

In most scenarios, the true parameters of the deployed mixture model is unavailable, and must be estimated instead. The parameter estimation process developed by Dang et al. (2015) is implemented as a R software package *mixSPE* (Browne et al., 2021). As to the NVM-MM and NVMM-MM, to our best knowledge, parameter estimation for their general forms is not yet available. However, under the NVM-MM framework, the R software package *teigen* (Andrews et al., 2018) fits the t mixture model, and under the NVMM-MM framework, the *MixGHD* R package (Tortora et al., 2021) fits the generalized hyperbolic mixture model.

Parsimonious Variants

Many multivariate parametric distributions involve a matrix-variate positive definite scale parameter $\boldsymbol{\Sigma}$, whose dimension increases quadratically against the data's dimension p. Therefore, one may consider parameter count reduction via structural constraints on $\boldsymbol{\Sigma}$. The eigen-decomposition and its variants are a popular choice, and are implemented in several finite mixture model packages in R, including *mclust* (Scrucca et al., 2016), *teigen* (Andrews et al., 2018) and *mixSPE* (Browne et al., 2021). Consider the eigen-decomposition of $\boldsymbol{\Sigma} = \boldsymbol{PDP}'$ where $\boldsymbol{P}$ is a $(p \times p)$-dimensional orthogonal matrix and $\boldsymbol{D} = \text{diag}(d_1, \ldots, d_p)$ is a diagonal matrix of eigenvalues arranged in a descending order. Fraley and Raftery (2002) introduced a scaled variant of the eigen-decomposition where the diagonal matrix $\boldsymbol{D}$ is decomposed into a product of a proportionality constant $\lambda > 0$ and a normalized diagonal matrix $\boldsymbol{D}^\star$ so that $\boldsymbol{D} = \lambda \boldsymbol{D}^\star$ such that $\det(\boldsymbol{D}^\star) = 1$. The resulting component-wise decomposition is

$$\boldsymbol{\Sigma}_g = \lambda_g \boldsymbol{P}_g \boldsymbol{D}_g^\star \boldsymbol{P}_g', \tag{8.13}$$

where λ_g represents the volume occupied by a mixture component, $\boldsymbol{P}_g$ its directional orientation, and $\boldsymbol{D}_g^\star$ its shape. The total number of free parameters across $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_G$ can be reduced by imposing cross-component equality constraints on λ_g, $\boldsymbol{P}_g$ or $\boldsymbol{D}_g^\star$.

The cross-component equality constraint on the orientation matrices, $\boldsymbol{P}_g = \boldsymbol{P}$, achieves the largest parameter count reduction and the most dramatic simplification of mean shift iteration formulae. For example, consider the constraint $\boldsymbol{P}_g = \boldsymbol{P}$ but with free λ_g and $\boldsymbol{D}_g$. The mean shift update $\boldsymbol{x}^{(t+1)}$ for the general NVM-MM and NVMM-MM are reduced to

$$\boldsymbol{x}_{NVM}^{(t+1)} = \boldsymbol{P} \left[\sum_{g=1}^{G} \frac{z_g^{(t)} w_g^{(t)}}{\lambda_g} (\boldsymbol{D}_g^\star)^{-1} \right]^{-1} \left[\sum_{g=1}^{G} \frac{z_g^{(t)} w_g^{(t)}}{\lambda_g} (\boldsymbol{D}_g^\star)^{-1} \boldsymbol{P}' \boldsymbol{\mu}_g \right],$$

$$\boldsymbol{x}_{NVMM}^{(t+1)} = \boldsymbol{P} \left[\sum_{g=1}^{G} \frac{z_g^{(t)} w_g^{(t)}}{\lambda_g} (\boldsymbol{D}_g^\star)^{-1} \right]^{-1} \left[\sum_{g=1}^{G} \frac{z_g^{(t)}}{\lambda_g} (\boldsymbol{D}_g^\star)^{-1} \boldsymbol{P}' (w_g^{(t)} \boldsymbol{\mu}_g + \boldsymbol{\alpha}_g) \right],$$

where one needs to invert diagonal matrices only, which is computationally lighter than a general matrix inversion.

Mode Initialization in StableMerge

The StableMerge is initialized with component-wise modes, so one must obtain the mode estimates first. With the PEMM and NVM-MM, the mode estimates $\hat{\boldsymbol{m}}_g$ are equal to component-wise means $\boldsymbol{\mu}_g$, as they do not model for skewness. However, with the NVMM-MM, the component-wise mean and mode are not equal unless the skewness vector $\boldsymbol{\alpha}_g$ is zero. Thus, the component-wise modes must be obtained first. The component-wise modes can be computed via mean-shift on individual components. For a single component k, the update $\boldsymbol{x}_k^{(t+1)}$ function simplifies to

$$\boldsymbol{x}_k^{(t+1)} = \boldsymbol{\mu}_k + \frac{\boldsymbol{\alpha}_k}{\mathrm{E}[W_k^{-1}|Z_k = 1, \boldsymbol{x}^{(t)}, \boldsymbol{\Theta}_k]}, \tag{8.14}$$

and the resulting mode estimates become the initialization for the StableMerge.

8.4 Numerical Experiments

In this section, we use simulated and real data sets to compare and contrast the performance of the StableMerge against several existing component-merging methods. The experimental scenarios and their objectives are described below.

- **Simulation: Detecting clusters of various shapes.** The cluster-detecting methods are compared on their cluster detection using a 2-dimensional, 6-components data set divided into three well-separated clusters.
- **Real data illustration: Olive.** The cluster-detecting methods are applied to the Olive data set, which records the eight fatty acid compositions of 572 Italian olive oils.

8.4.1 Considered Mixture Models

In all experimental scenarios, the following four finite mixture models are fitted using appropriate R packages. The selection criterion used during the fitting process is the Bayesian Information Criterion (BIC) by Schwarz (1978). Each model is fitted over a range of component count values, and the one resulting in the highest BIC value is selected.

- **Gaussian:** A GMM is fitted using the R package *mclust* (Scrucca et al., 2016). The component membership for the data set is initialized by the package's default method, which is agglomerative hierarchical clustering. In simulation, all observations in a replication are used for initialization. In real data experiment, 80% of observations are randomly chosen in each initialization.
- **Student-t:** A *t*MM is fitted using the R package *teigen* (Andrews et al., 2018). The component membership for the data set is initialized by k-means clustering for the simulation, and by random assignment for the real data experiment.
- **Power exponential:** A PEMM is fitted using the R package *mixSPE* (Browne et al., 2021). The component membership for the data set is initialized by k-means clustering for the simulation, and by random assignment for the real data experiment.
- **Generalized hyperbolic:** A GHMM is fitted using the R package *MixGHD* (Tortora et al., 2021). The component membership for the data set is initialized by k-means clustering for the simulation, and by random assignment for the real data experiment.

8.4.2 Merging Methods

On each mixture model, the following cluster-detecting methods are applied.

- **StableMerge.** StableMerge is the proposed method in this work.
- **ICL maximization.** Integrated Complete-data Likelihood (ICL) by Biernacki et al. (2000), as outlined in chapter 2.1.

- **DEMP+** as outlined in chapter 2.3. However, sampling from an arbitrary distribution can be technically complicated and computationally expensive. Therefore, in this work, we approximate $q_{\mathcal{G}_1|\mathcal{G}_2}$ using the posterior component membership probabilities $\hat{z}_{ig}$. Let

$$\mathcal{P}_s = \left\{ \left(\sum_{g \in \mathcal{G}_s} \hat{z}_{ig}, \sum_{k \in \mathcal{G}_t} \hat{z}_{ik} \right) \right\}_{i \,:\, i\text{th observation from } \mathcal{G}_s} \tag{8.15}$$

  denote the set of observation-wise posterior membership probability tuple for cluster s, and similarly $\mathcal{P}_t$ for cluster t. Then, for each $p \in \mathcal{P}_s$, a number from $\{s, t\}$ is sampled with $P(\text{choose } s) \propto p[1]$ and $P(\text{choose } t) \propto p[2]$. Finally, $\hat{q}_{\mathcal{G}_t|\mathcal{G}_s}$ is computed as the proportion of sampled numbers that are not s. The same process is applied to $\mathcal{P}_t$ to approximate $\hat{q}_{\mathcal{G}_s|\mathcal{G}_t}$. This approximation is faster than the originally-proposed estimation method, as it need not generate random samples from appropriate distributions, and the posterior membership probabilities are computed as part of the parameter estimation process.

- **EntropyMerge** as outlined in chapter 2.3.

Each method's performance is measured primarily by the Additive Margin (AM) by Ben-David and Ackerman (2009), and also by the Adjusted Rand Index (ARI) by Hubert and Arabie (1985), when the ground truth is available. Both measures are outline in chapter 2.1.

8.4.3 Simulated Data

In this experiment, the four cluster-detecting methods are compared using a 2-dimensional, 6-components data set divided into three clusters. The goal is to compare the efficacy of cluster-detecting methods on mixture models with over-estimated G. Each cluster is generated from the following distributions. The component-wise colours are shown in figure 8.2.

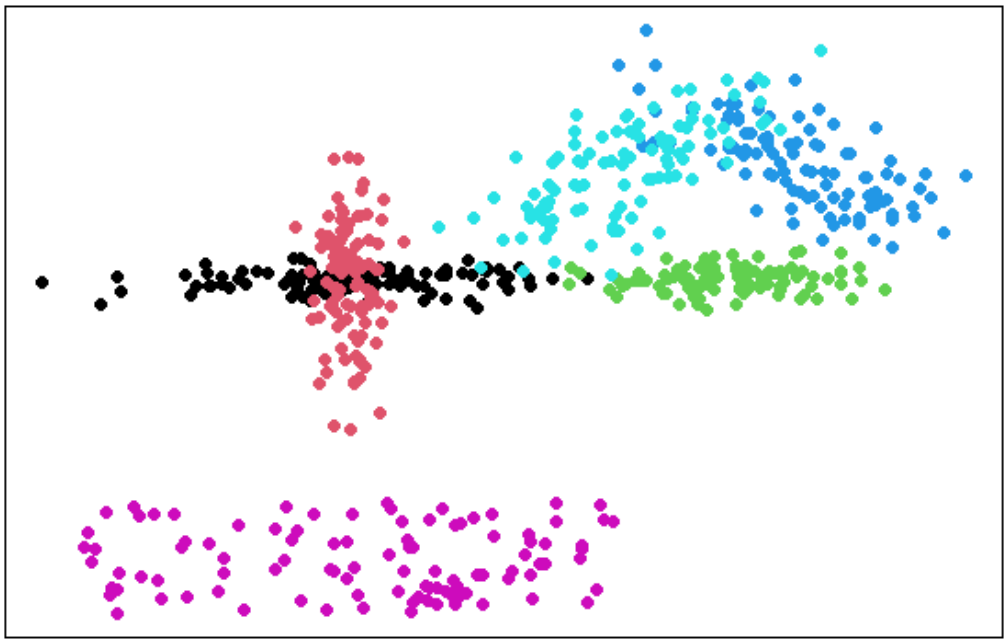

Figure 8.2: An instance of the simulated 2-dimensional, 6-component data set.

- Cross-shaped cluster (black, red):

$$\text{black} \sim N\left((3,0)', \begin{bmatrix} 1 & 0 \\ 0 & 0.05 \end{bmatrix}\right)$$
$$\text{red} \sim N\left((3,0)', \begin{bmatrix} 0.05 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

- Triangular cluster (green, blue, turquoise):

$$\text{green} \sim N\left((7,0)', \begin{bmatrix} 0.6 & 0 \\ 0 & 0.05 \end{bmatrix}\right)$$
$$\text{blue} \sim N\left((8,2)', \begin{bmatrix} 0.7 & -0.5 \\ -0.5 & 0.7 \end{bmatrix}\right)$$
$$\text{turquoise} \sim N\left((6,2)', \begin{bmatrix} 0.7 & 0.5 \\ 0.5 & 0.7 \end{bmatrix}\right)$$

- Rectangular cluster (magenta):

$$\text{magenta} \sim U[0,6] \times U[-6,-4]$$

The proportion of all components are equal at $\pi_g = 1/6$ for all g. Also, three values of sample size are considered: $n = 100, 200, 300$, and for each n, the component-wise sample size is $n_g = \lfloor n/6 \rfloor$ for all g. Once a data set is generated, the four mixture models are fitted over $G = 4, \ldots, 10$ and selected by the BIC. On the BIC-selected model, each of StableMerge, DEMP+ and EntropyMerge is applied to identify clusters. With ICL maximization, the mixture models are directly fitted over $G = 1, \ldots, 10$ instead, since the ICL seeks clusters at the outset. For each n, this process is replicated 500 times, each with a newly-generated data set.

$n = 100$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	7	**3**	4	2	4
	(2)	(1)	(1)	(2)	(0)
tMM	8	**3**	4	2	4
	(1)	(1)	(1)	(0)	(1)
PEMM	6	5	2	1	-
	(1)	(1)	(2)	(0)	-
GHMM	5	**3**	4	4	4
	(1)	(1)	(1)	(2)	(1)

$n = 200$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	6	4	4	**3**	4
	(2)	(2)	(1)	(2)	(1)
tMM	7	5	5	2	4
	(4)	(2)	(2)	(1)	(1)
PEMM	6	4	4	6	-
	(2.25)	(1)	(2)	(0)	-
GHMM	4	**3**	4	2	4
	(1)	(1)	(1)	(0)	(0)

$n = 300$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	7	4	5	2	5
	(2)	(1)	(1)	(1)	(2)
tMM	8	5	5	**3**	4
	(1)	(1)	(2)	(1)	(1)
PEMM	6	4	4	6	-
	(1)	(1)	(1)	(1)	-
GHMM	5	**3**	**3**	2	4
	(1)	(1)	(1)	(0)	(1)

Table 8.2: Tables of the median (and inter-quartile range in brackets) cluster count estimated by StableMerge, DEMP+, EntropyMerge and ICL. Init denotes the initial model selected by BIC before one of StableMerge or DEMP+ or EntropyMerge is applied. The top-left label in each table indicates the sample size used in generating the results in the corresponding table. The row labels indicate the mixture model upon which the cluster-detecting methods were applied. The number 3 is bolded, as it is the true number of clusters. ICL on PEMM is not reported as the mixSPE package did not support ICL maximization at the time of simulation.

$n = 100$	StableMerge	DEMP+	EntropyMerge	ICL
GMM	**287**	20	24	45
tMM	**302**	14	32	39
PEMM	27	**55**	2	-
GHMM	**197**	28	1	196

$n = 200$	StableMerge	DEMP+	EntropyMerge	ICL
GMM	**144**	29	112	18
tMM	147	12	**162**	1
PEMM	**166**	78	2	-
GHMM	**253**	57	7	77

$n = 300$	StableMerge	DEMP+	EntropyMerge	ICL
GMM	89	13	**126**	0
tMM	68	13	**192**	0
PEMM	**135**	64	2	-
GHMM	**253**	108	12	9

Table 8.3: Tables recording the number of replications where each cluster-detecting method identified 3 clusters over 500 replications. The row labels indicate the mixture model upon which the cluster-detecting methods were applied. ICL on PEMM is not reported as the mixSPE package did not support ICL maximization at the time of simulation. The highest count in each row is bolded.

Table 8.2 shows that all cluster-detecting methods identified significantly fewer numbers of clusters than the initial BIC-based model across all n. In particular, the StableMerge was most successful in identifying three clusters from the data set (5 out of 12 rows have

median G of 3), followed by EntropyMerge and DEMP+. Furthermore, the replication-wise comparison shown in table 8.3 indicates that the StableMerge identified three components most frequently in 8 out of 12 rows. EntropyMerge was more effective on more rigid mixture models (GMM, tMM). This demonstrates the efficacy of StableMerge in both rigid and flexible mixture distributions.

$n = 100$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.58	**0.81**	0.60	0.39	0.66
	(0.18)	(0.31)	(0.18)	(0.08)	(0.15)
tMM	0.58	**0.78**	0.62	0.39	0.67
	(0.18)	(0.41)	(0.21)	(0)	(0.16)
PEMM	0.38	**0.58**	0.29	0	-
	0.09	(0.11)	(0.67)	(0)	-
GHMM	0.66	**0.73**	0.67	0.43	0.72
	0.15	(0.36)	(0.22)	(0.27)	(0.32)

$n = 200$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.46	**0.69**	0.64	0.39	0.65
	(0.13)	(0.25)	(0.20)	(0.04)	(0.14)
tMM	0.41	0.66	0.57	0.39	**0.67**
	(0.15)	(0.20)	(0.18)	(0.03)	(0.17)
PEMM	0.50	**0.70**	0.69	0.50	-
	(0.15)	(0.25)	(0.22)	(0.15)	-
GHMM	0.60	**0.69**	0.61	0.39	0.66
	(0.16)	(0.35)	(0.20)	(0)	(0.17)

$n = 300$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.44	**0.69**	0.66	0.39	0.56
	(0.07)	(0.09)	(0.15)	(0.03)	(0.19)
tMM	0.40	**0.67**	0.61	0.39	0.66
	(0.06)	(0.06)	(0.14)	(0.03)	(0.18)
PEMM	0.48	**0.70**	**0.70**	0.48	-
	(0.12)	(0.16)	(0.17)	(0.12)	-
GHMM	0.53	**0.65**	0.53	0.39	0.55
	(0.16)	(0.21)	(0.16)	(0)	(0.14)

Table 8.4: Tables of the median (and inter-quartile range in brackets) ARI estimated by StableMerge, DEMP+, EntropyMerge and ICL, rounded to 2 decimal places. Init denotes the initial model selected by BIC before one of StableMerge or DEMP+ or EntropyMerge is applied. The top-left label in each table indicates the sample size used in generating the results in the corresponding table. The row labels indicate the mixture model upon which the cluster-detecting methods were applied. The highest row-wise median values are bolded. ICL on PEMM is not reported as the mixSPE package did not support ICL maximization at the time of simulation.

$n = 100$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.89	0.94	0.94	0.84	**1.00**
	(0.37)	(0.51)	(0.46)	(0.30)	(0.32)
tMM	0.93	**0.98**	0.91	0.82	0.97
	(0.39)	(0.48)	(0.43)	(0.28)	(0.32)
PEMM	0.65	**0.95**	0.32	0	-
	(0.17)	(0.28)	(0.76)	(0)	-
GHMM	0.96	0.94	0.97	0.91	**1.03**
	(0.33)	(0.47)	(0.39)	(0.29)	(0.35)

$n = 200$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.68	**0.90**	0.88	0.75	0.87
	(0.18)	(0.35)	(0.34)	(0.28)	(0.26)
tMM	0.73	**0.92**	0.91	0.77	0.91
	(0.23)	(0.33)	(0.31)	(0.27)	(0.28)
PEMM	0.67	0.86	**0.89**	0.66	-
	(0.16)	(0.34)	(0.33)	(0.15)	-
GHMM	0.86	0.80	0.85	0.82	**0.90**
	(0.28)	(0.51)	(0.37)	(0.28)	(0.29)

$n = 300$	Init	StableMerge	DEMP+	EntropyMerge	ICL
GMM	0.67	**0.93**	0.89	0.75	0.75
	(0.14)	(0.25)	(0.25)	(0.26)	(0.27)
tMM	0.66	**0.93**	0.90	0.77	0.87
	(0.14)	(0.26)	(0.31)	(0.25)	(0.29)
PEMM	0.65	0.88	**0.91**	0.65	-
	(0.15)	(0.33)	(0.27)	(0.15)	-
GHMM	0.72	0.66	0.70	**0.84**	0.75
	(0.24)	(0.41)	(0.29)	(0.25)	(0.25)

Table 8.5: Tables of the median (and inter-quartile range in brackets) AM obtained from StableMerge, DEMP+, EntropyMerge and ICL, rounded to 2 decimal places. Init denotes the initial model selected by BIC before one of StableMerge or DEMP+ or EntropyMerge is applied. The top-left label in each table indicates the sample size used in generating the results in the corresponding table. The row labels indicate the mixture model upon which the cluster-detecting methods were applied. The highest row-wise median values are bolded. ICL on PEMM is not reported as the mixSPE package did not support ICL maximization at the time of simulation.

In terms of clustering quality, the StableMerge exhibited a marked improvement in both ARI and AM, per tables 8.4 and 8.5. Interestingly, the ICL-based clusters were of similarly high quality to that of the StableMerge, especially at $n = 100, 200$, whereas its ARI often trailed behind that of the StableMerge. We could hypothesize from this pattern that the ICL is producing well-separated clusters in its own way, but not necessarily focusing on modality. Overall, this simulation demonstrates the benefit of using StableMerge on a variety of mixture models in detecting clusters characterized by modality.

8.4.4 Real Data Illustration: Olive

In this section, the Olive data set from the R package *pgmm* (McNicholas et al., 2018) is clustered via the four mixture models, using the four cluster-detecting methods. The data set is 8-dimensional consisting of the percentage composition of eight fatty acids in 572 Italian olive oils. The primary and secondary ground truths are the region and the area. The regions (and the areas within each region) are

- Southern Italy (North Apulia, Calabria, South Apulia, Sicily),
- Sardinia (Inland Sardinia, Coastal Sardinia),
- Northern Italy (East Liguria, West Liguria, Umbria).

	N. Apulia	Calabria	S. Apulia	Sicily
S. Italy	25	56	206	36

	I. Sardinia	C. Sardinia	E. Liguria	W. Liguria	Umbria
Sardinia	65	33			
N. Italy			50	50	51

Table 8.6: Table of observations per region (row) and area (column). Empty cells indicate zero observations.

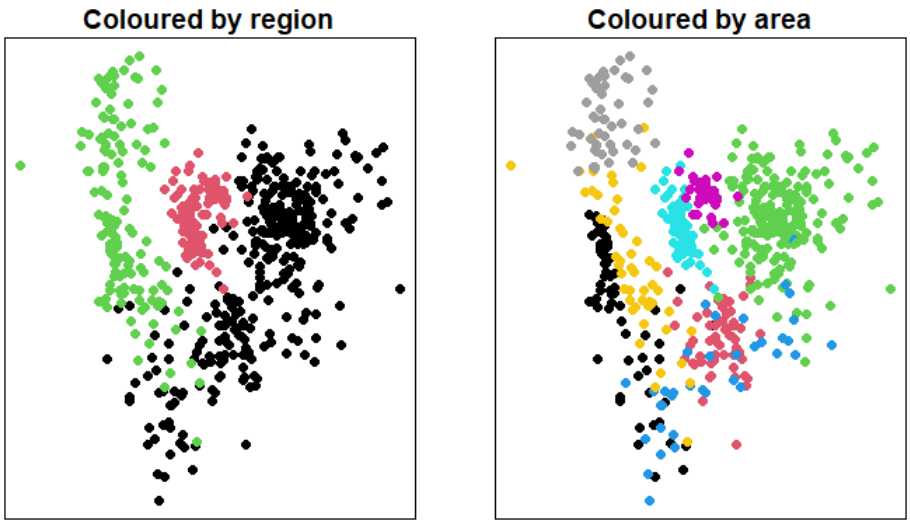

Figure 8.3: Scatterplot of the Olive data, projected onto the first two principal components, and coloured by region (left) and area (right).

Table 8.6 indicates a mild class imbalance, while figure 8.3 suggests deviations from normality both at region- and area-levels. Thus, more restrictive mixture models might over-estimate the number of regions and/or mis-identify the shape of area-wise clusters.

The range of components considered is $G = 1, \ldots, 10$ for initial BIC-based model and the ICL-based model. The models are fitted over 100 initializations, per the initialization strategy outlined in section 8.4.1. In terms of ARI, the value against both region and area are reported.

Model	Method	AM(Init)	AM	G(Init)	G	ARI(region)	ARI(area)
GMM	StableMerge	0.41	**0.79**	9	5	**0.60**	0.78
	DEMP+	0.46	0.67	8	7	0.54	**0.86**
	EntropyMerge	0.65	0.65	7	7	0.51	0.78
	ICL	-	0.70	-	7	0.52	0.84
tMM	StableMerge	0.64	0.78	6	5	0.53	0.72
	DEMP+	0.80	**0.80**	5	5	0.55	**0.74**
	EntropyMerge	0.39	0.60	8	3	**0.56**	0.27
	ICL	-	**0.80**	-	5	0.55	**0.74**
PEMM	StableMerge	0.51	**0.78**	6	5	**0.62**	**0.77**
	DEMP+	0.66	0.66	4	4	0.48	0.64
	EntropyMerge	0.64	0.64	6	6	0.58	0.76
	ICL	-	-	-	-	-	
GHMM	StableMerge	0.49	**0.75**	6	5	**0.61**	0.73
	DEMP+	0.74	0.74	5	5	0.53	0.69
	EntropyMerge	0.57	0.57	6	6	**0.61**	**0.76**
	ICL	-	0.74	-	5	0.51	0.72

Table 8.7: Table of cluster quality measurements from the cluster-detecting methods for GMM, tMM, PEMM and GHMM. For example, In the GMM subtable, the best (AM, G) pair obtained from the StableMerge is (0.79, 5), which is based on a preliminary GMM (chosen by the BIC) with (AM(Init), G(Init)) = (0.41, 9). The same StableMerge solution produced ARI of 0.6 against regions (3 classes), and 0.78 against areas (9 classes). For the ICL, no AM(Init) or G(Init) are reported since the mixture model is fitted directly using the criterion. An analogous interpretation applies to the remaining subtables. For the PEMM, the ICL row is not reported as the mixSPE package did not support ICL-based model selection at the time of the experiment. The AM and ARI are rounded to 2 decimal places, and within each subtable, the highest AM, ARI(region) and ARI(area) values are bolded.

Table 8.7 reports the best instances of the cluster-detecting methods over the 100 initializations, for each mixture model. The StableMerge produced the best clustering, per AM, for GMM, PEMM and GHMM. In particular, its extent of improvement in AM ranges between $((0.78 - 0.64) \div 0.64)\% \approx 22\%$ and $((0.79 - 0.41) \div 0.41)\% \approx 93\%$. This means

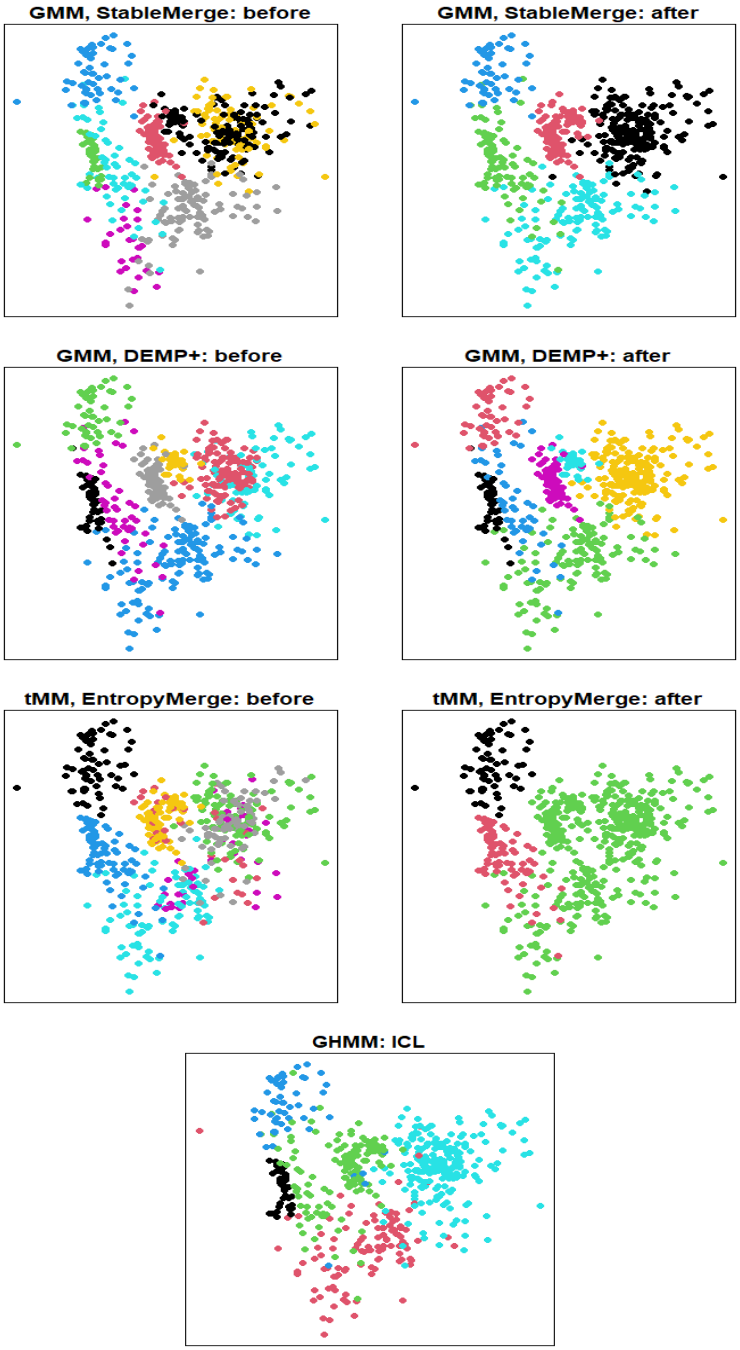

Figure 8.4: Scatterplot examples of the Olive data set on its first two principal components, coloured by the components generated by (GHMM, StableMerge), (GMM, DEMP+), (tMM, EntropyMerge) and (GHMM, ICL).

that the StableMerge was able to reduce the component count for all four mixture models, whose flexibility varies considerably. Another point of interest is that the best instances of DEMP+ and EntropyMerge were often identical to their preliminary models, as shown from the unchanged AM and G. This means that the StableMerge was able to handle a poor initial fit better than the aforementioned two methods. The ICL performed consistently well, and it appears to have preferred the more granular area-wise grouping than the region-wise one. Figure 8.4 shows illustrative instances of pre- and post-merging. The most drastic changes are shown from the (GMM, StableMerge) and (tMM, EntropyMerge) pairs, where the number of components was reduced from 9 to 5, and from 8 to 3, respectively, per table 8.7. Overall, this analysis shows that the StableMerge is a viable option for improving the clustering structure of a wide range of mixture models, even when the initial fit may over-estimate the component count significantly.

8.5 Discussion

In this chapter, a mixture component merging framework by detecting component-wise modes through novel mean-shift algorithms for the PEMM, NVM-MM and NVMM-MM, and the StableMerge, a novel stability-based mode-merging procedure that is threshold-free. We have demonstrated its effectiveness in both simulated and real data settings against various existing methodologies. Directions for further research include the application of the StableMerge to other clustering methods, and the development of mean-shift algorithms for other finite mixture models.

Chapter 9

Conclusion

This thesis introduced several novel methodologies for parsimonious finite mixture modelling and mixture component merging, in order to enhance the presence of interpretable methods in the model-based clustering literature. Chapters 3 and 4 showed the potential of the novel Stiefel Elastic Net in estimating matrix parameters with simpler structures, supported by desirable theoretical properties. Chapter 5 showcased a hypothesis test-based alternative to the scree test where both the hypotheses and the hyper-parameter are readily interpreted. Chapter 6 showed that a highly flexible mixture regression model could still benefit from combining its components, where the aggregated components were better distinguished than before. Chapters 7 and 8 enabled the detection of density modes in several families of mixture models with varying flexibility, which would be particularly useful in applications where modes signify clusters. The approaches taken in this collection are certainly not exhaustive. Model interpretability has many aspects, each of which can be address differently. Furthermore, there are numerous directions for further work, as mentioned in the discussion section of each chapter.

References

K. Adachi and N. T. Trendafilov. Sparse orthogonal factor analysis. In *Advances in Latent Variables*, pages 227–239. Springer, 2014.

K. Adachi and N. T. Trendafilov. Sparsest factor analysis for clustering variables: A matrix decomposition approach. *Advances in Data Analysis and Classification*, 12(3):559–585, 9 2018. doi: 10.1007/s11634-017-0284-z.

A. C. Aitken. On Bernoulli's numerical solution of algebraic equations. *Proceedings of the Royal Society of Edinburgh*, 46:289–305, 1926.

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19(6):716–723, 1974.

T. W. Anderson. *An introduction to multivariate statistical analysis, 3rd edition.* Wiley Series in Probability and Statistics, 2003.

J. L. Andrews and P. D. McNicholas. Extending mixtures of multivariate t-factor analyzers. *Statistics and Computing*, 21(3):361–373, 2011.

J. L. Andrews and P. D. Mcnicholas. Model-based clustering, classification, and discriminant analysis via mixtures of multivariate t-distributions. *Statistics and Computing*, 22 (5):1021–1029, 2012.

J. L. Andrews, J. R. Wickins, N. M. Boers, and P. D. McNicholas. teigen: An R package for model-based clustering and classification via the multivariate t distribution. *Journal of Statistical Software*, 83(7):1–32, 2018.

K. Askew. Counting the cost of fish fraud: 'billions' lost to illicit fisheries, 2020. URL https://www.foodnavigator.com/Article/2020/03/12/Counting-the-cost-of-fish-fraud-Billions-lost-to-illicit-fisheries.

J. D. Banfield and A. E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, pages 803–821, 1993.

O. E. Barndorff-Nielsen. Hyperbolic distributions and distributions on hyperbolae. *Scandinavian Journal of Statistics*, pages 151–157, 1978.

O. E. Barndorff-Nielsen and N. Shephard. Non-Gaussian Ornstein–Uhlenbeck-based models and some of their uses in financial economics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):167–241, 2001.

J.-P. Baudry, A. E. Raftery, G. Celeux, K. Lo, and R. Gottardo. Combining mixture components for clustering. *Journal of computational and graphical statistics*, 19(2):332–353, 2010.

R. Bellman. *Dynamic programming*. Princeton University Press, USA, 2010. ISBN 0691146683.

S. Ben-David and M. Ackerman. Measures of clustering quality: A working set of axioms for clustering. In *Advances in neural information processing systems*, pages 121–128, 2009.

T. Benaglia, D. Chauveau, D. R. Hunter, and D. Young. mixtools: An R package for analyzing finite mixture models. *Journal of Statistical Software*, 32(6):1–29, 2009. URL http://www.jstatsoft.org/v32/i06/.

L. Bergé, C. Bouveyron, and S. Girard. HDclassif: An R package for model-based clustering and discriminant analysis of high-dimensional data. *Journal of Statistical Software*, 46 (6):1–29, 2012. URL http://www.jstatsoft.org/v46/i06/.

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE transactions on pattern analysis and machine intelligence*, 22(7):719–725, 2000.

D. Böhning, E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 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, 1994.

M. Bolla, G. Michaletzky, G. Tusnády, and M. Ziermann. Extrema of sums of heterogeneous quadratic forms. *Linear Algebra and its Applications*, 269(1-3):331–365, 1998.

H. W. Borchers. *pracma: Practical Numerical Math Functions*, 2021. URL https://CRAN.R-project.org/package=pracma. R package version 2.3.3.

C. Bouveyron, S. Girard, and C. Schmid. High-dimensional data clustering. *Computational statistics & data analysis*, 52(1):502–519, 2007.

C. Bouveyron, E. Côme, J. Jacques, et al. The discriminative functional mixture model for a comparative analysis of bike sharing systems. *The Annals of Applied Statistics*, 9 (4):1726–1760, 2015.

R. P. Browne and P. D. McNicholas. Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification*, 8(2):217–226, 2014.

R. P. Browne and P. D. McNicholas. A mixture of generalized hyperbolic distributions. *Canadian Journal of Statistics*, 43(2):176–198, 2015.

R. P. Browne, U. J. Dang, M. P. B. Gallaugher, and P. D. McNicholas. *mixSPE: Mixtures of power exponential and skew power exponential distributions for use in model-based clustering and classification*, 2021. URL https://CRAN.R-project.org/package=mixSPE. R package version 0.9.1.

J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Transactions on pattern analysis and machine intelligence*, 20 (5):572–575, 1998.

D. Cai, X. He, and J. Han. Spectral regression for efficient regularized subspace learning. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

F. Camastra. Data dimensionality estimation methods: a survey. *Pattern recognition*, 36 (12):2945–2954, 2003.

E. Candes, T. Tao, et al. The Dantzig selector: Statistical estimation when p is much larger than n. *Annals of Statistics*, 35(6):2313–2351, 2007.

S. Cao, W. Chang, and C. Zhang. *RobMixReg: robust mixture regression*, 2020. URL https://CRAN.R-project.org/package=RobMixReg.

M. A. Carreira-Perpinan. Mode-finding for mixtures of Gaussian distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, 2000.

M. A. Carreira-Perpinan. Gaussian mean-shift is an EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):767–776, 2007.

K. M. Carter, R. Raich, and A. O. Hero III. On local intrinsic dimension estimation and its applications. *IEEE Transactions on Signal Processing*, 58(2):650–663, 2009.

R. B. Cattell. The scree test for the number of factors. *Multivariate Behavioral Research*, 1(2):245–276, 1966.

G. Celeux and J. Diebolt. The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2: 73–82, 1985.

J. E. Chacón. Mixture model modal clustering. *Advances in Data Analysis and Classification*, 13(2):379–404, 2019.

J. E. Chacón. The modal age of statistics. *International Statistical Review*, 88(1):122–141, 2020.

F. Chamroukhi. Robust mixture of experts modeling using the t distribution. *Neural Networks*, 79:20–36, 2016.

F. Chamroukhi. Skew t mixture of experts. *Neurocomputing*, 266:390–408, 2017.

D. B. Clarkson and R. I. Jennrich. Quartic rotation criteria and algorithms. *Psychometrika*, 53(2):251–259, 1988.

W. S. Cleveland. LOWESS: A program for smoothing scatterplots by robust locally weighted regression. *American Statistician*, 35(1):54, 1981.

D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

R. D. Cook and X. Yin. Dimension reduction and visualization in discriminant analysis (with discussion). *Australian & New Zealand Journal of Statistics*, 43(2):147–199, 2001.

U. J. Dang, R. P. Browne, and P. D. McNicholas. Mixtures of multivariate power exponential distributions. *Biometrics*, 71(4):1081–1089, 2015.

U. J. Dang, M. P. B. Gallaugher, R. P. Browne, and P. D. McNicholas. Model-based clustering and classification using mixtures of multivariate skewed power exponential distributions. *arXiv preprint:1907.01938*, 2019.

N. E. Day. Estimating the components of a mixture of normal distributions. *Biometrika*, 56(3):463–474, 1969.

J. de Leeuw and K. Lange. Sharp quadratic majorization in one dimension. *Computational Statistics & Data Analysis*, 53(7):2471–2484, 2009.

R. D. De Veaux. Mixtures of linear regressions. *Compuational Statistics & Data Analysis*, 8:227–245, 1989.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

S. Epskamp, A. O. J. Cramer, L. J. Waldorp, V. D. Schmittmann, and D. Borsboom. qgraph: Network visualizations of relationships in psychometric data. *Journal of Statistical Software*, 48(4):1–18, 2012. URL http://www.jstatsoft.org/v48/i04/.

N. B. Erichson, P. Zheng, and S. Aravkin. *sparsepca: Sparse Principal Component Analysis (SPCA)*, 2018. URL https://CRAN.R-project.org/package=sparsepca. R package version 0.1.2.

N. B. Erichson, P. Zheng, K. Manohar, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin. Sparse principal component analysis via variable projection. *SIAM Journal on Applied Mathematics*, 80(2):977–1002, 2020.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

M. Fan, N. Gu, H. Qiao, and B. Zhang. Intrinsic dimension estimation of data by principal component analysis. *arXiv preprint:1002.2050*, 2010.

G. A. Ferguson. The concept of parsimony in factor analysis. *Psychometrika*, 19(4):281–290, 1954.

E. Fokoué and D. M. Titterington. Mixtures of factor analysers. Bayesian estimation and inference by stochastic simulation. *Machine Learning*, 50(1-2):73–94, 2003.

C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002.

B. C. Franczak, R. .P Browne, and P. D. McNicholas. Mixtures of shifted asymmetric Laplace distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1149–1157, 2013.

B. C. Franczak, R. P. Browne, P. D. McNicholas, and K. L. Burak. *MixSAL: Mixtures of multivariate shifted asymmetric Laplace (SAL) distributions*, 2018. URL https://CRAN.R-project.org/package=MixSAL. R package version 1.0.

J. Friedman, T. Hastie, R. Tibshirani, et al. *The elements of statistical learning*, volume 1. Springer Series in Statistics New York, 2001.

K. Fukunaga and D. R. Olsen. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 100(2):176–183, 1971.

L. A. García-Escudero, A. Gordaliza, F. Greselin, S. Ingrassia, and A. Mayo-Íscar. Robust estimation of mixtures of regressions with random covariates, via trimming and constraints. *Statistics and Computing*, 27(2):377–402, 2017.

N. Gershenfeld. Nonlinear inference and cluster-weighted modeling. *Annals of the New York Academy of Sciences*, 808(1):18–24, 1997.

Z. Ghahramani and G. E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical report, Technical Report CRG-TR-96-1, University of Toronto, 1996.

E. Gómez, M. A. Gomez-Viilegas, and J. M. Marín. A multivariate generalization of the power exponential family of distributions. *Communications in Statistics-Theory and Methods*, 27(3):589–600, 1998.

E Gómez-Sánchez-Manzano, MA Gómez-Villegas, and JM Marín. Multivariate exponential power distributions as mixtures of normal distributions with bayesian applications. *Communications in Statistics—Theory and Methods*, 37(6):972–985, 2008.

B. Grün and F. Leisch. FlexMix version 2: finite mixtures with concomitant variables and varying and constant parameters. *Journal of Statistical Software*, 28(4):1–35, 2008. URL https://www.jstatsoft.org/v28/i04/.

W. K. Härdle et al. *Smoothing techniques: with implementation in S*. Springer Science & Business Media, 1991.

C. Hennig. Identifiablity of models for clusterwise linear regression. *Journal of Classification*, 17(2):273–296, 2000.

C. Hennig. Methods for merging Gaussian mixture components. *Advances in Data Analysis and Classification*, 4(1):3–34, 2010.

K. Hirose and M. Yamamoto. Estimation of an oblique structure via penalized likelihood factor analysis. *Computational Statistics and Data Analysis*, 79:120–132, 2014. doi: 10.1016/j.csda.2014.05.011.

K. Hirose and M. Yamamoto. Sparse estimation via nonconcave penalized likelihood in factor analysis model. *Statistics and Computing*, 25(5):863–875, 2015.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, 2012.

A. Houdard, C. Bouveyron, and J. Delon. High-dimensional mixture models for unsupervised image denoising (HDMI). *SIAM Journal on Imaging Sciences*, 11(4):2815–2846, 2018.

H. Hu, W. Yao, and Y. Wu. The robust EM-type algorithms for log-concave mixtures of regression models. *Computational Statistics & Data Analysis*, 111:14–26, 2017.

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

D. R. Hunter and K. Lange. Quantile regression via an MM algorithm. *Journal of Computational and Graphical Statistics*, 9(1):60–77, 2000.

D. R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58 (1):30–37, 2004.

D. R. Hunter and D. S. Young. Semiparametric mixtures of regressions. *Journal of Nonparametric Statistics*, 24(1):19–38, 2012.

C. Hurley. *gclus: Clustering graphics*, 2019. URL https://CRAN.R-project.org/package=gclus. R package version 1.3.2.

S. Ingrassia, S. C. Minotti, and G. Vittadini. Local statistical modeling via a cluster-weighted approach with elliptical distributions. *Journal of Classification*, 29(3):363–401, 2012.

S. Ingrassia, S. C. Minotti, and A. Punzo. Model-based clustering via linear cluster-weighted models. *Computational Statistics & Data Analysis*, 71:159–182, 2014.

ISTAT. Italian tourist flow data (retrieved from www.robertocellini.it), 2013. URL http://www.robertocellini.it/doc/master_specializzazione/Cellini-Cuccia_ApEc2013_data1996-2010.pdf.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *ISLR: Data for an introduction to statistical learning with applications in R*, 2017. URL https://CRAN.R-project.org/package=ISLR. R package version 1.2.

J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1):175–193, 1906.

K. Johnsson and Lund University. *intrinsicDimension: Intrinsic dimension estimation*, 2019. URL https://CRAN.R-project.org/package=intrinsicDimension. R package version 1.2.0.

K. Johnsson, C. Soneson, and M. Fontes. Low bias local intrinsic dimension estimation from expected simplex skewness. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):196–202, 2014.

Kaggle. Movehub city rankings. https://www.kaggle.com/blitzr/movehub-city-rankings, 2017.

H. F. Kaiser. The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, 23(3):187–200, 1958.

H. A. L. Kiers. Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. *Computational Statistics & Data Analysis*, 41(1):157–170, 2002.

N.-H. Kim and R. Browne. Subspace clustering for the finite mixture of generalized hyperbolic distributions. *Advances in Data Analysis and Classification*, 13(3):641–661, 2019.

N.-H. Kim and R. P. Browne. Mode merging for the finite mixture of t-distributions. *Stat*, 10(1):e372, 2021a.

N.-H. Kim and R. P. Browne. In the pursuit of sparseness: A new rank-preserving penalty for a finite mixture of factor analyzers. *Computational Statistics & Data Analysis*, 160: 107244, 2021b.

S. Kotz, T. Kozubowski, and K. Podgorski. *The Laplace distribution and generalizations: A revisit with applications to communications, economics, engineering, and finance*. Springer Science & Business Media, 2012.

S. X. Lee and G. J. McLachlan. On mixtures of skew normal and skew t-distributions. *Advances in Data Analysis and Classification*, 7(3):241–266, 2013.

E. Levina and P. J. Bickel. *Maximum likelihood estimation of intrinsic dimension*. 2005.

K.-C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

D. Liang and C.-F. Tsai. Company bankrupty prediction, 2016. URL https://www.kaggle.com/fedesoriano/company-bankruptcy-prediction.

T.-I. Lin, J. C. Lee, and W. J. Hsieh. Robust mixture modeling using the skew t distribution. *Statistics and Computing*, 17(2):81–92, 2007a.

T.-I. Lin, J. C. Lee, and S. Y. Yen. Finite mixture modelling using the skew normal distribution. *Statistica Sinica*, pages 909–927, 2007b.

T.-I. Lin, G. J. McLachlan, and S. X. Lee. Extending mixtures of factor models using the restricted multivariate skew-normal distribution. *Journal of Multivariate Analysis*, 143: 398–413, 2016.

M. J. Lindstrom and D. M. Bates. Newton—Raphson and EM algorithms for linear mixed-effects models for repeated-measures data. *Journal of the American Statistical Association*, 83(404):1014–1022, 1988.

M. Liu and T.-I. Lin. A skew-normal mixture regression model. *Educational and Psychological Measurement*, 74(1):139–162, 2014.

Y. Ma, S. Wang, L. Xu, and W. Yao. Semiparametric mixture regression with unspecified error distributions. *Test*, 30(2):429–444, 2021.

G. McLachlan and D. Peel. *Finite mixture models*. John Wiley & Sons, 2004.

G. J. McLachlan, R. W. Bean, and L. B.-T. Jones. Extension of the mixture of factor analyzers model to incorporate the multivariate t-distribution. *Computational Statistics & Data Analysis*, 51(11):5327–5338, 2007.

A. J. McNeil, R. Frey, and P. Embrechts. *Quantitative risk management: concepts, techniques and tools-revised edition*. Princeton university press, 2015.

P. D. McNicholas. *Mixture model-based classification*. CRC press, 2016.

P. D. McNicholas and T. B. Murphy. Parsimonious Gaussian mixture models. *Statistics and Computing*, 18(3):285–296, 2008.

Paul D. McNicholas, Aisha ElSherbiny, Aaron F. McDaid, and T. Brendan Murphy. *pgmm: Parsimonious Gaussian Mixture Models*, 2018. URL https://CRAN.R-project.org/package=pgmm. R package version 1.2.3.

S. M. McNicholas, P. D. McNicholas, and R. P. Browne. A mixture of variance-gamma factor analyzers. In *Big and Complex Data Analysis*, pages 369–385. Springer, 2017.

V. Melnykov. Merging mixture components for clustering through pairwise overlap. *Journal of Computational and Graphical Statistics*, 25(1):66–90, 2016.

G. Menardi. A review on modal clustering. *International Statistical Review*, 84(3):413–433, 2016.

X.-L. Meng and D. B. Rubin. Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993.

X.-L. Meng and D. Van Dyk. The EM algorithm—an old folk-song sung to a fast new tune. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(3): 511–567, 1997.

Movehub. Movehub city rankings. https://www.movehub.com/city-rankings/, 2019.

P. M. Murray, R. P. Browne, and P. D. McNicholas. Mixtures of skew-t factor analyzers. *Computational Statistics & Data Analysis*, 77:326–335, 2014.

P. M. Murray, R. P. Browne, and P. D. McNicholas. Mixtures of hidden truncation hyperbolic factor analyzers. *Journal of Classification*, 37(2):366–379, 2020.

J. O. Neuhaus and C. Wrigley. The Quartimax method: An analytic approach to orthogonal simple structure 1. *British Journal of Statistical Psychology*, 7(2):81–91, 1954.

N. Neykov, P. Filzmoser, R. Dimova, and P. Neytchev. Robust fitting of mixtures using the trimmed likelihood estimator. *Computational Statistics & Data Analysis*, 52(1):299–308, 2007.

F. Nie, H. Huang, X. Cai, and C. Ding. Efficient and robust feature selection via joint $l_{2,1}$-norms minimization. *Proc. Adv. Neural Inf.Process. Syst.*, pages 1813 –1821, 2010.

OECD. OECD tourism trends and policies 2020, 2020. URL https://www.oecd-ilibrary.org/sites/3d4192c2-en/index.html?itemId=/content/component/3d4192c2-en.

W. Pan and X. Shen. Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research*, 8(May):1145–1164, 2007.

K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

D. Peel and G. J. McLachlan. Robust mixture modelling using the t distribution. *Statistics and Computing*, 10(4):339–348, 2000.

D. Peña, F. J. Prieto, and J. Viladomat. Eigenvectors of a kurtosis matrix as interesting directions to reveal cluster structure. *Journal of Multivariate Analysis*, 101(9):1995–2007, 2010.

A. Pesevski, B. C. Franczak, and P. D. McNicholas. Subspace clustering with the multivariate-t distribution. *Pattern Recognition Letters*, 112:297–302, 2018.

V. Pestov. An axiomatic approach to intrinsic dimension of a dataset. *Neural Networks*, 21(2-3):204–213, 2008.

A. Punzo and P. D. McNicholas. Parsimonious mixtures of multivariate contaminated normal distributions. *Biometrical Journal*, 58(6):1506–1537, 2016.

A. Punzo and P. D. McNicholas. Robust clustering in regression analysis via the contaminated Gaussian cluster-weighted model. *Journal of Classification*, 34(2):249–293, 2017.

A. Pyae. Fish market data set, 2019. URL https://www.kaggle.com/aungpyaeap/fish-market/metadata.

R Core Team. *R: a language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL https://www.R-project.org/.

C. Ramirez, R. Sanchez, V. Kreinovich, and M. Argaez. $\sqrt{x^2+\mu}$ is the most computationally efficient smooth approximation to $|x|$: A proof. *Journal of Uncertain Systems*, 8(3):205–210, 2014.

W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

A. V. Rao, D. Miller, K. Rose, and A. Gersho. Mixture of experts regression modeling by deterministic annealing. *IEEE Transactions on Signal Processing*, 45(11):2811–2820, 1997.

C. R. Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society: Series B (Methodological)*, 10(2):159–193, 1948.

S. Ray and B. G. Lindsay. The topography of multivariate normal mixtures. *The Annals of Statistics*, 33(5):2042–2065, 2005.

S. Ray and D. Ren. On the upper bound of the number of modes of a multivariate normal mixture. *Journal of Multivariate Analysis*, 108:41–52, 2012.

D. B. Rubin and D. T. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47 (1):69–76, 1982.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, pages 461–464, 1978.

D. W. Scott and W. F. Szewczyk. From kernels to mixtures. *Technometrics*, 43(3):323–335, 2001.

L. Scrucca. Dimension reduction for model-based clustering. *Statistics and Computing*, 20 (4):471–484, 2010.

L. Scrucca. Graphical tools for model-based mixture discriminant analysis. *Advances in Data Analysis and Classification*, 8(2):147–165, 2014.

L. Scrucca. Identifying connected components in Gaussian finite mixture models for clustering. *Computational Statistics & Data Analysis*, 93:5–17, 2016.

L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, 8(1):205–233, 2016. URL https://journal.r-project.org/archive/2016-1/scrucca-fop-murphy-etal.pdf.

G. A. F. Seber. *A matrix handbook for statisticians*, volume 15. John Wiley & Sons, 2008.

A. Sharp and R. Browne. Functional data clustering by projection into latent generalized hyperbolic subspaces. *Advances in Data Analysis and Classification*, pages 1–23, 2021.

W. Song, W. Yao, and Y. Xing. Robust mixture regression model fitting by Laplace distribution. *Computational Statistics & Data Analysis*, 71:128–137, 2014.

N. Städler, P. Bühlmann, and S. Van De Geer. l1-penalization for mixture regression models. *Test*, 19(2):209–256, 2010.

Statistica. Number of visitors to state museums, monuments, archaeological sites, and museum complexes with both free and paying entrance in italy in 2019, by month, 2020. URL https://www.statista.com/statistics/737980/visits-to-paying-free-state-museums-monuments-and-archeological-sites-by-month-italy

D. Steinley. Properties of the Hubert-Arable adjusted Rand Index. *Psychological Methods*, 9(3):386, 2004.

K. Strimmer, T. Jendoubi, A. Kessy, and A. Lewin. *whitening: Whitening and high-dimensional canonical correlation analysis*, 2020. URL https://CRAN.R-project.org/package=whitening. R package version 1.2.0.

S. Subedi, A. Punzo, S. Ingrassia, and P. D. McNicholas. Clustering and classification via cluster-weighted factor analyzers. *Advances in Data Analysis and Classification*, 7(1): 5–40, 2013.

N. Sugiura and H. Nagao. Unbiasedness of some test criteria for the equality of one or two covariance matrices. *The Annals of Mathematical Statistics*, 39(5):1686–1692, 1968.

F. Takens, H. W. Broer, and B. L. J. Braaksma. *Dynamical systems and bifurcations*. Springer Verlag, 1985.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical analysis of finite mixture distributions*. Wiley, 1985.

C. Tortora, P. D. McNicholas, and R. P. Browne. A mixture of generalized hyperbolic factor analyzers. *Advances in Data Analysis and Classification*, 10(4):423–440, 2016.

C. Tortora, R. P. Browne, A. El Sherbiny, B. C. Franczak, and P. D. McNicholas. Model-based clustering, classification, and discriminant analysis using the generalized hyperbolic distribution: MixGHD R package. *Journal of Statistical Software*, 98(3):1–24, 2021. doi: 10.18637/jss.v098.i03.

Travel and Leisure. The best and worst times to visit italy, 2021. URL https://www.travelandleisure.com/travel-tips/best-time-to-visit-italy.

L. N. Trefethen and D. Bau III. *Numerical linear algebra*, volume 50. SIAM, 1997.

N. T. Trendafilov and K. Adachi. Sparse versus simple structure loadings. *Psychometrika*, 80(3), 2015. ISSN 00333123. doi: 10.1007/s11336-014-9416-y.

N. T. Trendafilov, S. Fontanella, and K. Adachi. Sparse exploratory factor analysis. *Psychometrika*, 82(3):778–794, 9 2017. ISSN 00333123. doi: 10.1007/s11336-017-9575-8.

G. V. Trunk. Stastical estimation of the intrinsic dimensionality of a noisy signal collection. *IEEE Transactions on Computers*, 100(2):165–171, 1976.

D. E. Tyler, F. Critchley, L. Dümbgen, and H. Oja. Invariant co-ordinate selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):549–592, 2009.

UN. The state of world fisheries and aquaculture 2020, 2020. URL http://www.fao.org/state-of-fisheries-aquaculture.

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL http://www.stats.ox.ac.uk/pub/MASS4. ISBN 0-387-95457-0.

I. Vrbik and P. D. McNicholas. Parsimonious skew mixture models for model-based clustering and classification. *Computational Statistics & Data Analysis*, 71:196–210, 2014.

L. Wang, M. D. Gordon, and J. Zhu. Regularized least absolute deviations regression and an efficient algorithm for parameter tuning. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 690–700. IEEE, 2006.

K. Warner, W. Timme, B. Lowell, and M. Hirschfield. *Oceana study reveals seafood fraud nationwide*. Oceana Washington, DC, 2013.

D. J. Wilson. The harmonic mean p-value for combining dependent tests. *Proceedings of the National Academy of Sciences*, 116(4):1195–1200, 2019.

J. H. Wolfe. *Object cluster analysis of social areas*. PhD thesis, University of California, 1963.

J. H. Wolfe. Normix: Computational methods for estimating the parameters of multivariate normal mixtures of distributions. Technical report, Naval Personnel Research Activity San Diego Calif, 1967.

J. H. Wolfe. Pattern clustering by multivariate mixture analysis. *Multivariate Behavioral Research*, 5(3):329–350, 1970.

C. F. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, pages 95–103, 1983.

B. Xie, W. Pan, and X. Shen. Penalized model-based clustering with cluster-specific diagonal covariance matrices and grouped variables. *Electronic Journal of Statistics*, 2:168, 2008.

W. Yao, Y. Wei, and C. Yu. Robust mixture regression using the t-distribution. *Computational Statistics & Data Analysis*, 71:116–127, 2014.

C. Yu, W. Yao, and K. Chen. A new method for robust mixture regression. *Canadian Journal of Statistics*, 45(1):77–94, 2017.

C. Yu, W. Yao, and G. Yang. A selective overview and comparison of robust mixture regression estimators. *International Statistical Review*, 88(1):176–202, 2020.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

X.-T. Yuan, B.-G. Hu, and R. He. Agglomerative mean-shift clustering. *IEEE Transactions on Knowledge and Data Engineering*, 24(2):209–219, 2010.

C.-H. Zhang et al. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010.

H. Zhang, J. Yang, J. Xie, J. Qian, and B. Zhang. Weighted sparse coding regularized nonconvex matrix regression for robust face recognition. *Information Sciences*, 394: 1–17, 2017.

J. Zhang and F. Liang. Robust clustering using exponential power mixtures. *Biometrics*, 66(4):1078–1086, 2010.

H. Zhou and L. Li. Regularized matrix regression. *Journal of the Royal Statistical Society: Series B (Methodological)*, 76(2):463–483, 2014.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Methodological)*, 67(2):301–320, 2005.