THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

# Deep Gaussian Mixtures Models with Common Factor Loadings

David Banh

*A thesis submitted for the degree of Master of Science at The University of Queensland in 2020*

School of Mathematics and Physics

# Abstract

The use of depth in a hierarchical setting has provided great flexibility and power to fit a wide range of models on large complex datasets with the advent of Deep Learning in Neural Networks. The novel introduction of Deep Gaussian Mixture Models by Viroli and McLachlan (2019), works with great success in unsupervised clustering on a range of datasets. This thesis now introduces an improvement on top of the deep Mixture Models, through enabling it to visualise the latent factor scores, with the design based on a hierarchical setting of Mixtures of Common Factor Analysers in the latent variable space initially introduced by Baek, McLachlan and Flack (2010). It is here we show that the new design compares well on small datasets with state of the art visualisation methods such as t-Stochastic Neighbourhood Embedding and Uniform Manifold Approximation and Projection. Most importantly, Deep Gaussian Mixture Models with Common Factor Loadings are capable of unifying dimensionality reduction, unsupervised clustering, and visualisation.

1. The code for the original Deep Gaussian Mixture Model can be found here:
   https://github.com/suren-rathnayake/deepgmm
2. The code for the original Mixtures of Common Factor Analysers can be found here:
   https://github.com/suren-rathnayake/EMMIXmfa
3. All code for the Deep Gaussian Mixture Model with Common Factor Loadings is provided at:
   https://github.com/david-b-123/dgmmcfl

## Declaration by author

*(All candidates to reproduce this section in their thesis verbatim)*

# Contents

# Chapter 1

# Introduction

## 1.1 Overview of Deep Hierarchical Models

Advances made by modifying statistical and machine learning models to be more hierarchical and non-linear, with a greater number of transformations between the regression variable and the covariates has greatly helped the field find direction - from mixtures of linear models, through to mixture of experts in the machine learning domain, and linear mixed models widely used in the field of modern day genomics [1].

The idea of combining simple units to make more complex machinery has been a widely held practice in many practical quantitative domains, from the simple model of the single layer perceptron, to the construction of the modern era of deep learning, a culmination of several decades of research from the simplest of ideas to the development of a world-changing revolution in artificial intelligence [2].

The purpose of deep models is to extract valuable information from hierarchical layers of interconnected latent variables. By combining multiple layers of latent variables, the idea is to construct a richer model, whereby the density of the probabilistic model is more flexible and has greater fit on a wider array of more complex non-linear data generating processes.

The potential benefits of making standard statistical models deeper improves the flexibility of the model to fit a greater variety of functions, enabling the construction of a better model. With the standard Mixtures of Factor Analysers and their link to Mixture of Gaussians, we discuss the inclusion of latent layers into the standard Factor Analytic representation, thereby generating a hierarchical model that enables much greater power to fit both simple and complex processes.

## 1.2 Chapter Summary

The aim of this chapter is to introduce the reader to the motivations of the work and provide direction along a defined trajectory. This chapter begins with a summary of the chapters in order of presentation and an overview of the field thus far.

Chapter 2 gives a thorough and in-depth explanation of Deep Gaussian Mixture Models by viewing it as part of an array of methods used for statistical modelling. The main focus on modelling involves the use of Factor Analytic variants: the standard Factor Analysis model, the Mixtures of Factor Analysers, the Mixtures of Common Factor Analysers, and other deep variants leading up to the state of the art model in Deep Gaussian Mixture Models.

Chapter 3 focuses on the novel development and main contribution of this Thesis: the introduction of Common Factors in the Deep Gaussian Mixture Model when viewed as a Factor Analytic model. It is introduced first as a naive approach of simple stacking of the original Mixture of Common Factor Analyser model. Then it is shown that this model lacks flexibility in terms of the original Deep Gaussian Mixture Model that had a greater number of parameter components. Thus, two new models are introduced - the first model having Common Factor components based on the mixtures at each layer, and the second model having only an initial layer of Common Factors, followed by the original Deep Gaussian Mixture Model.

Chapter 4 will develop the methods to assess the performance on identical datasets as given by the original paper on Deep Gaussian Mixture Models. Furthermore, new datasets will be introduced on single-cell RNA-seq data, a development on the standard RNA-seq data on gene expression profiling for bulk cell populations. There is a need for more advanced techniques that allow clustering of cell-based data to discover and annotate newly discovered cells, aiding in the development of new therapeutic targets and further research [3]. With the introduction of the models in Chapter 3, the factor scores can now be projected to a common plane, thus allowing a unified statistical approach to dimensionality reduction, visualisation, and clustering.

Chapter 5 will show results as part of three different sections. Firstly, the section on Quality Control highlights the convergence of each Expectation Maximisation based algorithm and shows that the methods are able to pick out true signals. Secondly, the performance in unsupervised clustering will be provided on the benchmark datasets in Viroli and McLachlan [4], and several single-cell gene expression datasets. Thirdly, and lastly, the visualisation of the new designs will be shown, aiming to highlight the relevance of the new methods in conjunction with state-of-the-art visualisation methods.

Chapter 6 will discuss the results in retrospect with the Deep Gaussian Mixture Model, and other state-of-the-art visualisation methods. Future work will be outlined, detailing improvements that can be made.

This will finally be followed by Chapter 7 to conclude this Thesis.

# Chapter 2

# Modelling with Factor Analytic representations and Extensions

## 2.1 Statistical Modelling

Statistical modelling is a paradigm with a focus on capturing the complexity of an observable system in order to estimate a set of parameters that have properties associated to the governance of the data generating process in both the observable and latent states. The estimated properties of the system captured by the parameters enable explanation of the process through which the data is generated and prediction of novel observations determined by the model [5].

It had been considered for a model to be valued, the greater the 'fit' through less intervening decisions by the practitioner on the modelling choices that aim to embody the mechanics and operations of the system, it is more useful to consider fewer parameters. This also enables greater interpretability of the underlying system to capture the most important information from the data generating process with the fewest number of parameters as possible. With the development of Deep Learning, mostly through Neural Networks, there has been tremendous innovation and breakthroughs in the area of predictive and generative modelling, although their use as a black-box machine learning tool through the optimisation of the loss function has made it difficult to interpret [2].

There has been a strong push for greater interpretability in the algorithms used in practice in many of the systems that interface with our lives, both at work, and, in research. Neural Networks interface with our general every-day lives through applications such as face and voice recognition, and in recommendation systems. On the other hand, while the field of genetics has been dominated by linear models for their usefulness of interepretable properties and fit as a well-rounded model, transcriptomics (gene expression) is still open as the research field has yet to construct a biological functional explanation as to how genes work and interact [3].

A statistical modelling approach to best capture the underlying data generating process involves assumptions on the fixed deterministic and random variable components of the proposed model fitting the system. As stated by Konishi  Kitagawa [5] the three main aims of statistical modelling are:

1. The level of fit as a model, which considers stochastic variables in the data generating process.
2. The ability to predict novel observations, not observed from the prior samples.
3. The extraction of information and properties, enabling greater interpretation of the system through evaluation of parameters.

Furthermore, according to the information criterion as outlined by Konishi and Kitgaram [5] having fewer free parameters increases the number of degrees of freedom of the model, improving its' stability and prediction ability. It is therefore ideal to consider the number of free parameters as a way to compare similar model structures. Model based statistical comparisons such as the Akaike Information Criterion and the Bayesian Information Criterion both include calculation of the degrees of freedom of the statistical model as a way to compare different models and comparative fit. Both criterion also make use of the log-likelihood, a probabilistic measure indicative of the performance of the statistical model given the parameters.

$$AIC = 2k - 2ln(\hat{L})$$

$$BIC = ln(n)k - 2ln(\hat{L})$$

Here, k is the number of free parameters in the model, related to the difference between the total number of feasible parameters (n, representative of the sample size) and the degrees of freedom. The expectation of the conditional likelihood is represented by L, the likelihood from when the model obtains convergence after achieving an optimal state [6].

In order to estimate the likelihood, the most common technique used as a frequentist (in the statistical sense) is Maximum Likelihood Estimation. Through formulation of the model as a function of the parameters and data samples, as well as compartmentalising the observations being attributed to both known and unknown statistics, thereby derives the maximum likelihood at which the parameters are 'most likely' to have generated the idealised modelled data process.

## 2.2 Maximum Likelihood Estimation

There are several approaches that can be taken to estimate the parameters through Maximum Likelihood Estimation (MLE): Likelihood based methods and Gradient-based methods.

Of most well known is the Expectation Maximisation (EM) algorithm, a likelihood based method that reformulates the problem into two steps - the Expectation of the conditional log-likelihood, and the Maximisation of the likelihood through calculation of parameters via the conditional expectation. Through several iterations of this process, the likelihood of the model improves monotonically, and as an optimisation algorithm, it can converge to find solutions even on non-convex problems [1]

However, one issue yet to be addressed in the field is the issue of potential identification of local maxima, rather than global maximum estimates of the parameters. As a method, it has proven to be successful in a variety of situations, from: the Viterbi Algorithm in Hidden Markov Models to the training of Neural Networks and Hierarchical Mixtures of Experts [1].

For a formal introduction to the EM algorithm, we take a given observed sample dataset $X$, and a set of unobserved latent variables $Z$, with the vector of unknown parameters of interest $\theta$. The MLE of any statistical model which generates the observed data can be calculated via the EM algorithm in the following manner:

Let the likelihood function be:

$$L(\theta;X) = p(X|\theta) = \int_Z p(X,Z|\theta)dz$$

Through iterating across the two steps as given:

Expectation step (E step): Compute the expected log likelihood function

$$E_{Z|X,\theta^t}(logL(\theta;X,Z)) = \int_Z p(Z|X,\theta^t)logL(\theta;X,Z)dz$$

Maximisation step (M step): Find parameters by optimising over the expected log likelihood function by computing

$$argmax_{\theta^t}E_{Z|X,\theta^t}$$

To allow for improved convergence properties, several variants of the EM algorithm can be introduced, including the Monte Carlo EM (MCEM) algorithm, and the Expectation Conditional Maximisation (ECM) algorithm, where the MCEM and ECM algorithms will be discussed later [1].

Gradient-based methods for MLE often include Newton's second order method. These involve solving the linear Taylor series expansion about the current estimate of $\theta^t$, giving the following expression:

$$S(y;\theta) \approx S(y;\theta^t) - I(\theta^t;y)(\theta - \theta^t)$$

Through solving the likelihood equation for $S(y;\theta) = 0$, the updated estimate becomes

$$\theta^{t+1} = \theta^t + I^{-1}(\theta^t;y)S(y;\theta^t)$$

It should be noted that when the likelihood function is concave and unimodal, then the iterations over $\theta^t$ converges to the true MLE, however, if it is not concave, then there is no guarantee to converge to the global optima [1].

One issue with statistical models is that of identifiability, when two or more parameterizations of the model are equivalent and indistinguishable from another despite the difference in parameter estimates. In the example of utilising the EM algorithm to calculate the mixture of two Gaussian distributions, it is possible (and occurs frequently), that the parameters identifying the probability of occurrence for each of the two mixtures swap. Nevertheless, through careful consideration of the structure of these parameters, one can restructure the equations and add further constraints to make the model identifiable. In the example of the mixture of two Gaussian distributions, one can make a parameter constraint that it must be greater than the other [1].

## 2.3 Factor Analysis

A thorough example for the MLE of parameters for the standard Factor Analytic (FA) model will now be given. A FA model is used to explain the variables of interest, by decomposing them into a set of lower dimensional features. The observed variables in the sampled data are reduced down to a set of correlated latent unobservable factors [1].

Whereas other methods such as Principal Component Analysis seeks to find a set of linearly uncorrelated features that explain the most of variance in each orthogonal direction, FA attempts to find the most correlation among a set of variables by factoring the observations according to correlated latent factors. That makes FA more advanced compared to PCA, particularly in terms of normalising the density modelled around each point, such that the newly produced correlated latent factors attempt to explain more information than what can be produced by a set of linear orthogonal projected principal components.

The standard FA model is given an intercept $\mu$ and an error term $e_j$ which is orthogonal to the latent variables $z_j$, known as the factors, which are modelled as a set of Gaussian random variables with parameters B, known as the factor loadings. The expression is given as:

$$Y_{i,j} = \eta + \Lambda z_j + e_j$$

The random variables $z_j$ are $N(0, I_q)$ (where $I_q$ is the identity matrix of dimension $q \cdot q$ ) independently of errors $e_j$, assumed to be independent identically distributed as $N(0, \Psi)$ where $\Psi$ is a diagonal matrix.

Once $Y_{ij}$ is conditioned on $z_j$, the distribution of the factor analytic model becomes $N(\eta + \Lambda z_j, \Psi)$, whereas the unconditional distribution of $Y_{ij}$ is given as $N(\eta, \Lambda\Lambda^T + \Psi)$.

One can consider taking the full expression of the FA model (here indexing is dropped for convenience of notation):

$$Y = \eta + \Lambda z + e$$

Through rearranging the equation,

$$\Lambda(Y - \eta) = \Lambda\Lambda^T z + \Lambda e$$

Noticing that $\Lambda e$ is orthogonal, and thus remains as a Gaussian error term $e$, in addition to the fact that $f(z) = N(0, I_q)$, we can take the following step:

$$z = (\Lambda\Lambda^T + \Psi)^{-1}\Lambda(Y - \eta)$$

In fields such as Psychology and Bioinformatics, the calculated $a$ are known as factor scores and when transformed such that they are projected onto the same dimension, the dataset can be reduced from a high-dimensional space to a two-dimensional space to be plotted on the standard Cartesian plane.

Finally, from the factor scores one can calculate the predicted observation, as depicted by the FA model:

$$\hat{y}_{ij} = \sum_{\tilde{i}} (\hat{\eta}_i + \hat{\Lambda}_{\tilde{i}} \hat{z}_j)$$

The free parameters given by the standard FA model [1] is given as:

1. $\Lambda$ has $p \cdot q$ parameters
2. $\Psi$ has $p$ parameters
3. Given $\Lambda\Lambda^T$, there are $\frac{1}{2}q \cdot (q-1)$ less parameters, as this reflects the upper-diagonal part of the symmetric matrix $\Lambda\Lambda^T$, that is required to remain unique in the parameterisation of the FA model.

There are thus, $d_1 = (pq + p - \frac{1}{2}q(q-1))$ parameters in total

**Algorithm 1:** Learning Factor Analysers [7]

```
function FA
Input   :Parameters $\theta : \{\eta, \Lambda, \Psi\}$
Output : Optimised parameters $\theta$
    1. In the E-step of the EM algorithm [8], first compute:
           $\beta = \Lambda^T (\Psi + \Lambda\Lambda^T)^{-1}$
       to find,
           $E[z_j | y_{ij}] = \beta y_{ij}$
       and,
           $E[z_j z_j^T | y_{ij}] = I - \beta\Lambda + \beta y_{ij} y_{ij}^T \beta^T$
    2. Then, in the M-step of the EM algorithm, compute:
           $\Lambda^{\{t+1\}} = (\sum_{ij} (z_j E[z_j | y_{ij}]^T))(\sum_{ij} E[z_j a_j^T | y_{ij}])^{-1}$
           $\Psi^{\{t+1\}} = \frac{1}{n} diag\{\sum_{ij} y_{ij} y_{ij}^T - \Lambda^{t+1} E[z_j | y_{ij}] y_{ij}\}$
```

## 2.4 Mixtures of Factor Analysers

Given the linearity of the specified model by FA, it is unable to deal with non-linear data distributions. To accommodate for this, a non-linear alternative to FA can be introduced, that is, Mixtures of Factor Analysers (MFA). Through use of mixtures, MFA can approximate non-linear observations by making local linear estimations of the functional data generating process [8].

Unlike Principal Component Analysis, through modelling the density as Gaussians via a factor analytic representation, the FA methods and variants do not encode each data point as equal anywhere along the principal component space. By considering the density of each data point, FA normalises the points on a global scale, and MFA further improves this by normalising the density locally [8].

It should be noted that the factors are taken to be Gaussian with covariance matrix equivalent to the identity $N(0, I_q)$. As such, these are not true projections, and as later discussed, do not not add a layer of interpretability as more advanced methods are able to do [7].

As an expression, there are $g_k$ mixtures of the standard multivariate normals of the Factor Analytic form $f(g_k) = \pi_{g_k}$. The factors conditioned on the mixtures are of the form $f(z_j|g_k) = f(z_j) = N(0, I_q)$. Here, the parameters used in the previous explanation of the FA model are used:

1. Sampled observations: $y$
2. Estimated observations: $\hat{y}_j$
3. Diagonal Covariance error terms: $\Psi$
4. Factors: $z_j$
5. Factor Loadings: $\Lambda$
6. Intercept: $\eta$

The new parameters in the MFA model are:

1. The mixture probabilities: $\pi_{g_k}$

The parameterisation of the conditional MFA given both the mixtures and factors is given as:

$$f(y_{ij}|z_j, g_k) = N(\eta_{g_k} + \Lambda_{g_k} z_j, \Psi_{g_k)}$$

The unconditional probability of $y$ given $g_k$ is defined as:

$$f(y|g_k) = \int_z f(y|z, g_k) f(z|g_k) dz = N(\eta_{g_k}, \Lambda_{g_k} \Lambda_{g_k}^T + \Psi_{g_k}) \ \ with \ \ prob \ \ \pi_{g_k}$$

where $g_k = g_1, g_2, ..., g_{l-1}, g_l$

Notice, by integrating the conditional MFA with respect to the latent factor variables, the factor loadings within the mean of the conditional are carried across into the variance term in the unconditional density distribution.

Alternatively, one can take $Z$ as the standard normal distribution, and carry out the computation as follows with the random variables;

$$Y_{ij}|z_j, g_k = \eta_{g_k} + \Lambda_{g_k} z_j + \Psi_{g_k}^{\frac{1}{2}} Z$$

For completeness, one can consider taking the full expression of the MFA model:

$$Y = \eta_{g_k} + \Lambda z + e$$

Through rearranging the equation,

$$\Lambda_{g_k}(Y - \eta_{g_k}) = \Lambda_{g_k}\Lambda_{g_k}^T z + \Lambda_{g_k} e$$

Noticing that $\Lambda_{g_k} e$ is orthogonal, and thus remains as a Gaussian error term $e$, in addition to the fact that $f(z|g_k) = f(z) = N(0, I_q)$, we can take the following step:

$$z = (\Lambda_{g_k}\Lambda_{g_k}^T + \Psi_{g_k})^{-1}\Lambda_{g_k}(Y - \eta_{g_k})$$

From this, one can calculate the predicted observation, as depicted by the MFA model:

$$\hat{y}_{ij} = \sum_{\tilde{g_k}} \pi_{\tilde{g_k}}(\hat{\eta}_{\tilde{g_k}} + \hat{\Lambda}_{\tilde{g_k}} \hat{z}_j)$$

The free parameters given by the standard MFA model [7] is given as:

1. $\Lambda$ has $p \cdot q$ parameters
2. $\Psi$ has $p$ parameters
3. Given $\Lambda \cdot \Lambda^T$, there are $\frac{1}{2}q \cdot (q-1)$ less parameters, as this reflects the upper-diagonal part of the symmetric matrix $\Lambda \cdot \Lambda^T$, that is required to remain unique in the parameterisation of the Factor Analytic model.
4. Finally, for each of the $g$ mixtures, there are $g$ many parameterisations of the covariance matrix $\Lambda \cdot \Lambda^T + \Psi$.
5. In summary, given g mixtures, there are also a further $g-1$ parameters, one less than g as it is possible to know all $g$ mixtures given $g-1$ parameters.
6. The intercept parameters must be included, involving $g.p$ more parameters, one for each mixture

There are thus, $d_1 = (g-1) + 2gp + g(pq - \frac{1}{2}q(q-1))$ parameters in total

Alternatively, other methods to estimate the parameters of the FA models and variants include the method of moments, minimum distance, gradient and non-gradient methods (such as with the Covariance Matrix Adaptation Evolution Strategy).

**Algorithm 2:** Learning Mixtures of Factor Analysers [8]

```
function MFA
Input  :Parameters $\theta : \{\eta_{g_k}, Lambda_{g_k}, \Psi_{g_k}, \tau_{ij}\}$
Output : Optimised parameters $\theta$
```

1. In the E-step of the EM algorithm [8] , first compute for each mixture:

$$\pi_g = E(g_k|y_{ij})$$

and,

$$\beta = \Lambda^T(\Psi + \Lambda\Lambda^T)^{-1}$$

to find,

$$E[z_j|y_{ij}, m] = \beta y_{ij}$$

and,

$$E[z_j z_j^T|y_{ij}, g_k] = I - \beta\Lambda + \beta y_{ij} y_{ij}^T \beta^T$$

2. Then, in the M-step of the EM algorithm, compute:

$$[\Lambda_k|\eta_k] = (\sum_i y_{ij} E(g_k|y_{ij}) E[\tilde{z_j}|y_{ij}, g_k]^T).(\sum_l E(g_k|y_{ij}).[E[\tilde{z_j}\tilde{z_j}^T|y_{ij}, g_k])^{-1}$$

where,

$$E[\tilde{z_j}|y_{ij}, g_k] = \begin{bmatrix} E[z_j|y_{ij}, g_k] & 1 \end{bmatrix}$$

and,

$$E[\tilde{z_j}\tilde{z_j}^T|y_{ij}, g_k] = \begin{bmatrix} E[z_j z_j^T|y_{ij}, g_k] & E[z_j|y_{ij}, g_k] \\ E[z_j|y_{ij}, g_k] & 1 \end{bmatrix}$$

3. Thus, to find the diagonal variances of the error term:

$$\Psi = \frac{1}{n} diag \sum_i E(g_k|y_{ij})(y_{ij} - [\Lambda_k|\eta_k] E[\tilde{z_j}|y_{ij}, g_k]) y_{ij}^T$$

And re-estimation of the mixing probabilities, gives:

$$\pi_{g_k} = \frac{1}{n} \sum_{i,j,l \in g} \tau_{i,j,l}$$

For modelling and inferencing purposes (and for the calculation of the Aikaike and Bayesian Information Criterion) there are several parameters that the user must decide prior to the estimation of the learnable parameters from the data: the number of mixtures to use $g$ and the number of factors in each analyser $(r)$. Furthermore, the free parameters for the MFA model should be stated.

If the dimensionality of the dataset is large, or the number of components are relatively large, then the number of parameters are unlikely to be manageable [7]. It is for this reason that more advanced approaches have been taken to improve the model and reduce the number of free parameters such as will be introduced in the next section: Mixtures of Factor Analysers with Common Factor Loadings [7].

## 2.5 Mixtures of Common Factor Analysers

The MFA model with its' standard Gaussian distributed latent factor variables does not make it easy for high dimensional datasets to be visualised in a lower dimension, such as in a two-dimensional plane in a Cartesian plot. That is, there requires an extra constraint to restrict both the intercept $\mu_g$ and covariance $\Sigma_i$ such that there is a common matrix $A$, acting as a common projection between factor loadings [7].

Through the matrix projection A, the following constraints are required:

$$\mu_{g_k} = A\xi_{g_k} \quad (g_k = g_1, ..., g_l)$$

$$\Sigma_{g_k} = A\Omega_{g_k}A^T + D \quad (g_k = g_1, ..., g_l)$$

We now show that the matrix transform A with the additional constraints, puts the factor loadings on the same axes, thereby allowing their visualisation on the two axes of a Cartesian plane.

Firstly, by taking

$$Y_{ij} = \eta_{g_k} + \Lambda_{g_k} z_j + e_j \quad with \;\; prob \;\; \pi_{g_k} \quad (g_k = g_1, ..., g_l)$$

And letting $\Lambda_{g_k} = AK_{g_k}$, and $z_j = K_{g_k}^{-1}(\tilde{z}_j - \xi_{g_k})$, where the $K_{g_k}$ is chosen such that, $K_{g_k}^{-1}\Omega_j K_{g_k}^{-1T} = I_q; \quad (g_i = g_1, ..., g_l)$ where $\tilde{z}_j$ is a standard Gaussian of $N(\xi_{g_k}, \Omega_{g_k})$.

$$Y_{ij} = \mu_{g_k} + AK_{g_k}K_{g_k}^{-1}(\tilde{z}_j - \xi_{g_k}) + e_j$$

Notice a key important construct is that of $\Lambda_g = AK_{g_k}$, allowing a common matrix projection on each mixture - based on the condition that the projection $A$ is common for all mixtures. Furthermore the matrix transform $K_{g_k}$ acts as a scale, transforming the random variable $\tilde{z}_j$ into a standard normal, of the form $z_j = K_{g_k}^{-1}(\tilde{z}_j - \xi_{g_k})$. This occurs as a result of the definition of $K_{g_k}$ being the standard deviation of $\tilde{z}_j$, as defined in: $K_{g_k}^{-1}\Omega_{g_k}K_{g_k}^{-1T} = I_q$, where $\tilde{z}_{gj}$ is a standard Gaussian of $N(\xi_{g_k}, \Omega_{g_k})$.

With the constraints given above, $\mu_{g_k} = A\xi_{g_k}$

$$Y_{ij} = A\xi_{g_k} + A(\tilde{z}_j - \xi_{g_k}) + e_j$$

Giving the expression:

$$Y_{ij} = A(\tilde{z}_j) + e_j$$

Which is known as the general expression for the Mixtures of Factor Analyzers with Common Factor Loadings (MCFA). Thus the MCFA approach can be identified as a special case of the MFA model with additional restrictions:

$$\mu_{g_k} = A\xi_{g_k}$$

$$\Lambda_{g_k} = AK_{g_k}$$

$$\Psi_{g_k} = D$$

It is evident that the latent factors are of a standard Gaussian in MFA, and a transformed Gaussian in MCFA. Thus, in MFA:

$$a_{ij} = K_i^{-1}(\tilde{a}_{ij} - \xi_i)$$

Here, $a_{ij}$ is a standard Gaussian, now that it has been translated by $\xi_i$ and scaled by $K_i^{-1}$. Whereas in MCFA:

$$\tilde{a}_{ij} = a_{ij}K_i + \xi_i$$

where,

$$W_g = A(\tilde{a}_{gj}) + e_{gj}$$

The latent factors in MCFA can be seen to be a translated and scaled form of the standard Gaussian density distribution.

Furthermore, to calculate the common factor scores for plotting onto the same projected plane, the following form using the parameters can be taken:

$$U_j = \xi_{s_1} + \gamma_{s_1}(y - A\xi_{s_1})$$

where,

$$\gamma_{s_1} = (A\Omega_{s_1}A^T + D_{s_1})^{-1}A\Omega_{s_1}$$

The free parameters given by the standard MCFA model [7] is given as:

1. A has $pq - q^2$ parameters
2. D has $p$ parameters
3. Given $A \cdot \Omega \cdot A^T$, there are $\frac{1}{2}(q^2 + q)$ parameters, as this reflects the upper-diagonal part of the symmetric matrix $\Omega_i$, that is required to remain unique in the parameterisation of the Factor Analytic model. Additionally, given $\Omega_i$ occurs for each mixture, there are $g\frac{1}{2}(q^2 + q)$ parameters in total.
4. In summary, given g mixtures, there are also a further $g - 1$ parameters, one less than g as it is possible to know all $g$ mixtures given $g - 1$ parameters.

**Algorithm 3:** Learning Mixtures of Common Factor Analysers [7]

```
function MCFA
Input  : Parameters $\theta$ : $\{A, \xi_{g_2}, \Omega_{g_2}, D\}$
Output : Optimised parameters $\theta$
    1. For each iteration (t), conduct the following:
    2. In the E-step of the algorithm [7], first compute for each mixture:
```

$$\pi_g^{(t+1)} = E(g_k|z_j) = \sum_j \frac{\tau_{ij}^t}{n}$$

and,

$$\beta^{(t)} = (A^{(t)}\Omega_{g_k}^{(t)}A^{(t)T} + D^{(t)})^{-1}A^{(t)}\Omega_{g_k}^{(t)}$$

to find,

$$E(z_j|y_{ij}, g_k) = \beta^{(t)T}y_{ij}^{(t)}$$

```
    3. In the M-step of the algorithm, now compute:
```

$$\xi_{g_k}^{(t+1)} = \xi_i^{(t)} + \frac{\sum_j \tau_i j^k E(z_j|y_{ij}, g_k)}{\sum_j \tau_{ij}^k}$$

$$\Omega_i^{(t+1)} = \frac{\sum_j \tau_{ij}^k E(z_j|y_{ij}, g_k)E(z_j|y_{ij}, g_k)^T}{\sum_j \tau_i j^k} + \Omega_{g_k}^{(t)}(I_q - \beta_{g_k}^{(t)T}A^{(t)})$$

Then,

$$D^{(t+1)} = diag(D_1^{(t)} + D_2^{(t)})$$

where,

$$D_1^{(t)} = \frac{\sum_i \sum_j \tau_{ij}^{(t)} D^{(t)}(I_p - \gamma_{g_k}^{(t)})}{\sum_i \sum_j \tau_{ij}^{(t)}}$$

$$D_2^{(t)} = \frac{\sum_i \sum_j \tau_{ij}^{(t)} \gamma_{g_k}^{(t)T} y_{ij}^{(t)} y_{ij}^{(t)T} \gamma_{g_k}^{(t)}}{\sum_i \sum_j \tau_{ij}^{(t)}}$$

here,

$$\gamma_{g_k}^{(t)} = (A^{(t)}\Omega_{g_k}^{(t)}A^{(t)T} + D^{(t)})^{-1}D^{(t)}$$

Finally,

$$A^{(t+1)} = (\sum_i A_{1i}^{(t)})(\sum_i A_{2i}^{(t)})^{-1}$$

where,

$$A_{1i}^{(t)} = \sum_j \tau_{ij}^{(t)}(w_j \xi_{g_k}^{(t)T} + w_j \gamma_{g_k}^{(t)T})$$

$$A_{2i}^{(t)} = \sum_j \tau_{ij}^{(t)}(I_q - \gamma_{g_k}^{(t)T}A^{(t)})\Omega_{g_k}^{(t)} + r_{g_k}^{(t)} r_{g_k}^{(t)T}$$

here,

$$r_{g_k}^{(t)} = \xi_{g_k}^{(t)} + \gamma_{g_k}^{(t)T}y_{ij}^{(t)}$$

5. The location parameter from the latent variable must be included, involving $g.q$ more parameters, one for each mixture

There are thus, $d_1 = (g-1) + p + q(p+g) + \frac{1}{2}gq(q+1) - q^2$ parameters in total

Furthermore, in high-dimensional datasets, there is need for a reduction in the number of free parameters, especially when the number of variables is close to the sample size. The MCFA approach is able to reduce the number of parameters through the common factor projections when compared to MFA [7].

Next, it will be shown that by taking the latent factors as another layer in factor analytic form, the model becomes better at picking up the nuances in the data to reflect the non-linearities present between the data points, both at a local and global perspective in a hierarchical structure.

## 2.6 Deep Mixtures of Factor Analysers and Extensions

### 2.6.1 Deep Mixtures of Factor Analysers

Given that MFA already compartmentalises the sampled observations from the data generating process into clusters via mixture modelling, another factor analytic layer set within the initial latent factors greatly improves the ability to model a larger array of both linear and non-linear systems.

The first layer, represented by the blue and red clusters are composed of another set of clusters. When depicted together, they show a non-linear structure. The idea is then to compartmentalise these non-linear structures into smaller sub-parts defined by another layer of MFA, giving the newer, more advanced model the ability to capture greater amounts of non-linearities [9].

The Deep Mixtures of Factor Analysers (DMFA) algorithm is quite simple to understand - it requires computation of two layers of MFA, the first layer giving inputs into the next layer. To begin learning unsupervised the clusters, train a MFA as the first step (the EM algorithm can be seen in the setup given in the above section on MFA) [9].

Once the first layer of MFA has been trained, create another dataset, sampled from the latent factors, thus deriving a set of factor scores for each mixture of the MFA, thus giving $m$ datasets, where $m$ is the number of mixtures.

In terms of the formulaic expression, the observed data is given as $y$

$$p(y|z^{(1)}, g_1) = N(y, \eta_{g_1} + \Lambda_{g_1}^T z^{(1)}, \Psi_{g_1})$$

where the factor analytic expression is given as:

$$y = \eta_{g_1} + \Lambda_{g_1}^T z^{(1)} + e_{g_1}$$

where $e_{g_1} \sim N(0, \Psi_{g_1})$. Here, $z^{(1)}$ is given as:

$$z^{(1)} = \eta_{g2} + B_{g2}^T z^{(2)} + e_{g2}$$

where $z^{(2)} \sim N(0,I)$

Now, another layer of MFA is trained from the sampled $z^{(1)}$ allowing a greater number of sub-mixtures to be formed from each of the first set of mixtures in the initial layer. This requires fixing the parameters in the first layer to sample from $z^{(1)}$ to generate the inputs into the next layer, as follows:

**Algorithm 4:** Learning Deep Mixtures of Factor Analysers

```
function DMFA
Input  : Parameters $\theta : \{\eta_{g_k}, \Lambda_{g_k}, \Psi_{g_k}\}$
Output : Optimised parameters $\theta$
    1. Train first MFA layer on $y$ to identify first layer parameters $\eta_{g_1}, \Lambda_{g_1}, \Psi_{g_1}$ with $g_1$ mixtures and $r_1$ dimensions in the factors.
    2. Generate samples $z_1$ from factor analytic expression $z^{(1)} = \eta_{g_2} + \Lambda_{g_2}^T z^{(2)} + e_{g_2}$ by clustering the data into mixtures based on $p(g_1|y)$
    3. Train second MFA layer on top of $z_1$ to identify second layer parameters $\eta_{g_2}, \Lambda_{g_2}, \Psi_{g_2}$ with $g_2$ mixtures and $r_2$ dimensions in the factors.
```

Through extending the standard model of MFA, the latent variables can be further represented in a factor analytic form as follows:

$$y = \eta_{g_1}^{(1)} + \Lambda_{g_1}^{(1)} z^{(1)} + e_{g_1}$$

This is the standard form of a MFA. Now, when taking $z^{(1)}$ as another factor analytic representation, a new layer can be generated in a hierarchical structure giving:

$$z^{(1)} = \eta_{g_2}^{(2)} + \Lambda_{g_2}^{(2)} z^{(2)} + e_{g_2}$$

Where,

$$p(z^{(2)}|g_2) = N(z^{(2)};0,I)$$

Alternatively, one can write the Deep Mixtures of Factor Analysers (DMFA) as a probabilistic expression with the following distributions:

$$p(g_1) = \pi_{g_1}^{(1)}$$
$$p(y|z^{(1)}, g_1) = N(y; \eta_{g_1}^{(1)} + B_{g_1}^{(1)} z^{(1)}, \Psi_{g_1}^{(1)})$$

$$p(z^{(1)}|z^{(2)}, g_2) = N(z^{(1)}; \eta_{g_2}^{(2)} + \Lambda_{g_2}^{(2)} z^{(2)}, \Psi_{g_2}^{(2)}]$$

$$p(g_2) = \pi_{g_2}^{(2)}$$
$$p(z^{(2)}|g_2) = N(z^{(2)};0,I)$$

The free parameters given by the standard MFA model [7] is given as:

1. $\Lambda$ has $p \cdot q$ parameters
2. $\Psi$ has $p$ parameters
3. Given $\Lambda \cdot \Lambda^T$, there are $\frac{1}{2}q \cdot (q-1)$ less parameters, as this reflects the upper-diagonal part of the symmetric matrix $\Lambda \cdot \Lambda^T$, that is required to remain unique in the parameterisation of the Factor Analytic model.
4. Finally, for each of the $g$ mixtures, there are $g$ many parameterisations of the covariance matrix $\Lambda \cdot \Lambda^T + \Psi$.
5. In summary, given g mixtures, there are also a further $g-1$ parameters, one less than g as it is possible to know all $g$ mixtures given $g-1$ parameters.
6. The intercept parameters must be included, involving $g.p$ more parameters, one for each mixture
7. For the $c$ mixtures in the second layer, follow a similar procedure to the previous layer with $g$ mixtures with an initial dimension of $q$ and a factor dimension of $d$

There are thus, $d_1 = (g-1) + g(2p + pq - \frac{1}{2}q(q-1)) + c(2q + qd - \frac{1}{2}d(d-1))$ parameters in total

### 2.6.2 Deep Mixtures of Factor Analysers with Common Factor Loadings

Given the increased number of parameters through addition of a second factor analytic layer, it is of great importance to reduce the parameter space by including common factor loadings. This refers back to the MCFA model, with inclusion of a second layer similar to DMFA - thereby forming a Deep Mixtures of Common Factor Analysers (DMCFA) [10] [11] [12].

The additional benefits of DMCFA over DMFA is the ability to visualise the results given the common projection matrix. Similar to other techniques such as Principal Component Analysis, t-Stochastic Neighbourhood Embedding [13] and Uniform Manifold Approximation and Projection [14], common factor loadings under a form of Factor Analysis provides a dimensionality reduction, with the benefit of added clustering. Users must specify the number of clusters prior to the running of the algorithm, and then let the algorithm choose the model parameters (number of layers, dimension of factor loadings) through optimising the Maximum Likelihood of the model.

This begins by taking the first and second layer of the DMFA to be both modelled by two layers of MCFA. The first MCFA layer is learned, with the parameters in the first layer taken to the next layer by sampling from the latent factor variable. Through doing so, this allows the next layer to learn a deeper parameterisation, thereby enriching the model. The benefit of using MCFA over MFA is that the number of parameters reduces, thus reducing the complexity and improving the stability of the model [11].

Similar to DMFA, a layer by layer approach is taken. Firstly, the observed data is given as $y$

$$p(y|z^{(1)}, g_1) = N(y, A_1^{(1)T} z^{(1)}, D^{(1)})$$

here the factor analytic expression is given as:

$$y = A_1^{(1)T} z^{(1)} + e_{g_1}^{(1)}$$

where $e_{g_1}^{(1)} \sim N(0, D_{g_1}^{(1)})$.

Notice that this is identical to a factor analytic layer with common loadings. Now, another layer of MCFA is trained from the sampled $z^{(1)}$. Again, similar to DMFA this requires fixing the parameters in the first layer to sample from $z^{(1)}$ to generate the inputs into the next layer, as follows:

$$p(z^{(1)}|z^{(2)}, g_2) = N(y, A_{g_2}^{(2)T} z^{(2)}, D_{g_2}^{(2)})$$

where the factor analytic expression is given as:

$$z^{(1)} = A_{g_2}^{(2)T} z^{(2)} + e_{g_2}^{(2)}$$

here $e_{g_2}^{(2)} \sim N(0, D_{g_2}^{(2)})$.

Importantly, $z^{(2)}$ comes from a certain distribution that is not of the standard normal, that is to say, it comes from a Gaussian distribution of the following form:

$$z^{(2)} \sim N(\xi_{g_2}^{(2)}, \Sigma_{g_2}^{(2)})$$

**Algorithm 5:** Learning Deep Mixtures of Common Factor Analysers

```
function DMCFA
Input :Parameters $\theta : \{A^{(1)}, A_{g_2}^{(2)}, \xi_{g_2}^{(2)}, \Omega_{g_2}^{(2)}, D^{(1)}, D_{g_2}^{(2)}\}$
Output: Optimised parameters $\theta$
  1. Train first MCFA layer on $y$ to identify first layer parameters $A_{(1)}, D^{(1)}$ with $g_1$ mixtures and $r_1$ dimensions in the factors.
  2. Generate samples $z_1$ from factor analytic expression $z^{(1)} = A_{g_2}^{(2)T} z^{(2)} + e_{g_2}^{(2)}$ by clustering the data into mixtures based on $p(g_1|y)$
  3. Train second MCFA layer on top of $z_1$ to identify second layer parameters $A_{g_2}^{(2)}, D_{g_2}^{(2)}, \xi_{g_2}^{(2)}, \Omega_{g_2}^{(2)}$ with $g_2$ mixtures and $r_2$ dimensions in the factors.
```

The parameters $\xi_{g_2}^{(2)}; \Sigma_{g_2}^{(2)}$ are learned in conjunction with the parameters in the second layer, similar to that of MCFA, that is, $A_{g_2}^{(2)}, D_{g_2}^{(2)}, \xi_{g_2}^{(2)}, \Sigma_{g_2}^{(2)}$ are learned in the same M-step of the EM-algorithm.

As such, the full marginal density of $y$ is given as:

$$p(y|g_1) = N(A^{(1)} A_{g_2}^{(2)} \xi_{g_2}^{(2)}, A^{(1)}(A_{g_2}^{(2)} \Omega_{g_2} A_{g_2}^{(2)} + D_{g_2}^{(2)}) A^{(1)} + D^{(1)})$$

It is evident that the projection matrices $A^{(1)}$ and $A^{(2)}_{g_2}$ enable a simple visualisation of the first layer, given the common projection $A^{(1)}$. Within the second layer, there are separate projections for each mixture within the first layer $A^{(2)}_{g_2}$ enabling visualisations of the results, provided appropriate consideration is made for the mixtures in the first layer

One of the disadvantages of the DMFA and the DMCFA model is in it's limited capacity to combine the learning of the entire structure as one separate model instead of two separate parts - one model as the top layer and the second model as the bottom layer. Given the model trains the layers separately, there is lack of sharing of information during the learning phase between the parameters, preventing the model from being fit optimally. A model that takes into account the shared learning in the parameter space will be introduced as a Deep Gaussian Mixture Model in the following Chapter [4].

The free parameters given by the standard DMCFA model [7] is given as:

1. $A^{(1)}$ has $pq - q^2$ parameters
2. $D^{(1)}$ has $p$ parameters
3. Given $A^{(1)} \cdot \Omega \cdot A^{(1)T}$, there are $\frac{1}{2}(q^2 + q)$ parameters, as this reflects the upper-diagonal part of the symmetric matrix $\Omega^{(2)}_{g_1}$, that is required to remain unique in the parameterisation of the Factor Analytic model. Additionally, given $\Omega_i$ occurs for each mixture, there are $g\frac{1}{2}(q^2 + q)$ parameters in total.
4. In summary, given g mixtures, there are also a further $g - 1$ parameters, one less than g as it is possible to know all $g$ mixtures given $g - 1$ parameters.
5. For the $c$ mixtures in the second layer, follow a similar procedure to the previous layer with $g$ mixtures with an initial dimension of $q$ and a factor dimension of $d$
6. Remember, for each of the latent factors in the last layer, there are two extra parameters - the location and scale parameters in the Gaussian latent factor. This gives an extra $d$ parameters for each location parameter of the mixture $s$ and, an extra $\frac{d^2+d}{2}$ parameters in the scale parameter, giving $s(\frac{d^2+d}{2} + d) = s\frac{d^2+3d}{2}$ many parameters

There are thus, $d_1 = (g-1) + gp + pq - q^2 + c(2q + \frac{q(q+1)}{2} + qd - d^2) + c\frac{d^2+3d}{2}$ parameters in total

# Chapter 3

# A novel design for Deep Gaussian Mixture Models with Common Factor Loadings

## 3.1 Deep Gaussian Mixture Models

A richer model can be leveraged by taking advantage of more layers by jointly learning multiple layers of factor analysers. By altering the learning procedure and estimation of parameters, the algorithm is able to fit more than two layers (compared to that of DMFA) within what is called a Deep Gaussian Mixture Model (DGMM) [4]. Similar to the layers of a Neural Network, the depth of each layer composes a series of non-linear transformations, allowing this novel method to model more complex datasets.

The composition of each layers is similar to that of DMFA, however, there are some key differences. An important concept put forth by Viroli et. al. [4] is the notion of joint learning by Maximum Likelihood in the parameter space, thereby circumnavigating around the issue of fixing each layer once parameters have been learned, as in DMFA and DMCFA. It does this by expressing the DMFA under a conditional distribution to learn the parameters layer by layer through sampling the conditional distributions in a stochastic Monte-Carlo E-step, then learning the parameters in the M-step. Through cycling this stochastic-EM algorithm, the parameters are learned together in a series of iterations, forming a greater whole, as follows:

### 3.1.1 Model Overview

$$y = \eta_{s_1} + \Lambda_{s_1} z^{(1)} + e_1$$

$$z^{(1)} = \eta_{s_2} + \Lambda_{s_2} z^{(2)} + e_2$$

$$z^{(2)} = \eta_{s_3} + \Lambda_{s_3} z^{(3)} + e_3$$

$$z^{(3)} \sim N(0, I)$$

$$e_i \sim N(0, D_{s_i})$$

where $s_k = s_1, s_2, s_3$, with k representing the layer index (total number of layers is $h = 3$ in this example). Thus, for each $s_i \in s_k$, there is $s_i = g_1, g_2, ..., g_{l-1}, g_l$, where $g_l$ represents the different l mixture components in layer i of $s_i$.

### 3.1.2 Formulating the Marginal

The full form of $p(z^{(i)})$ is given by:

$$p(z^{(i)}) = N(\tilde{\mu}, \tilde{\Sigma})$$

$$\tilde{\mu} = \eta_{s_1}^{(1)} + \Lambda_{s_1}^{(1)}(\eta_{s_2}^{(2)} + \Lambda_{s_2}^{(2)}(...(\eta_{s_1}^{(1)} + \Lambda_{s_{h-1}}^{(h-1)}\eta_{s_h}^{(s_h)})))$$

$$\tilde{\Sigma} = \Psi_{s_1}^{(1)} + \Lambda_{s_1}^{(1)}(\Lambda_{s_2}^{(2)}(...(\Lambda_{s_h}^{(h)}\Lambda_{s_h}^{(h)T} + \Psi_{s_h}^{(h)})...)\Lambda_{s_2}^{(2)T})\Lambda_{s_1}^{(1)T}$$

where the form of $y$ is given by $z^{(0)}$.

Notice, now compared to the shallower standard Factor Analytic representations, the full model of the function of the observations, $y$ is a rich set of transformations of the final layer, where the final latent variable is represented by a standard normal. This supports the notion that the subsequent layers are capable of modelling functions that are of greater complexity.

### 3.1.3 Formulating the Posterior

The posterior can be constructed from the following densities:

$$p(z^{(i-1)}|z^{(i)}) = N(\eta_{s_l} + \Lambda_{s_l}z^{(i)}, \Psi_{s_l})$$

where,

$$z^{(i-1)}|z^{(i)} = \eta_{s_l} + \Lambda_{s_l}z^{(i)} + e_i$$

$$p(z^{(i)}) = \int\int p(z^{(i)}|z^{(i+1)})p(z^{(i+1)}|z^{(i+2)})p(z^{(i+2)})dz^{(i+1)}dz^{(i+2)}$$

where the latter formulation can be continued indefinitely to form a deeper model of Gaussian Mixtures.

From here, using the above formulations, one can construct the posterior to sample the conditional expectation in the stochastic step of the Stochastic Expectation Maximisation (S-EM) algorithm:

$$p(z^{(i)}|z^{(i-1)}) \propto p(z^{(i-1)}|z^{(i)}).p(z^{(i)})$$

where in the DGMM $p(z^{(i)}|z^{(i-1)})$ has the form based off the above expressions given by $p(z^{(i)})$ and $p(z^{(i-1)}|z^{(i)})$:

$$p(z^{(i)}|z^{(i-1)}) = N(\rho_{(s_l)}.(z_{(s_{l-1})}), \xi_{s_l})$$

where $\tilde{\Sigma}_{s_l}^{l+1}$ and $\tilde{\mu}_{s_l}^{l+1}$ are taken from the marginals,

$$\rho_{(s_l)} = \xi_{s_l}((\Lambda_{s_l}^l)^T(\Psi_{s_l}^l)^{-1}(z^{l-1} - \eta_{s_l}^l) + \tilde{\Sigma}_{s_l}^{l+1}\tilde{\mu}_{s_l}^{l+1})$$

$$\xi_{s_l} = (\tilde{\Sigma}_{s_l}^{l+1})^{-1} + (\Lambda_{s_l}^l)^T(\Psi_{s_l}^l)^{-1}\Lambda_{s_l}^l$$

### 3.1.4 Initialising the model

The initial observations $y$, are subset into clusters by the k-nearest neighbour algorithm to form $y_{g_k}$, and used to train k separate standard Factor Analysers for each $g_k$. Then for each layer, the latent factor scores are sampled according to this model to form $z_{g_l}$, whereby the same process enacted on $y_{g_k}$ is carried out on the $z_{g_l}$, iterating again for each layer until all parameters are eventually initialised [4].

### 3.1.5 Maximum Likelihood Estimation

Given the normalisation constant can be dealt with fairly easily in the above Bayes formulation, one can find the stochastic Gaussian random variable to sample from by finding the posterior, completing the S-E step of the S-EM algorithm.

Afterwards, the M-step aims to find the parameters. Utilising $p(z^{(i-1)}|z^{(i)})$, one can find the parameters $\Theta_{s_l}^t = \Lambda_{s_l}^t, \Psi_{s_l}^t, \eta_{s_l}^t, \pi_{s_l}^t$ using a Maximum Likelihood approach, as given by the parameters in Algorithm 6.

For a description of the parameters, model set-up, and an example implementation, see Chapter 4.2.

The difficulties with the stochastic EM algorithm has been well studied. It suffers from stability issues under certain conditions related to $n$, the sample size. Furthermore, as a result of constant sampling it is relatively slow compared to other methods which does not require stochastic generation of new data points under a given density distribution. As a result, the parameters must be tuned according to the likelihood converging towards its' optima (as this is the EM algorithm, it is not guaranteed to be the global optima) [1].

There are also considerations that need to be made with regards to the identifiability of the model. Firstly, the model identifiability needs to be considered in Deep Models - for example, a deep model with a layer-by-layer structure composed of 2 mixtures, and finally 3 mixtures may not be identifiable compared to one of 3 mixtures, followed by 2 mixtures. This can be solved by forcing the model to have progressively decreasing dimensions thereby enforcing a different parameterisation for each mixture within the multi-layered model. Secondly, the other identifiability issue comes from the factor loadings which can be solved by having the marginalised distribution of the observations or latent factor variables with covariance matrix (composed of the factor loadings and the diagonal error covariance term) to be diagonal with decreasing elements [4].

Another important consideration to make is that with increasing number of layers, the model becomes more complex making it become more difficult to model the underlying true density distribution

**Algorithm 6:** Learning Deep Gaussian Mixture Models

```
function DGMM
Input  : Parameters $\theta : \{\eta_{s_l}, \Lambda_{s_l}, \Psi_{s_l}\}$
Output : Optimised parameters $\theta$
    1. Initialise the parameters via a series of standard Factor Analysers, taking the factor scores to run
       another Factor Analysis - similar to the fitting of DMFA. See Section 3.1.4 for more details.
    2. a) Begin by calculating the expectation of the conditional likelihood of the full marginal of $y$,
          considered to be $z^{(0)}$, given in Section 3.1.2.
       b) Formulate the posterior of the next layer as given in Section 3.1.3, through:
          $p(z^{(i)}|z^{(i-1)}) = p(z^{(i-1)}|z^{(i)})p(z^{(i)})$
       c) Sample from the posterior $p(z^{(i)}|z^{(i-1)}$ in the stochastic S-step
       d) In the E-step, use the samples drawn in the S-step to find $E(z_l^{(l)}|z_i^{(l-1)}, s_l)$ and
          $E(z_i^{(l)} z_i^{(l)}|z_i^{(l-1)}, s_l)$ to carry through to the M-step.
       e) Run this through the M-step to optimise for the following parameters:
          $\Lambda_{s_l}^{(l)} = \frac{\Sigma_{i=1}^n (p(s_l|z_i^{(l-1)})(z_i^{(l-1)} - \eta_{s_l}^{(l)})E(z_l^{(l)}|z_i^{(l-1)}, s_l)E(z_i^{(l)} z_i^{(l)}|z_i^{(l-1)}, s_l)^{-1})}{\Sigma_{i=1}^n p(s_l|z_i^{(l-1)})}$
          $\Psi_{s_l}^{(l)} = \frac{\Sigma_{i=1}^n (p(s_l|z_i^{(l-1)})[(z_i^{(l-1)} - \eta_{s_l}^{(l)})(z_i^{(l-1)} - \eta_{s_l}^{(l)})^T - (z_i^{(l-1)} - \eta_{s_l}^{(l)})E(z_l^{(l)}|z_i^{(l-1)}, s_l)\Lambda_{s_l}^T}{\Sigma_{i=1}^n p(s_l|z_i^{(l-1)})}$
          $\eta_{s_l}^{(l)} = \frac{\Sigma_{i=1}^n (p(s_l|z_i^{(l-1)})[z_i^{(l-1)} - \Lambda_{s_l} E(z_i^{(l)}|z_i^{(l-1)}, s_l)]}{\Sigma_{i=1}^n p(s_l|z_i^{(l-1)})}$
          $\pi_{s_l}^{(l)} = \Sigma_{i=1}^n f(s_l|y_i)$
       f) Repeat this for multiple iterations, forming one full iteration by going through all layers of
          this conditional S-EM algorithm
```

of the data generating process. Through the algorithm of searching the model parameters in a grid, the BIC or AIC, can be compared for each model [6]. This allows a robust and reasonable way to compare models, thereby choosing an appropriate model for the observed data.

It is important for a model to be interpretable and allow more detailed analysis through visualisations of the parameters. However, it is not as simple as it is for DGMM compared to MCFA to visualise the latent variable factor scores onto a similar dimensional plane through common loadings.

It is for this reason that the Thesis will focus on integrating MCFA into DGMM to allow for visualisation of the factors learned in a Deep Mixture Model framework.

## 3.2 Integrating Common Factors into the Deep Gaussian Mixture Model

For MCFA to be implemented under a DGMM framework, the concept and formulae should be similar to that of DMCFA, that is, every layer should be of a MCFA but instead, with jointly learned parameters under the same framework as DGMM. For the model to be projected into the same space, several assumptions must hold. Unlike DMCFA, each of the mixtures are not accompanied by a unique set of projections - thus, there is a common projection for each layer (where a layer is composed of a set of mixtures), where the only parameter conditional on the mixtures is the covariance error term, and the parameters of the latent variable factor scores in the last MCFA layer. Given that the last layer is instrumental in defining the location of the parameter space, it is highly important in being considered the core of the Deep Mixture Model. This can be seen as follows:

### 3.2.1 Model Overview

$$\begin{aligned} y &= A_1 z^{(1)} + e_1 \\ z^{(1)} &= A_2 z^{(2)} + e_2 \\ z^{(2)} &= A_3 z^{(3)} + e_3 \\ z^{(3)} &\sim N(\xi_{s_3}, \Omega_{s_3}) \end{aligned}$$

where each layer incorporates a fixed predefined number of mixture components based on the error term, $e_i \sim N(0, D_{s_k})$

where $s_k = s_1, s_2, s_3$, with k representing the layer index (total number of layers is $h = 3$). Thus, for each $s_i \in s_k$, there is $s_i = g_1, g_2, ..., g_{l-1}, g_l$, where $g_l$ represents the different l mixture components in layer i of $s_i$.

### 3.2.2 Formulating the Marginal

The full form of $p(z^{(i)})$ is given by:

$$p(z^{(i)}) = N(\tilde{\mu}, \tilde{\Sigma})$$

$$\tilde{\mu} = A_1^{(1)}(A_2^{(2)}(...(A_{s_h}^{(h)} \xi_h^{(h)})))$$

$$\tilde{\Sigma} = D_{s_1}^{(1)} + A_1^{(1)}(A_2^{(2)}(...(A_h^{(h)} \Omega_{s_h} A_h^{(h)T} + D_{s_h}^{(h)})...)A_2^{(2)T} + D_{s_2}^{(2)})A_1^{(1)T}$$

where the form of $y$ is given by $z^{(0)}$

It is now evident that the location parameter of this DGMM based MCFA hybrid is a projection based on a series of common factors transforming the mean vector of the latent variable of the last layer. Therefore of great importance the main mixture parameters in the DGMM based MCFA model should have the final layer be taken, unlike the DGMM model where the main mixture parameters are taken from the first layer.

However, for clarity, and completeness it should be stated that the density distributions of $z^{(i)}$ are of the MCFA form,$z^{(i)} \sim N(\xi_{c_i}, \Omega_{c_i})$. It is for this reason that these parameters are also calculated in the BIC, but of great importance, are never used in the marginalised form of $y$, or $z^{(i)}$ for that matter. However, for visualisation purposes, these unused parameters in the model are made use of when plotting the common projected factor scores.

### 3.2.3 Formulating the Posterior

Given the algorithm samples the posterior in the S-step of the S-EM algorithm, the form of posterior is required, and for this DGMM - MCFA hybrid, it can be calculated from the following information:

$$p(z^{(i)}|z^{(i-1)}) \propto p(z^{(i-1)}|z^{(i)}).p(z^{(i)})$$

where the form of the above expressions are given by $p(z^{(i)})$ and $p(z^{(i-1)}|z^{(i)})$:

$$p(z^{(i)}|z^{(i-1)}) = N(\rho_{(s_l)}.(z_{(s_{l-1})}), \xi_{s_l})$$

here,

$$\rho_{(s_l)} = \xi_{s_l}((A^{(l)})^T (D^l_{s_l})^{-1}(z^{l-1}) + \tilde{\Sigma}^{l+1}_{s_l} \tilde{\mu}^{l+1}_{s_l})$$

$$\xi_{s_l} = (\tilde{\Sigma}^{l+1}_{s_l})^{-1} + (A^{((l)})^T (D^l_{s_l})^{-1} A^{(l)}$$

### 3.2.4 Initialising the model

Similarly, the initial observations $y$, are subset into clusters by the k-nearest neighbour algorithm to form $y_{g_k}$, and used to train k separate generalised Matrix Factorisations for each $g_k$ using the function gmf from the package EMMIXmfa. Similar to DGMM, the factor scores of the previous layer are sampled and used as input to the next layer, iterating through this process until all parameters are eventually initialised.

### 3.2.5 Maximum Likelihood Estimation

The algorithm is thus given by sampling from the posterior, using these samples for calculating the conditional expectation in the E-step, and then forming the parameters in the M-step.

To calculate the $n^{th}$ layer common factor scores, the form should be taken, using the parameters from only the $n^{th}$ layer, with respect to the previous factor scores:

$$\hat{z}^{(n+1)} = \xi^{(n)}_{s_n} + \gamma^{(n)}_{s_n}(\hat{z}^{(n)} - A^{(n)}\xi^{(n)}_{s_n})$$

**Algorithm 7:** Learning Deep Gaussian Mixture Models with Common Factor Loadings

```
function DGMM-CFL
Input   : Parameters $\theta : \{A^{(l)}, D^{(l)}_{s_l}, \xi^{(l)}_{s_l}, \Omega^{(l)}_{s_l},\}$
Output : Optimised parameters $\theta$
    1. Initialise the parameters via a series of standard Factor Analysers, taking the factor scores to run
       another Factor Analysis - similar to the fitting of DMFA. For more details, see Section 3.2.4.
    2. a) Calculate the expected conditional likelihood from the parameters by using the full
          marginal of $y$, considered to be $z^{(0)}$, outlined in Section 3.2.2.
       b) Formulate the posterior of the next layer as given in Section 3.2.3, through:
          $p(z^{(i)}|z^{(i-1)}) = p(z^{(i-1)}|z^{(i)})p(z^{(i)})$
       c) Sample from the posterior $p(z^{(i)}|z^{(i-1)}$ in the stochastic S-step
       d) In the E-step, use the samples drawn in the S-step to find $E(z^{(l)}_l|z^{(l-1)}_i, s_l)$ and
          $E(z^{(l)}_i z^{(l)}_i|z^{(l-1)}_i, s_l)$ to carry through to the M-step.
       e) Run this through the M-step to optimise for the parameters
          For each MCFA layer, utilise Algorithm 3 in Section 2.5 to identify the conditional
          probability $p(z^{(i)}|z^{(i+1)})$, which when solved gives the Maximum Likelihood estimates for
          the parameters.
       f) Repeat this for multiple iterations, forming one full iteration by going through all layers in
          this conditional S-EM structured algorithm
```

where,

$$\gamma^{(n)}_{s_n} = (A^{(n)}\Omega^{(n)}_{s_n}A^{(n)T} + D^{(n)}_{s_n})^{-1}A^{(n)}\Omega^{(n)}_{s_n}$$

Notice that this is similar to the factor scores given by MCFA.

Once the parameters are calculated, it is possible to plot the factor scores for each cluster together in the same plane, as the common factor projects the clusters under the same transformation. This gives derivation to a set of explanatory plots in the Cartesian plane with an intuitive meaning - the points are clustered together based on the modelling from the factor scores.

Importantly, the modelling of the location parameter within the marginalised form for the function of $y$, is a set of linear transformations $A_1A_2A_3$ as they are not indexed by the mixture parameters. This indicates that the location parameter is not a non-linear transform within the model, but rather, a set of linear transforms linked together. This makes the model less flexible and less powerful to capture non-linear signals in the data generating process.

To improve upon this, new models were derived to solve this issue, while maintaining the interpretability and model stability (through reduced number of parameters) with common factor loadings. This developed into two new solutions:

1. Deep Gaussian Mixture Models with an Initial Common Factor Loading (DGMM-ICFL): A first MCFA layer in a deep hierarchical mixture, followed by a series of MFA, that is, similar to that of a DGMM model with a first MCFA layer at the top to transform the subsequent deep MFA layers below it via a common projection through common factor loadings.

2. Deep Gaussian Mixture Models with Hierarchical Mixtures of Common Factor Loadings (DGMM-HMCFL): A series of MCFA layers, with the first being a standard MCFA layer, followed by a series of hierarchical MCFA layers with the common factor loadings based on the mixtures of the current MCFA layer.

## 3.3 Deep Gaussian Mixture Models with an Initial Common Factor Loading

An initial layer of MCFA ontop of a DGMM allows the full flexibility of the DGMM model, with an added layer to provide a common projection via the common factor loading for plotting purposes.

Thus, compared to the original MCFA model as outlined in Section 2.1, the following expressions outline the DGMM-ICFL model:

### 3.3.1 Model Overview

$$
\begin{aligned}
y &= A_1 z^{(1)} + e_1 \\
z^{(1)} &= \eta_{s_2} + \Lambda_{s_2} z^{(2)} + e_2 \\
z^{(2)} &= \eta_{s_3} + \Lambda_{s_3} z^{(3)} + e_3 \\
z^{(3)} &\sim N(0, I)
\end{aligned}
$$

where, $e_i \sim N(0, D_{s_i})$

where $s_k = s_1, s_2, s_3$, with k representing the layer index (total number of layers is $h = 3$). Thus, for each $s_i \in s_k$, there is $s_i = g_1, g_2, ..., g_{l-1}, g_l$, where $g_l$ represents the different l mixture components in layer i of $s_i$.

Thus it is evident that for all $z^{(i)}$, there are factor loadings with components based on the mixtures at each layer: $\Lambda_{s_i}$.

The idea is thus to learn an integrated model called the Deep Gaussian Mixture Model with an Initial Common Factor Loading (DGMM-ICFL) that learns all the parameters, including the first MCFA layer parameters jointly

### 3.3.2 Formulating the Marginal

The full form of $p(z^{(i)})$ is given by:

$$p(z^{(i)}) = N(\tilde{\mu}, \tilde{\Sigma})$$

$$\tilde{\mu} = A_1^{(1)}(\eta_{s_2}^{(2)} + \Lambda_{s_2}^{(2)}(...(\eta_{s_1}^{(1)} + \Lambda_{s_{h-1}}^{(h-1)}\eta_h^{(h)})))$$

$$\tilde{\Sigma} = D_{s_1}^{(1)} + A_1^{(1)}(\Lambda_{s_2}^{(2)}(...(\Lambda_{s_h}^{(h)}\Lambda_{s_h}^{(h)T} + D_{s_h}^{(h)})...)\Lambda_{s_2}^{(2)T})A_1^{(1)T}$$

where the form of $y$ is given by $z^{(0)}$

Evidently, the first layer is a common projection that enables the factor scores to be plotted onto a common plane. The subsequent layers are based off the DGMM, and thus ideally, enables the flexibility and power of the original model, while also adding a sense of interpretability from the common projection given by the first layer.

### 3.3.3 Formulating the Posterior

The posterior can be calculated from the following information:

$$p(z^{(i)}|z^{(i-1)}) \propto p(z^{(i-1)}|z^{(i)}).p(z^{(i)})$$

where the form of the above expressions are given by $p(z^{(i)})$ and $p(z^{(i-1)}|z^{(i)})$:

$$p(z^{(i)}|z^{(i-1)}) = N(\rho_{(s_l)}.(z_{(s_{l-1})}), \xi_{s_l})$$

Thus, the form of the posterior in the first modified MCFA layer is given by:

$$\rho_{(s_l)} = \xi_{s_l}((A^l)^T(D^l_{s_l})^{-1}(z^{l-1}) + \tilde{\Sigma}^{l+1}_{s_l}\tilde{\mu}^{l+1}_{s_l})$$

$$\xi_{s_l} = (\tilde{\Sigma}^{l+1}_{s_l})^{-1} + (A^l)^T(D^l_{s_l})^{-1}A^l$$

### 3.3.4 Initialising the model

Again, initial observations $y$, are subset into clusters by the k-nearest neighbour algorithm to form $y_{g_k}$, and used to train two separate algorithms for initialisation of both the first Common Factor analytic layer by generalised Matrix Factorisation, and the subsequent MFA layers by standard Factor Analysis - all on the k mixture components $g_k$ that subset the observations $y_{g_k}$.

### 3.3.5 Maximum Likelihood Estimation

The following depicts the S-EM algorithm for the DGMM model with an initial CFL.

One disadvantage of this technique is that the MCFA layer to be plotted requires a dimension of either 2 or 3, to be able to fit into a plot of the equivalent dimension. Thus, the first MCFA layer in this method requires to be fixed at either 2 or 3, ignoring the model identifiability constraint requiring the factors be of decreasing dimension (although, this still holds for the subsequent MFA layers).

However, note that this can be avoided with dimensionality reduction techniques, by extracting the two most important components from the corresponding factor scores of the initial MCFA layer (which can now be of dimension greater than 3). With the two components, plotting the reduced dimensionality of the factor scores from the initial MCFA layer is possible.

**Algorithm 8:** Learning Deep Gaussian Mixture Models with an Initial Common Factor Loadings

```
function DGMM-ICFL
Input  : Parameters $\theta : \{A_1^{(1)}, D_{s_l}, \eta_{s_l}, \Lambda_{s_l}, \eta_{s_l}, \Omega_{s_l}\}$
Output : Optimised parameters $\theta$
    1. Initialise the parameters via a series of standard Factor Analysers, taking the previous layers of
       factor scores to run the next layer of factor analysers - similar to the fitting of DMFA. For more
       details, see Section 3.3.4 .
    2.  a) Using the full marginal of $y$, considered to be $z^{(0)}$, calculate the expected conditional
           likelihood from the parameters outlined in Section 3.3.2.
        b) Formulate the posterior of the next layer as given in Section 3.3.3, through:
           $p(z^{(i)}|z^{(i-1)}) = p(z^{(i-1)}|z^{(i)})p(z^{(i)})$
        c) Sample from the posterior $p(z^{(i)}|z^{(i-1)})$ in the stochastic S-step
        d) In the E-step, use the samples drawn in the S-step to find $E(z_l^{(l)}|z_i^{(l-1)}, s_l)$ and
           $E(z_i^{(l)} z_i^{(l)}|z_i^{(l-1)}, s_l)$ to carry through to the M-step.
        e) Run this through the M-step to optimise for the parameters.
           In the first layer, the parameters for the MCFA are learned, whereas in subsequent layers,
           parameters for the MFA are learned.
           Thus for the first MCFA layer, utilise Algorithm 3 in Section 2.5 to identify the conditional
           probability $p(z^{(i+1)}|z^{(i)})$, whereas the subsequent layers will utilise Algorithm 6 in Section
           3.1.4 based on the DGMM to identify the parameters of the MFA layers.
        f) Repeat this for multiple iterations, forming one full iteration by going through all the
           conditional layers of this S-EM algorithm
```

## 3.4 Deep Gaussian Mixture Models with Hierarchical Mixtures of Common Factor Loadings

A similar proposal can be made to the DGMM-ICFL, where subsequent layers after the initial MCFA layer, are also MCFA (as compared to MFA in DGMM-ICFL). However, the common factor loadings must now be based on the mixtures to correct for the linearity in the location parameter as noted in Section 2.1.

### 3.4.1 Model Overview

$$y = A^{(1)}z^{(1)} + e_1$$

$$z^{(1)} = A_{s_2}^{(2)} z^{(2)} + e_2$$

$$z^{(2)} = A_{s_3}^{(3)} z^{(3)} + e_3$$

$$z^{(3)} \sim N(\xi_{s_3}, \Omega_{s_3})$$

where, $e_i \sim N(0, D_{s_i})$

where $s_k = s_1, s_2, s_3$, with k representing the layer index (total number of layers is $h = 3$). Thus, for each $s_i \in s_k$, there is $s_i = g_1, g_2, ..., g_{l-1}, g_l$, where $g_l$ represents the different l mixture components in layer i of $s_i$.

Thus it is evident that for all $z^{(k)}, k \geq 2$, there are common factor loadings based on the mixtures at each layer: $A^{(k)}_{s_k}, k \geq 2$.

By having hierarchical layers of MCFA with common factor loadings based on the mixtures of each layer, the method can be called the Deep Gaussian Mixture Model with Hierarchical Mixtures of Common Factor Loadings (DGMM-HMCFL).

### 3.4.2 Formulating the Marginal

The full form of $p(z^{(i)})$ is given by:

$$p(z^{(i)}) = N(\tilde{\mu}, \tilde{\Sigma})$$

$$\tilde{\mu} = A^{(1)}(A^{(2)}_{s_2}(...(A^{(h)}_{s_h}\xi^{(h)}_h)))$$

$$\tilde{\Sigma} = D^{(1)}_{s_1} + A^{(1)}(A^{(2)}_{s_2}(...(A^{(h)}_{s_h}\Omega_{s_h}A^{(h)T}_{s_h} + D^{(h)}_{s_h})...)A^{(2)T}_{s_2} + D^{(2)}_{s_2})A^{(1)T}$$

where the form of $y$ is given by $z^{(0)}$

It is now evident that the first layer is the common projection that enables the factor scores to be plotted onto a common plane. The subsequent layers are a series of common factors with components based on mixtures that enables non-linear modelling of the data generating process compared to the naive DGMM based MCFA hybrid that was formulated in the Section 3.1 of this chapter.

### 3.4.3 Formulating the Posterior

The form of the posterior is given by the Bayes formulation based on the prior and the likelihood:

$$p(z^{(i)}|z^{(i-1)}) = N(\rho_{(s_l)} \cdot (z_{(s_{l-1})}), \xi_{s_l})$$

Thus, the form of the posterior in the first modified MCFA layer is given by:

$$\rho_{(s_l)} = \xi_{s_l}((A^{(l)})^T(D^l_{s_l})^{-1}(z^{l-1}) + \tilde{\Sigma}^{l+1}_{s_l}\tilde{\mu}^{l+1}_{s_l})$$

$$\xi_{s_l} = (\tilde{\Sigma}^{l+1}_{s_l})^{-1} + (A^{(l)})^T(D^l_{s_l})^{-1}A^{(l)}$$

### 3.4.4 Initialising the model

Here, the initial training is similar to that of DGMM-CFL, where the initial observations $y$, are subset into clusters by the k-nearest neighbour algorithm to form $y_{g_k}$, and used to train k separate generalised Matrix Factorisations for each $g_k$ using the function gmf from the package EMMIXmfa. Importantly, the two models are distinguished by their structures from the second layer onwards, where the common projections $A_{g_k} fork \geq 2$ are now indexed by the model in the DGMM hierarchical mixtures of CFL. Thus, an important distinction is to learn the projections for each subset using the latent factor scores from the previous layer. This is iterated over each layer until all parameters are eventually learned.

### 3.4.5 Maximum Likelihood Estimation

With the S-EM algorithm, the DGMM with hierarchical mixtures of CFL can be trained to extract the parameters.

**Algorithm 9:** Learning Deep Gaussian Mixture Models with Hierarchical Mixtures of Common Factor Loadings

```
function DGMM-HMCFL
Input  : Parameters $\theta : \{A^{(1)}, A^{(l)}_{s_l}, D_{s_l}, \xi_{s_l}, \Omega_{s_l}\}$
Output : Optimised parameters $\theta$
    1. Initialise the parameters via a series of standard Factor Analysers, taking the factor scores to run
       another Factor Analysis - similar to the fitting of DMFA. For more details, see Section 3.4.4.
    2.  a) Using the full marginal of $y$, considered to be $z^{(0)}$, calculate the expected conditional
           likelihood from the parameters outlined in Section 3.3.2.
        b) Formulate the posterior of the next layer as given in Section 3.3.3, through:
           $p(z^{(i)}|z^{(i-1)}) = p(z^{(i-1)}|z^{(i)})p(z^{(i)})$
        c) Sample from the posterior $p(z^{(i)}|z^{(i-1)}$ in the stochastic S-step
        d) In the E-step, use the samples drawn in the S-step to find $E(z^{(l)}_l|z^{(l-1)}_i, s_l)$ and
           $E(z^{(l)}_i z^{(l)}_i|z^{(l-1)}_i, s_l)$ to carry through to the M-step.
        e) Run this through the M-step to optimise for the parameters.
           In every layer, the parameters for the MCFA are learned via Algorithm 3 (although, when
           the factor loadings are components that are based on the mixtures, the algorithm must be
           modified). It is only the first layer where the common factor loadings are not composed by
           the mixtures. This is to provide a single common projection for visualisation.
        f) Repeat this for multiple iterations, forming one full iteration by going through all
           conditional layers of the S-EM algorithm
```

Chapter 4

# Methodology for implementation of the Deep Gaussian Mixture Model with Common Factor Loadings

## 4.1 Introduction

An evaluation of the implemented models requires fair testing of the model parameters to find the most suitable parameters that give a high likelihood and information criterion measure which provide the basis for selection of a particular model. As given in Section 2, the following models will be evaluated using the Deep Gaussian Mixture Models framework:

1. Common Factor Loadings (DGMM-CFL) [Section 2.1]
2. Initial Common Factor Loadings (DGMM-ICFL) [Section 2.2]
3. Hierarchical Mixtures of Common Factor Loadings (DGMM-HMCFL) [Section 2.3]

These will also be evaluated in conjunction with the methods tested in Viroli and McLachlan [4], with the DGMM being evaluated under the same conditions as what was given in the paper.

### 4.1.1 Comparison with Deep Gaussian Mixture Models

Using R v3.5.1, three of the same five datasets as tested by Viroli and McLachlan [4] were used to evaluate the implementations of the novel use of common factor loadings. The larger datasets including both the Vehicle dataset and the Satellite dataset were not implemented, as these datasets were prohibitively large. The source for the baseline datasets can be found at the UCI repository [15], or accessed via the Github page:

The datasets consist of the following:

1. Wine Data: Coming from a study, the dataset consists of 27 chemical and physical properties of three types of wine: Barolo ($n_1$ = 59), Grignolino ($n_2$ = 71), Barbera ($n_3$ = 48).

2. Olive Data: Coming from a study, the dataset consists of fatty acids found by lipid fraction of olive oils. The oils come from classes consisting of three different regions from where the olives are sourced: Southern Italy ($n_1$ = 323), Sardinia ($n_2$ = 98), Northern Italy ($n_3$ = 151).

3. Ecoli Data: Coming from a study, the dataset consists of 7 variables based on biological information pertaining to amino acid sequences. Classes consist of 8 unbalanced groups: cytoplasm ($n_1$ = 143), inner membrane without signal sequence ($n_2$ = 77), perisplasm ($n_3$ = 52), inner membrane uncleavable signal sequence ($n_4$ = 35), outer membrane ($n_5$ = 20), outer membrane lipoprotein ($n_6$ = 5), inner membrane lipoprotein ($n_7$ = 2), inner membrane cleavable signal sequence ($n_8$ = 2)

4. Vehicle Data: Coming from a study, the dataset has 18 features representing the angles made by the silhoutte of 4 different vehicle types, which represent the 4 clusters: bus ($n_1$ = 118 ), saab ($n_2$ = 217 ), van ($n_3$ = 199 )and opel ($n_4$ = 212).

5. Satellite Data: Coming from a study, the dataset has 36 features with 6 mixture components of spectral-based images: red soil ($n_1$ = 1533), cotton crop ($n_2$ = 703), grey soil ($n_3$ = 1358), damp grey soil ($n_4$ = 626), soil with vegetation stubble ($n_5$ = 707) and very damp grey soil ($n_6$ = 1508).

Prior to input into the model, the datasets were scaled to have a standard normal distribution, that is, with zero vector mean and an identity variance-covariance matrix. This is in order to meet the assumptions of the model, requiring standard normal scaled observations.

Furthermore, BIC was used as the main choice in deciding between models. This is a result of the information criterion incorporating both the likelihood and the number of parameters under the same formula.

The results given in the paper on the standard Deep Gaussian Mixture Model was measured by the Adjusted Rand Index (ARI). The ARI is a good metric for cluster-based data, especially in the unsupervised domain (as with the DGMM and variants). This premise is based on the intended purpose, that is in its' design to measure the similarity between two clusters. A strong result will be given by a high ARI, and vice versa, where the highest value is 1, and the lowest value is 0 (results can be negative, but this is an artefact of the metric).

Given the stochastic nature of Monte Carlo Expectation Maximisation, and the initial seeding to find suitable starting parameters, ten iterations were made for each run of a full model [4]. This is to ensure that both the BIC and ARI metrics did not occur by chance, and that the final summary statistics are robust to initial seeds.

### 4.1.2 Comparison with Gene Expression datasets

Additionally, five experimental datasets sourced from single-cell RNA-sequencing (scRNA-seq) will be used. These datasets involve gene expression at the single cell level, compared to tissue level as with the standard RNA-seq protocols [16] [17] [18] [19] [20]. The source for the single cell gene expression datasets can be accessed via the Github page.

Gene expression is the abundance level at which genes are expressed after being copied: transcripted, and translated from DNA into RNA. They come in the form of transcripts, unique count-based measures, and thus, most protocols involve integer-based data. There are some protocols where the transcripts are measured as continuous, due to the nature of the processing protocol [21].

Given the prohibitively large sizes of these datasets, the data will be reduced via dimensionality reduction to extract the relevant signals in a lower dimension, allowing for DGMM and variants to run properly.

Similar to a state-of-the-art unsupervised clustering technique, SC3 [22], the original raw untransformed data will have cells measured pair-wise for the Pearson correlation, Spearman Correlation, and a proportionality based measure, 'Phi-S', published by Quinn et al [23], which is based on the variance between two cells across the level of expression for all selected genes. Once these pair-wise measures are taken, their positive definite symmetric matrices are taken and run through Principal Component Analysis (PCA), where the first 3 Principal Components are taken, and then put through the DGMM variants.

It is important to note that the genes are selected based on frequency of expression (greater than zero), across more than ten percent of cells, and less than ninety percent of cells, following the protocol by [22].

| Dataset | Organism | Biological Source | Sample Size | Cluster Size |
|---|---|---|---|---|
| Goolam [16] | Mouse | Embryo Development | 124 | 5 |
| Darmanis [17] | Human | Brain tissue | 420 | 9 |
| Li [18] | Human | Cell Line | 460 | 10 |
| Deng [20] | Mouse | Embryo Development | 286 | 10 |
| Yan [19] | Human | Embryo Development | 90 | 6 |

Figure 4.1: Extra details on the Single Cell Gene Expression datasets

## 4.2 Implementing the Deep Gaussian Mixture Model and Common Factor Loading designs

Here, details are given on the preprocessing, prior to the training of the model when the parameters are learned.

Following, several examples are given with specifications on the pre-defined model parameters that are tunable, as evident in the model_selection function of the deepgmm package and the dgmmcfl github page.

### 4.2.1 Preprocessing

Across the baseline datasets, it is evident from the models that each DGMM and DGMM-CFL design expects pre-processed sampled observations scaled to the standard normal distribution, that is $Z \sim N(0, I)$.

For the single-cell gene expression datasets, a similar protocol to SC3 is followed [22].

1. Filter the raw, unscaled, yet sparse dataset such that: if $CC$ were the total number of cells, than only keep x, in the following: $0.1 * CC < x < 0.9 * CC$

2. The filtered, original dataset is taken and all genes are scaled to a variance of 1, without centering.

3. The cells (where subsets contain the clusters of interest) from the now scaled dataset are taken and calculated using the Phi-S measure, a pair-wise measure based on comparing the separate and pooled variances, termed 'proportionality' (a ratio between the sum of the variances of each cell in the pair, against the variance of the paired pooled cells). This measure is positive definite symmetric.

4. These are then run through Principal Component Analysis to extract the top components; usually the eigenvalues in descending order that contain 70% of the variance have their corresponding eigenvectors taken. However, for computational purposes, the top 5 components are taken.

5. Using these principal components from Phi-S, they are then scaled to a standard normal and put through the DGMM and CFL designs.

   Note, the reason why the eigenvectors of the pair-wise matrix Phi-S is used as the set of features is explained by Skinnider, Squair and Foster [24]. Given the success of the method as a highly ranked pair-wise measure (showing improvement above standard correlation measures such as Pearson, Spearman and Kendall) for hierarchical clustering, it was the preferred choice to use in the unsupervised clustering of the DGMM and CFL designs. The added process in the workflow of scaling but not centering the data is in order to enable a normalised comparison in the variance ratio measure between cells.

### 4.2.2 Examples on selecting predefined model parameters

Note, in the models described in Section 3, all the models were of three layers, including the first layer with $\hat{y}$, the function estimating the observations from the data generating process $y$. Given a predefined number of layers, anywhere in the space of all positive integers (Note: Tang et al [9] advises to stay within 1, 2, or 3 layers deep as any more layers becomes overly complex), the choice is then made at each layer on the number of mixtures (represented by $k$ ) and dimensionality of the latent factors (represented by $r$ ).

### Example of a 2-layer DGMM implementation

For example, a 2 layer DGMM modelled on the 'Wine' dataset with 3 clusters across the defined labels should have:

1. 1st layer:
$$k_1 = 3$$
$$r_1 = R^1$$

2. 2nd layer:
$$k_2 = K^2$$
$$r_2 = R^2$$

where $R^1, R^2, K^2$ are positive integers, and the constraint $R^1 > R^2$ must hold for identifiability purposes, including for calculating the BIC. Here it is important to note that the 1st layer is always denoted the 'main' layer of the model.

### Example of a 3-layer DGMM-CFL and DGMM-ICFL implementation

For example, a 3 layer DGMM-CFL and a 3 layer DGMM-ICFL modelled on the 'Ecoli' dataset with 6 clusters across the defined labels should have:

1. 1st layer:
$$k_1 = K^1$$
$$r_1 = R^1$$

2. 2nd layer:
$$k_2 = K^2$$
$$r_2 = R^2$$

3. 3rd layer:
$$k_3 = 6$$
$$r_3 = R^3$$

where $R^1, R^2, R^3, K^1, K^2$ are positive integers, and the constraint $R^1 > R^2 > R^3$ must hold for identifiability purposes, including for calculating the BIC. Here it is important to note that the final layer is always denoted the 'main' layer of the model.

**Example of a 3-layer DGMM-HMCFL implementation**

For example, a 3 layer DGMM-HMCFL modelled on the 'Vehicle' dataset with 4 clusters across the defined labels should have:

1. 1st layer:
$$k_1 = K^1$$
$$r_1 = R^1$$
2. 2nd layer:
$$k_2 = 4$$
$$r_2 = R^2$$
3. 3rd layer:
$$k_3 = K^3$$
$$r_2 = R^3$$

where $R^1, R^2, K^1, K^3$ are positive integers, and the constraint $R^1 > R^2 > R^3$ must hold for identifiability purposes, including for calculating the BIC. Here it is important to note that the 2nd layer is always denoted the 'main' layer of the model.

**Model Parameters for the standard DGMM and across all variants**

Thus, the model parameters exhaustively include:

1. layers: the number of layers the Deep Gaussian Mixture Model contains
2. g: the number of mixtures the main layer has to cluster the objective labels in an unsupervised manner
3. k: the number of mixtures for each layer
4. r: the dimensions of the factors from each layer
5. seed: the seed for initialisation of parameters, and sampling of the posterior in the S-EM algorithm
6. iter: the number of iterations for the S-EM algorithm
7. eps: the threshold at which the likelihood converges to

To find the appropriate model, all model parameters for *k* and *r* are iterated over, with BIC providing the definitive factor between the top models. Thus, the performance measure for unsupervised clustering, the adjusted Rand Index, is supported and guided by statistical theory on BIC.

# Chapter 5

# Results: Evaluating the Deep Gaussian Mixture Model and Common Factor methods

This Chapter will provide three main pieces of results:

1. Quality Control: to check that the algorithms are converging and that the visualisation of the DGMM-CFL common factor scores are detecting true signals from the observed process.

2. Main Results: to show the adjusted Rand Index (a measure of unsupervised clustering accuracy) against each dataset (Viroli and McLachlan datasets, and Gene Expression datasets), for each method.

3. Method Comparison: to compare the visualisation of the methods in comparison to state of the art methods: t-distributed Stochastic Neighbourhood Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP).

Unless otherwise stated all plots involve true labels. This is to show the ability to visualise the dimensionally reduced datasets across the methods.

## 5.1 Preliminary Results: Quality Control

The purpose of quality control is to ensure initial results are valid, and to support downstream results and conclusions made. Here, we will look at the convergence of the maximum likelihood across all methods. Then, we will assess the visualisation ability against MCFA, and whether it is able to detect true signals.

### 5.1.1 Convergence of Maximum Likelihood via Stochastic Expectation Maximisation (S-EM) algorithm

For correct results, algorithms must converge monotonically after several rounds from the initial seed. This is seen in the next figure, showing a series of plots from the S-EM based algorithm, highlighting the monotonic increase of the likelihood, quickly converging as a result of a good initialisation strategy.

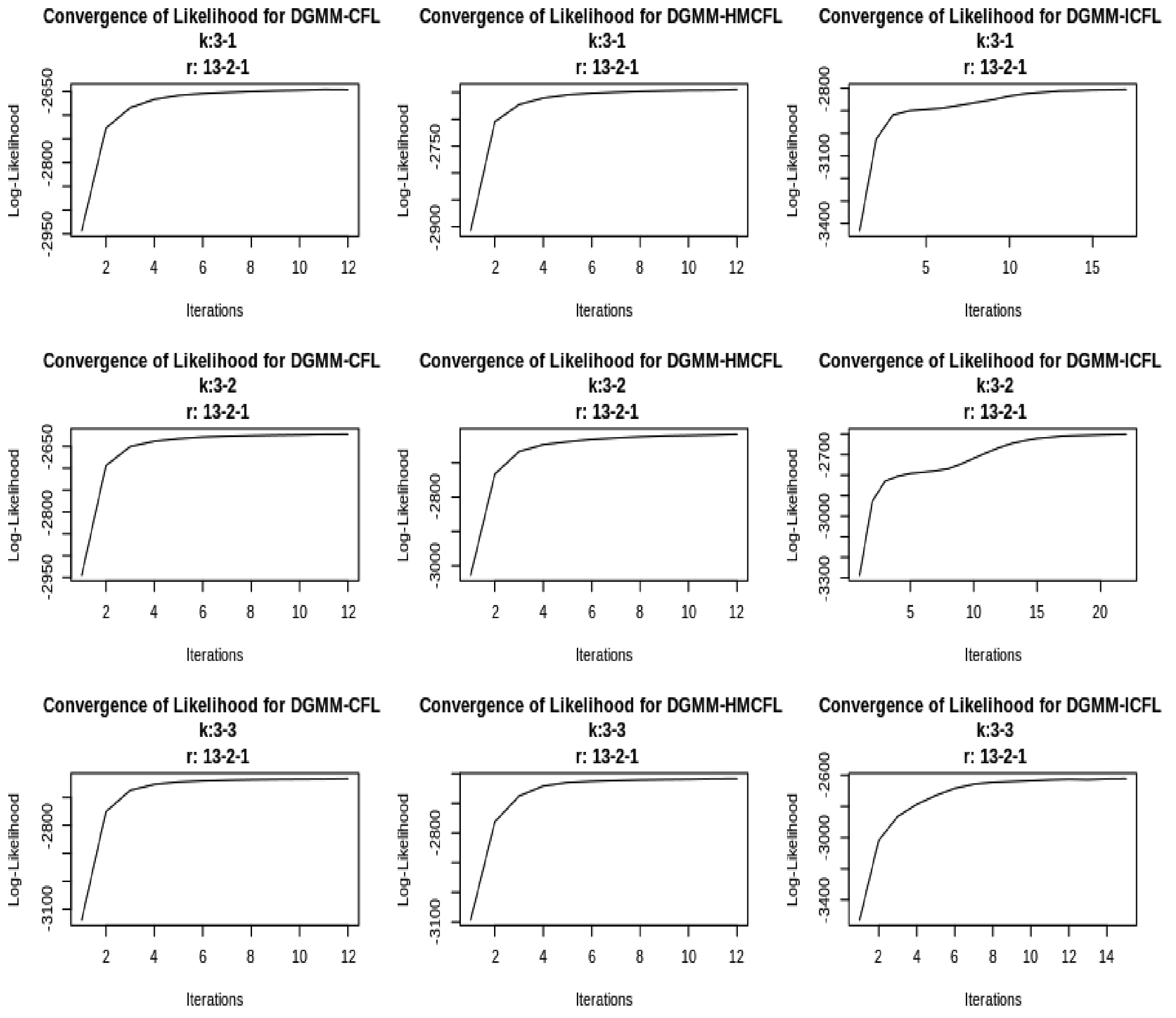


Figure 5.1: For each novel method: DGMM-CFL, DGMM-HMCFL, DGMM-ICFL, the likelihood is shown after each iteration of the S-EM algorithm. Notice all plots show a monotonic increase in the likelihood, converging fairly quickly after a relatively short number of iterations.

### 5.1.2 Visualisation check with Mixtures of Common Factor Analysers (MCFA)

The aim of this section is to provide evidence for the new designs in being able to detect true signals. Here, each of the factor models provides a dimensionally reduced set of factor scores, allowing a visualisation technique, here t-SNE is used on top of each factor analytic method by extracting the mean factor scores in order to provide a better depiction of the data.

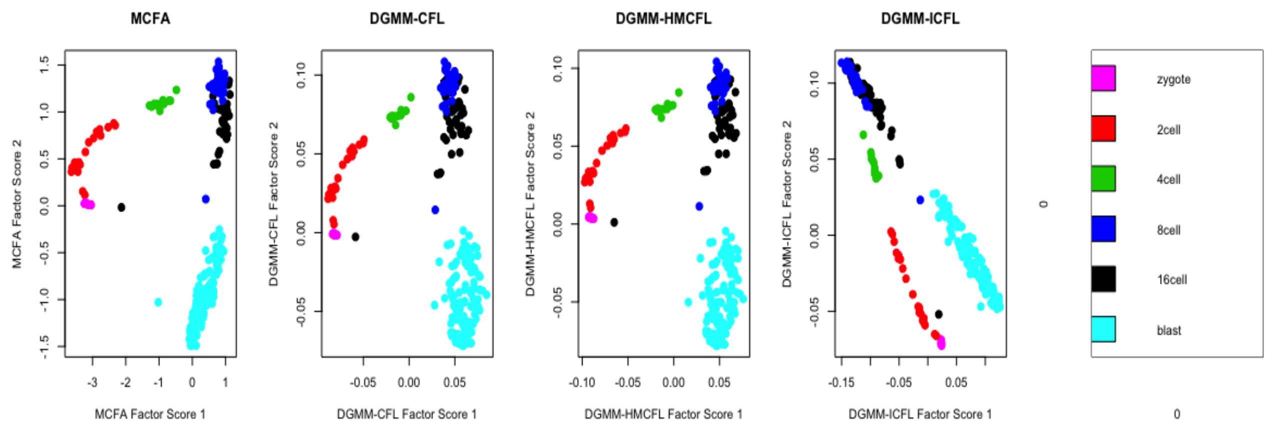


Figure 5.2: From the zygote cluster to the blastocyte, tracing the points reveals a trajectory that reflects the lineage of embryonic development (from zygote, 2 cell, 4 cell, 8 cell, 16 cell, blastocyte).

## 5.2 Main Results: Performance of Deep Gaussian Mixture Models with Common Factors

We here show that the DGMM performs well, attributed to its' flexibility and purpose for unsupervised clustering. This method is followed by the DGMM with hierarchical mixtures of CFL variant, combining both MCFA and DGMM and we show that it is competitive with the original MCFA method.

### 5.2.1 Viroli and McLachlan Datasets

The purpose of comparing the methods with the Viroli and McLachlan datasets is to compare the methods with DGMM as a baseline. By acting as a baseline, and through our knowledge that the methods converge and reveal relevant signals, we can tell whether the methods are now accurate and comparable with the original DGMM.

| Dataset | DGMM | MCFA | DGMM-CFL | DGMM-HMCFL | DGMM-ICFL |
|---|---|---|---|---|---|
| Wine | 0.981 | 0.527 | 0.898 | 0.896 | 0.912 |
| Olive Oil | 0.520 | 0.812 | 1.00 | 1.00 | 0.870 |
| Ecoli | 0.770 | 0.01 | 0.415 | 0.643 | 0.641 |
| Vehicle | 0.203 | 0.164 | 0.140 | 0.135 | 0.07 |
| Satellite | 0.601 | 0.442 | 0.425 | 0.451 | 0.533 |

Figure 5.3: Adjusted Rand Index from the datasets of Viroli and McLachlan [4] for baseline comparison

### 5.2.2 Single-cell Gene Expression Datasets

It is ideal to now compare methods with more modern complex biological systems from the array of Single-cell Gene Expression datasets. There is a very strong push in the field of single-cell research, for more advanced techniques that allow clustering of cell-based data to discover and annotate newly discovered cells types, groups and populations, aiding in the development of new therapeutic targets and further research. Here, we also take the results from SC3, a state of the art single-cell clustering method and compare them with the suite of DGMM methods. The best BIC is taken for each DGMM method, out of 10 runs of the EM algorithm.

| Dataset | SC3 | DGMM | MCFA | DGMM-CFL | DGMM-HMCFL | DGMM-ICFL |
|---|---|---|---|---|---|---|
| Goolam | 0.630 | 0.678 | 0.544 | 0.665 | 0.678 | 0.537 |
| Darmanis | 0.728 | 0.586 | 0.521 | 0.654 | 0.514 | 0.501 |
| Li | 0.963 | 0.939 | 0.942 | 0.824 | 0.930 | 0.865 |
| Deng | 0.555 | 0.581 | 0.466 | 0.503 | 0.510 | 0.354 |
| Yan | 0.621 | 0.532 | 0.815 | 0.260 | 0.501 | 0.505 |

Figure 5.4: Adjusted Rand Index from a variety of complex single cell datasets

## 5.3 Method Comparison: Visualisation with state-of-the-art dimensionality reduction techniques

This chapter will aim to reveal the strength of the novel DGMM-CFL methods in terms of visualisation, in addition to the added benefit of dimensionality reduction and unsupervised clustering.

The first of three figures will outline the benefit of dimensionality reduction by the new Common Factor based DGMM designs through comparison with t-SNE [13] and UMAP [14]. Following the three new designs, DGMM-CFL, DGMM with hierarchical mixtures of CFL (DGMM-HMCFL), DGMM with initial CFL (DGMM-ICFL) will be compared in conjunction with PCA. Lastly and most importantly, it will be shown that a model-based visualisation (here, with MCFA, or DGMM-CFL) technique based on a series of hierarchical Gaussian latent variables provides good visualisations on small datasets compared to t-SNE and UMAP.

The following will now describe more of the actual plots and contents.

The first of three figures includes the following:

1. DGMM-HMCFL + t-SNE - the dimensionality of the dataset is initially reduced via DGMM with hierarchical mixtures of CFL, where the mean factor scores are taken and put through t-SNE

2. MCFA + t-SNE - similarly, Mixtures of Common Factor Analysers are used to reduced the dimension of the original dataset, where the mean factor scores are then put through t-SNE

3. raw t-SNE - t-SNE is used on the original dataset

4. raw UMAP - UMAP is used on the original dataset

Where the second of three figures contains the DGMM variants with PCA: the dimensionality of the dataset is initially reduced via the DGMM variants, where the mean factor scores are taken and put through PCA and the top 5 principal components are taken.

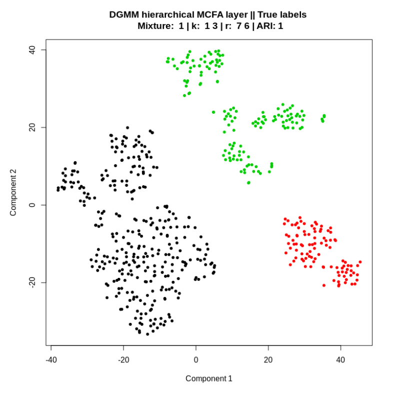
The final figure of three shows the raw factor scores from the naive DGMM-CFL. Unlike the shallow MCFA, this is a deep mixture of factor analysers and thus gives a better representation through a deep and hierarchical method.

In the figure below, Figure 5.5 reveals the benefit of dimensionally reducing the data via the model-based designs. Both MCFA and DGMM with hierarchical mixtures of CFL provide good visualisations of the true labelled clusters across small (Yan et al [19]), medium (Olive Oil [15]) and large datasets (Li et al [18]) when combined with t-SNE. When comparing to t-SNE and UMAP on the raw original untransformed data, we do notice that the model-based methods perform consistently well across sample sizes, compared to poor performance on Yan et al for t-SNE and the mixed cluster labels on the Olive Oil dataset for UMAP. Overall, model-based methods involving both MCFA and DGMM with hierarchical mixtures of CFL shows an improvement in terms of visualising true clusters.

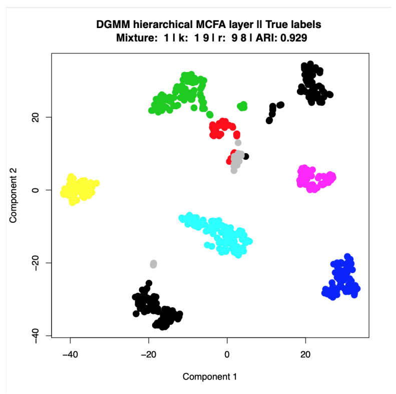
Figure 5.6 below, shows the mean factor scores from each method - DGMM-CFL, DGMM with hierarchical mixtures of CFL, and DGMM with initial CFL, dimensionally reduced by Principal Component Analysis (PCA), and reveals the top 5 principal components for the true labels on the Olive Oil dataset. Careful inspection shows that the clusters overlap the most in DGMM with hierarchical mixtures of CFL, followed by DGMM with initial CFL, with the naive implementation of deep MCFA: DGMM-CFL, showing the most promising visualisation once dimensionally reduced by PCA.

Furthermore, the appeal for DGMM-CFL as a tool for visualisation is depicted below in Figure 5.7. The model-based DGMM-CFL is compared with the factor scores from MCFA for the purpose of visualisation. Note that the two are similar in terms of cluster location (relative to one another), where both are model derived. These two visualisations are then compared with state-of-the-art methods t-SNE (perplexity is set at 15 for this smaller dataset), and UMAP, with both methods use the raw original untransformed data.

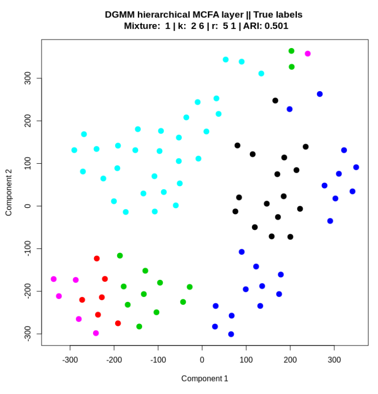
Of great importance, note that in Figure 5.7, no extra transformations are made at any point across all the methods, except for a Z-transform scale for the model-based methods (required based upon Gaussian assumptions of DGMM-CFL and MCFA).



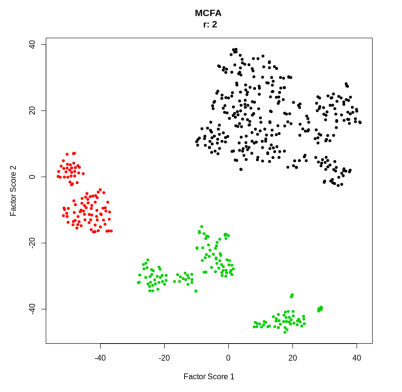
(a) Olive Oil with DGMM-HMCFL + t-SNE



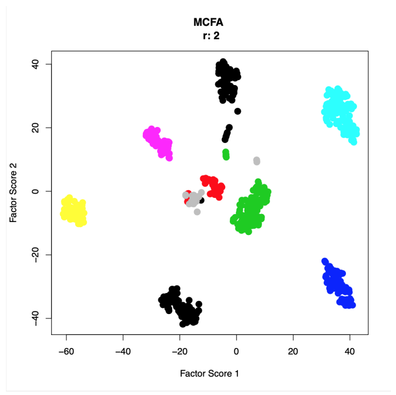
(b) Li et al with DGMM-HMCFL + t-SNE



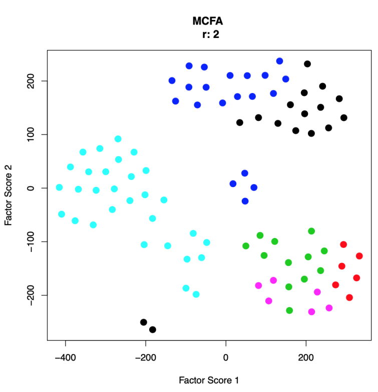
(c) Yan et al with DGMM-HMCFL + t-SNE



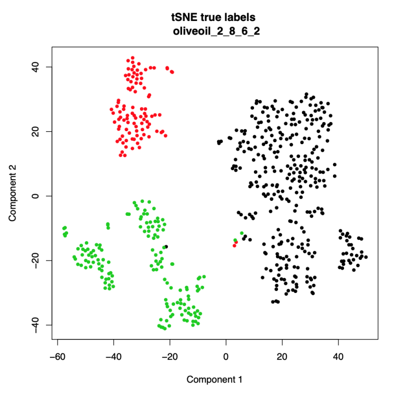
(d) Olive Oil with MCFA + t-SNE



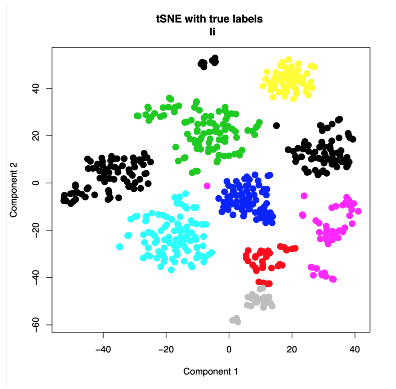
(e) Li et al with MCFA + t-SNE



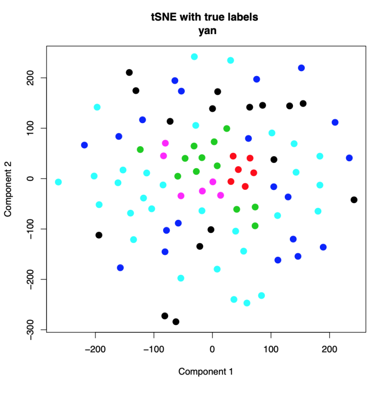
(f) Yan et al with MCFA + t-SNE



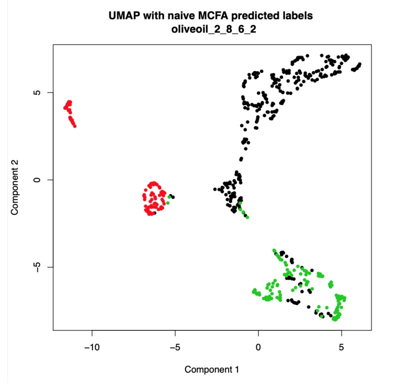
(g) Olive Oil with t-SNE



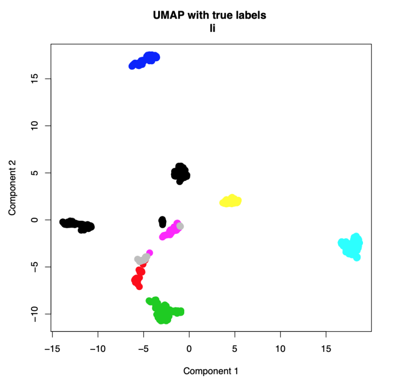
(h) Li et al with t-SNE



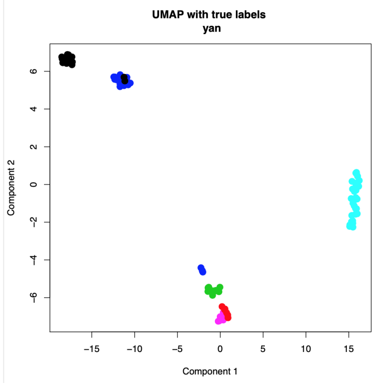
(i) Yan et al with t-SNE



(j) Olive Oil with UMAP

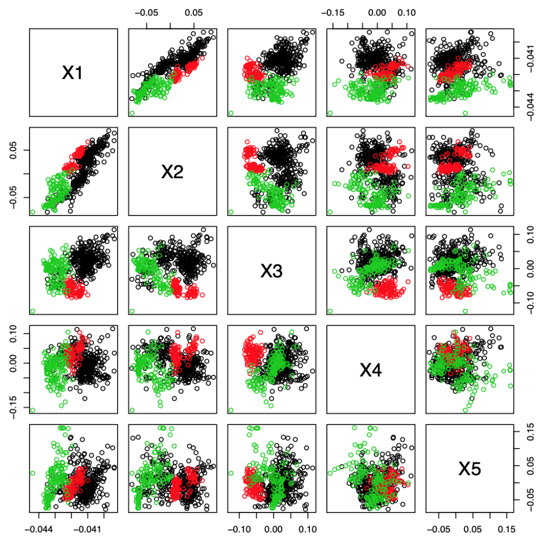


(k) Li et al with UMAP

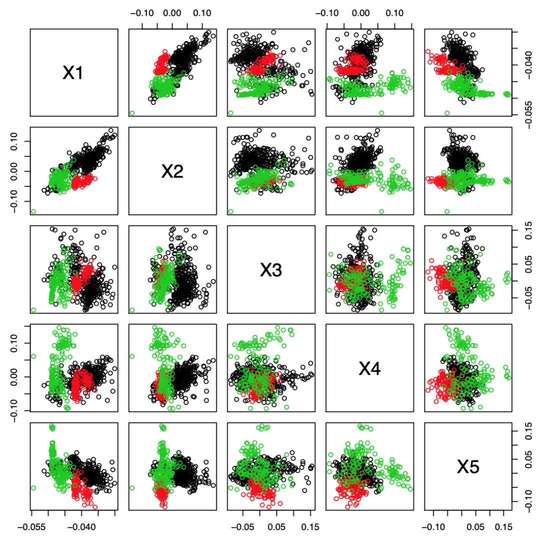


(l) Yan et al with UMAP

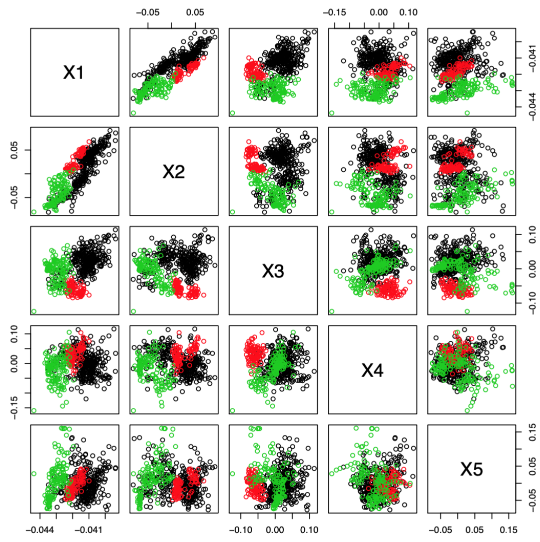
Figure 5.5: A visual display of dimensionality reduction methods with DGMM-HMCFL + t-SNE, MCFA + t-SNE, raw t-SNE and raw UMAP on the Viroli and McLachlan dataset, Olive Oil, and the Single-Cell Gene Expression datasets Li et al and Yan et al



(a) DGMM-CFL + PCA

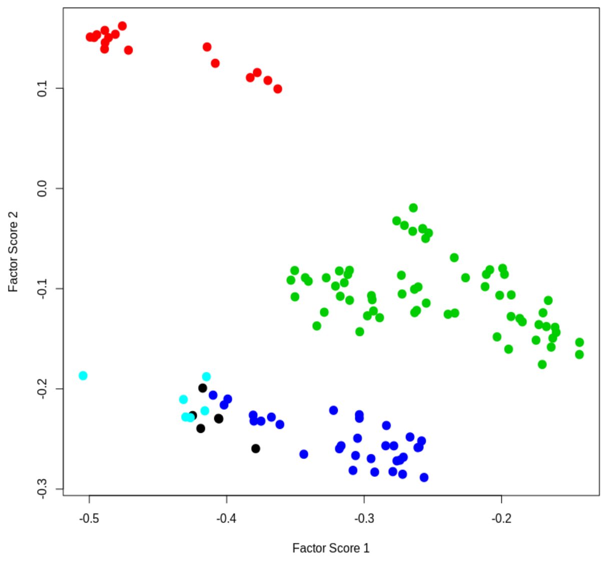


(b) DGMM-HMCFL + PCA

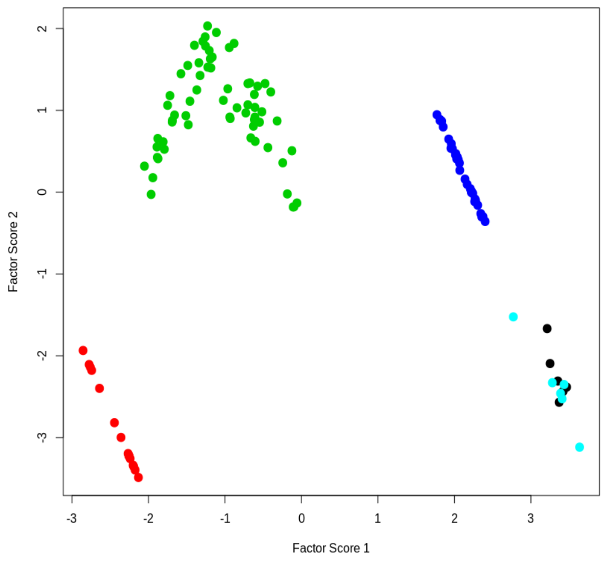


(c) DGMM-ICFL + PCA

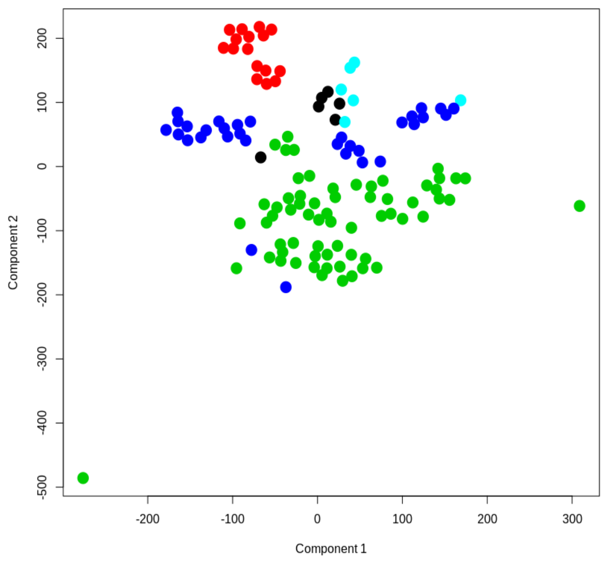
Figure 5.6: A visual display of dimensionality reduction methods for the Olive Oil dataset, showing each of the top 5 principal components



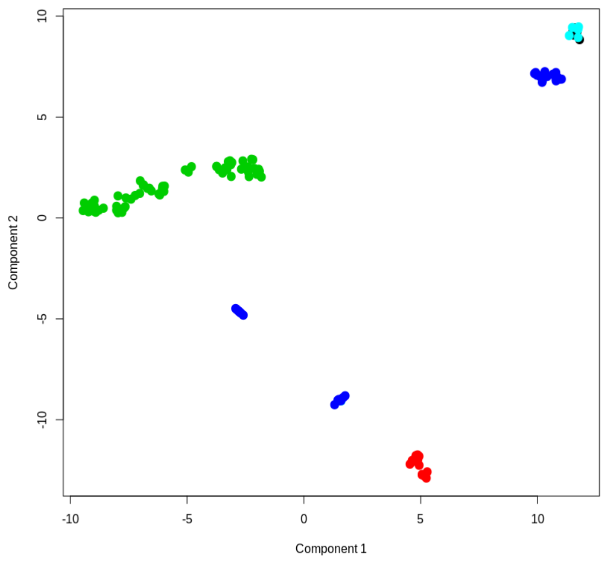
(a) Model-based 2nd layer Factor Scores from DGMM-CFL



(b) Model-based Factor Scores from MCFA



(c) tSNE components



(d) UMAP components

Figure 5.7: A visual display of the components and scores from DGMM-CFL, MCFA, tSNE and UMAP for the Goolam et al dataset.

# Chapter 6

# Discussion

The discussion will consist of three parts, an initial discussion on methods and a comparison of each with added explanation on hypotheses and evidence that supports each claim. Following, there is a section on the appeal of Deep Common Factor Analytic over shallower alternatives. Finally, future work will be addressed based on current claims supported by the results.

## 6.1 Method Comparison

Given the performance of the deep models: the DGMM and DGMM-CFL variants, over the shallower MCFA in Figure 5.3 and 5.4, the use of depth provides added benefit when finding a functional structure to fit the data generating process. Despite including more parameters in the deep model compared to the shallow model [4], the inclusion of parameter sharing across the mixtures in a interconnected structure, makes this method more efficient, and thus provides a highly optimised algorithm for the purpose of unsupervised clustering.

The naive implementation of DGMM-CFL was expected to perform worse compared to DGMM with hierarchical mixtures of CFL and DGMM with initial CFL, which is observed consistently across the ARI for both categories of datasets in Figure 5.3 and 5.4 (the baseline datasets and the single-cell expression datasets). This was expected, due to the linear transform of the location parameter of the very last latent factor in DGMM-CFL, compared to the non-linear transforms of DGMM with hierarchical mixtures of CFL and DGMM with initial CFL.

On the baseline datasets, the naive implementation of the deep DGMM-CFL consistently outperforms the shallow MCFA, with a similar structure with the only major difference being depth. Furthermore, The DGMM-CFL can have each layer be fully visualised, with the final layer being a transformation of all layers. As seen in Figure 5.7, DGMM-CFL is thus an improvement over MCFA and should be considered as an addition to the factor analytic family of methods.

Furthermore, we expected DGMM with initial CFL to perform on par with DGMM, but it falls short of this benchmark. This is because it is a model that contains DGMM as a subset (see Chapter 3 for more details). Given that the second layer in DGMM with initial CFL is always the main layer (where the DGMM subset begins), it is surprising it is fairly competitive with DGMM and DGMM with hierarchical mixtures of CFL, given it is not a fully ’deep’ model compared to a two layer DGMM or DGMM with hierarchical mixtures of CFL. To explain why, the first layer of the DGMM with initial CFL is of a MCFA, and thus, is expected to provide a dimensionality reduction. Therefore, one could expect that a two-layer DGMM with initial CFL acts as an initial dimension reduction via MCFA, and then the second layer being a standard MFA. Given this the two-layer DGMM with initial CFL is actually a factor analytic representation of a 1-layer DGMM with an initial dimension reduction. The jointly learned model shows the improved fit with increasing depth, despite only consisting of a 1-layer DGMM with initial dimension reduction by the MCFA.

Overall, DGMM compares well with the state of the art method in single-cell gene expression data, SC3, a consensus based unsupervised clustering method. The DGMM-CFL variants, while not as competitive, provide a good solution to a model that provides better interpretation of the model through visualisation, at the expense of flexibility and fit.

## 6.2 Improvements from Deep Factor Analytic methods

It has now been shown in Figure 5.3 and 5.4, that DGMM performs consistently well, better than the new methods in unsupervised clustering, where this is followed by the DGMM with hierarchical mixtures of CFL variant. The DGMM with hierarchical mixtures of CFL is also comparable to MCFA while also being more flexible, as shown in the improvements made in the Wine and Ecoli datasets. Although the DGMM has greater ability to fit a variety of models, it is not easy to visualise the results on a standard two dimensional plot via a common projection, similar to the DGMM-CFL variants and MCFA.

It should be stated that the visualisation of the model only takes into account the first layer of each deep DGMM-CFL method. Thus, the visualisation is not fully representative of every deep layer, where only the initial MCFA layer is visualised. However, for the naive DGMM-CFL implementation, it is possible to visualise both layers. As seen in Figure 5.7, this visualisation is shown to improve upon the t-SNE and UMAP methods, both state of the art in their respective areas, and provides a better visualisation given that it is a true ’deep’ MCFA, where each layer has only a single common factor loading (see Chapter 3 for more details).

In Figure 5.7, the benefit of using a model-based method for visualisation compared to optimisation based (t-SNE) or topological based methods (UMAP) is shown. It is evident that the deeper, second layer two dimensional DGMM-CFL visualisation in Fig 5.7a compares well against the shallow two dimensional MCFA model. Note that both of these methods compare well with t-SNE and UMAP, with t-SNE’s force-layout providing a visualisation of the data with non-overlapping points. Regardless, the naive DGMM-CFL implementation and extraction of the two-dimensional second layer provides better support for the clusters, in addition to a decent visualisation of those clusters.

To add further support to the notion of the DGMM-CFL designs (with all methods having an initial MCFA layer that provides a visualisation technique) the deep method DGMM with hierarchical mixtures of CFL performed consistently better across the datasets than its' shallow counterpart, MCFA in Figure 5.3 and 5.4 for the adjusted Rand Index. Furthermore, it was able to pick out the true signal in Figure 5.2, of the Deng et al dataset, showing the trajectory of embryonic function from an early stage embryo, to a late stage embryo. It was shown that all methods, including MCFA, naive DGMM-CFL, DGMM with hierarchical mixtures of CFL, and DGMM with initial CFL were able to distinguish the signal and provide a good representation of this biological function, supporting the notion of these novel methods through an improved unsupervised clustering method over MCFA, with a similar visualisation ability.

The convergence of all the implemented novel methods in Figure 5.1 provides support to the conclusions drawn. We expect that the EM algorithm to converge monotonically, improving the log-likelihood. This is visualised across all the methods and thus strengthens the current claims.

## 6.3 Future work

As discussed in Section 6.1, it is highly important in the next experiment that the DGMM with initial CFL is ran with three layers to be comparable with a two layer DGMM - a joint DGMM with initial CFL model with an initial MCFA layer, followed by a two layer DGMM. Given the current results, this could provide an improvement over DGMM with hierarchical mixtures of CFL, and is expected to provide much more support than the naive DGMM-CFL, while also providing the ability to visualise the first layer.

Another extension would be to have the current two-layer DGMM with initial CFL (consisting of the first MCFA layer, than a MFA layer) be used repeatedly in a sequential manner. The idea stems from the supporting notion that a jointly learned dimension reduction through a common factor projection via MCFA (where the latent variables are taken deeper) provides consistently better results on the baseline data over MCFA. Jointly learning the parameters, and then stacking the 2-layer DGMM with initial CFL design could provide an improved fit that projects the dimensionally reduced interpretation of each factor analytic layer repeatedly, similar to a deep multi-layer perceptron, and be a competitive alternative. Here, it would be wise and sensible to take the initial MCFA layer of each stack to have very few mixture components (perhaps 1, not more than 2), as the number of model parameters could exponentially explode.

Furthermore, the S-EM algorithm here is used to enable the DGMM to learn the parameters of each layer via the sampled posteriors. An alternative to the S-EM algorithm would be to reduce the stochastic inputs in the learning algorithm (the posterior samples), and replace it with a gradient-descent based method. This could provide a challenge, given the full form of the marginals, especially the marginal of the fitted $\hat{y}$, albeit, this could be learned layer by layer, similar to the set-up of the current algorithm.

# Chapter 7

# Conclusion

The use of depth in a hierarchical setting of factor analysers within Deep Gaussian Mixture Models provides the flexibility and improved fit for a wide range of data generating processes. The novel introduction of a series of deep Mixtures of Common Factor Analysers provides an improved visualisation tool, similar to other state of the art methods. Furthermore, other novel methods based on the original Deep Mixture Model with Common Factor Loadings are introduced, and provide both strong unsupervised clustering performance, and well-defined visualisation of the clusters. Through the current and future work planned, the aim is to improve both the visualisation and clustering performance, given that this has been shown to be a successful method capable of unifying dimensionality reduction, unsupervised clustering, and visualisation.

# Bibliography

[1] G. J. McLachlan, T. Krishnan, The EM algorithm and extensions, 2nd Edition, Wiley series in probability and statistics, Wiley-Interscience, Hoboken, N.J., 2008.

[2] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, Deep Learning (2016) 1–775.

[3] O. B. Poirion, X. Zhu, T. Ching, L. Garmire, Single-cell transcriptomics bioinformatics and computational challenges, Frontiers in Genetics 7 (2016). doi:ARTN16310.3389/fgene.2016.00163.

[4] C. Viroli, G. J. McLachlan, Deep gaussian mixture models, Statistics and Computing 29 (1) (2019) 43–51.

[5] S. Konishi, G. Kitagawa, Information Criteria and Statistical Modeling, 2008. doi:10.1007/978-0-387-71887-3.

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

[7] J. Baek, G. J. McLachlan, L. K. Flack, Mixtures of factor analyzers with common factor loadings: Applications to the clustering and visualization of high-dimensional data, Ieee Transactions on Pattern Analysis and Machine Intelligence 32 (7) (2010) 1298–1309. doi:10.1109/Tpami.2009.149.

[8] Z. Ghahramani, G. E. Hinton, The em algorithm for mixtures of factor analyzers, Tech. rep. (1997).

[9] Y. Tang, R. Salakhutdinov, G. E. Hinton, Deep mixtures of factor analysers, CoRR abs/1206.4635 (2012). arXiv:1206.4635.

[10] X. Yang, K. Huang, R. Zhang, A. Hussain, Introduction to Deep Density Models with Latent Variables, Springer International Publishing, Cham, 2019, pp. 1–29. doi:10.1007/978-3-030-06073-2_1.

[11] X. Yang, K. Huang, R. Zhang, Deep mixtures of factor analyzers with common loadings: A novel deep generative approach to clustering, in: D. Liu, S. Xie, Y. Li, D. Zhao, E.-S. M. El-Alfy (Eds.), Neural Information Processing, Springer International Publishing, pp. 709–719.

[12] X. Yang, K. Huang, R. Zhang, J. Y. Goulermas, A novel deep density model for unsupervised learning, Cognitive Computation 11 (6) (2019) 778–788. doi:10.1007/s12559-018-9566-9.

[13] L. van der Maaten, G. Hinton, Visualizing data using t-sne, Journal of Machine Learning Research 9 (2008) 2579–2605.

[14] E. Becht, L. McInnes, J. Healy, C. A. Dutertre, I. W. H. Kwok, L. G. Ng, F. Ginhoux, E. W. Newell, Dimensionality reduction for visualizing single-cell data using umap, Nature Biotechnology 37 (1) (2019) 38–+. doi:10.1038/nbt.4314.

[15] D. Dua, C. Graff, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml

[16] M. Goolam, A. Scialdone, S. J. L. Graham, I. C. Macaulay, A. Jedrusik, A. Hupalowska, T. Voet, J. C. Marioni, M. Zernicka-Goetz, Heterogeneity in oct4 and sox2 targets biases cell fate in 4-cell mouse embryos, Cell 165 (1) (2016) 61–74. doi:10.1016/j.cell.2016.01.047.

[17] S. Darmanis, S. A. Sloan, Y. Zhang, M. Enge, C. Caneda, L. M. Shuer, M. G. H. Gephart, B. A. Barres, S. R. Quake, A survey of human brain transcriptome diversity at the single cell level, Proceedings of the National Academy of Sciences of the United States of America 112 (23) (2015) 7285–7290. doi:10.1073/pnas.1507125112.

[18] H. P. Li, E. T. Courtois, D. Sengupta, Y. Tan, K. H. Chen, J. J. L. Goh, S. L. Kong, C. Chua, L. K. Hon, W. S. Tan, M. Wong, I. Cima, M. H. Tan, L. J. K. Wee, A. M. Hillmer, I. B. Tan, P. Robson, S. Prabhakar, Reference component analysis of single-cell transcriptomes elucidates cellular heterogeneity in human colorectal tumors (vol 50, pg 1754, 2018), Nature Genetics 50 (12) (2018) 1754–1754. doi:10.1038/s41588-018-0299-1.

[19] L. Y. Yan, M. Y. Yang, H. S. Guo, L. Yang, J. Wu, R. Li, P. Liu, Y. Lian, X. Y. Zheng, J. Yan, J. Huang, M. Li, X. L. Wu, L. Wen, K. Q. Lao, R. Q. Li, J. Qiao, F. C. Tang, Single-cell rna-seq profiling of human preimplantation embryos and embryonic stem cells, Nature Structural Molecular Biology 20 (9) (2013) 1131–+. doi:10.1038/nsmb.2660.

[20] Q. Deng, D. Ramsköld, B. Reinius, R. Sandberg, Single-cell rna-seq reveals dynamic, random monoallelic gene expression in mammalian cells, Science 343 (6167) (2014) 193–196. arXiv:https://science.sciencemag.org/content/343/6167/193.full.pdf, doi:10.1126/science.1245316.

[21] A. M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, V. Li, L. Peshkin, D. A. Weitz, M. W. Kirschner, Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells, Cell 161 (5) (2015) 1187–1201. doi:10.1016/j.cell.2015.04.044.

[22] V. Y. Kiselev, K. Kirschner, M. T. Schaub, T. Andrews, A. Yiu, T. Chandra, K. N. Natarajan, W. Reik, M. Barahona, A. R. Green, M. Hemberg, Sc3: consensus clustering of single-cell rna-seq data, Nature Methods 14 (5) (2017) 483–+. doi:10.1038/Nmeth.4236.

[23] T. P. Quinn, M. F. Richardson, D. Lovell, T. M. Crowley, propr: An r-package for identifying proportionally abundant features using compositional data analysis, Sci Rep 7 (1) (2017) 16252. doi:10.1038/s41598-017-16520-0.

[24] M. A. Skinnider, J. W. Squair, L. J. Foster, Evaluating measures of association for single-cell transcriptomics, Nature Methods 16 (5) (2019) 381–+. doi:10.1038/s41592-019-0372-4.

# Appendix A

# Appendix

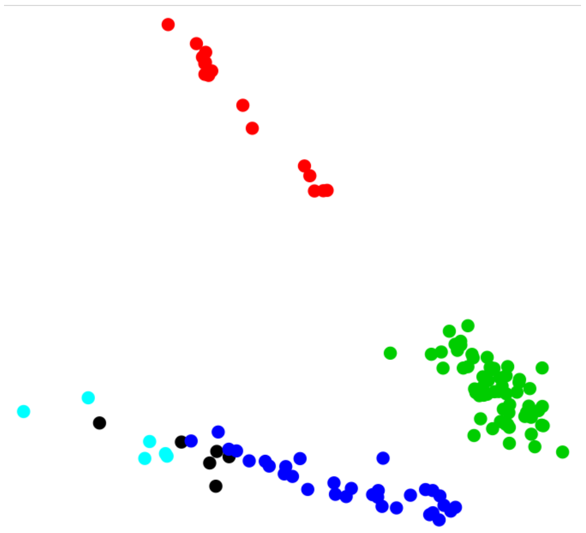
The following plots follow a different protocol when compared to the analysis in Figure 5.7. The protocol is identical to SC3, with the addition of the Phi-S measure from propr [23], evaluated as a top performer from hierarchical clustering [24].

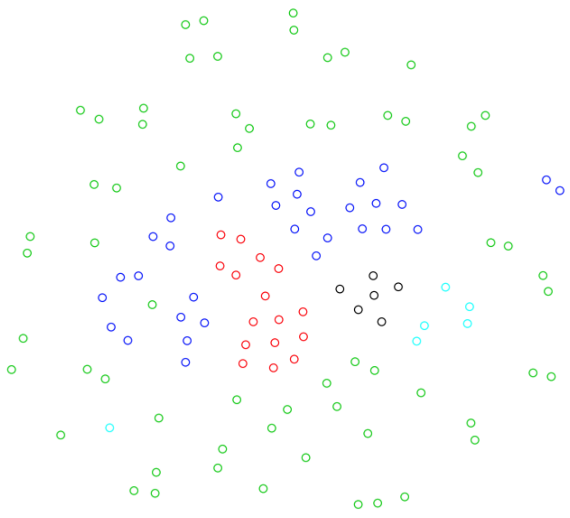
## A.1 Preprocessing

Across the baseline datasets, it is evident from the models that each DGMM and DGMM-CFL design expects pre-processed sampled observations scaled to the standard normal distribution, that is $Z \sim N(0,I)$.

For the single-cell gene expression datasets, a similar protocol to SC3 is followed.

1. Filter the raw, unscaled, yet sparse dataset such that only genes remain with less than 6% of zeros, that is 94% of count values contain a value equal to or greater than 1.
2. The filtered, unscaled, original dataset is taken and log-transformed with a pseudo-count of 1 without any further transformations.
3. The cells (where subsets contain the clusters of interest) from the now log-transformed dataset are taken and the following pair-wise measures are calculated: Pearson, Spearman, Euclidean and a proportionality based measure, Phi-S (in Phi-S the raw, unscaled, untransformed count data is used).
4. These are then run through Principal Component Analysis to extract the top components; usually the eigenvalues in descending order that contain 70% of the variance have their corresponding eigenvectors taken. However, for computational purposes, the top 5 components are taken.
5. Using these principal components across the 4 different measures, Pearson, Spaerman, Euclidean and Phi-S, scale these to standard normal and put these through the DGMM and CFL variants.


(a) Goolam et al with DGMM-CFL


(b) Goolam et al with t-SNE

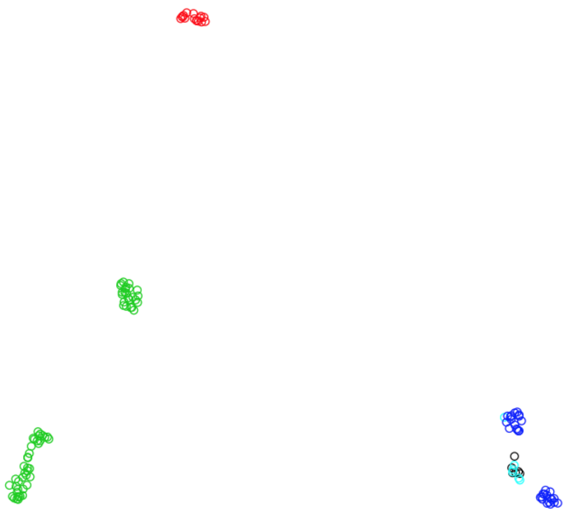
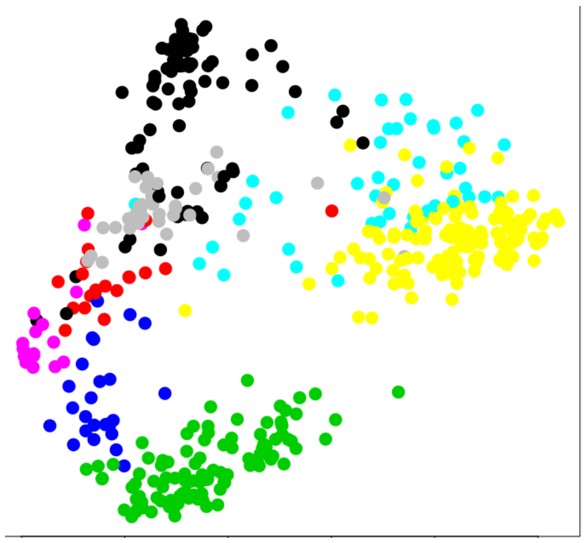
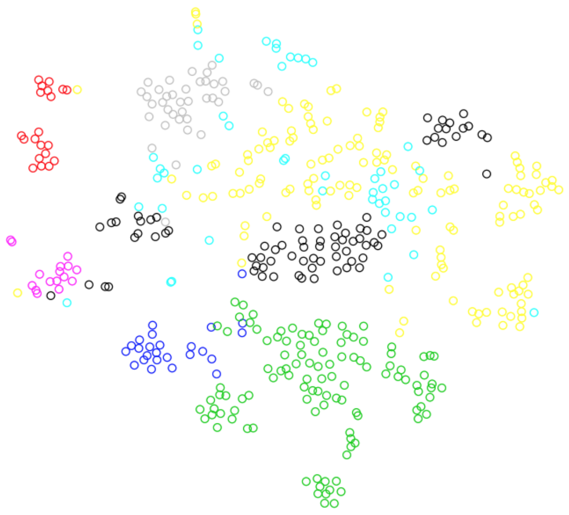

(c) Goolam et al with UMAP

Figure A.1: A visual display of dimensionality reduction methods with a deep 2-layer DGMM-CFL, raw t-SNE and raw UMAP on the Goolam et al Single-Cell Gene Expression datasets. This follows the SC3 protocol exactly, with the addition of the Phi-S measure (when compared to Figure 5.7)


(a) Darmanis et al with DGMM-CFL


(b) Darmanis et al with t-SNE

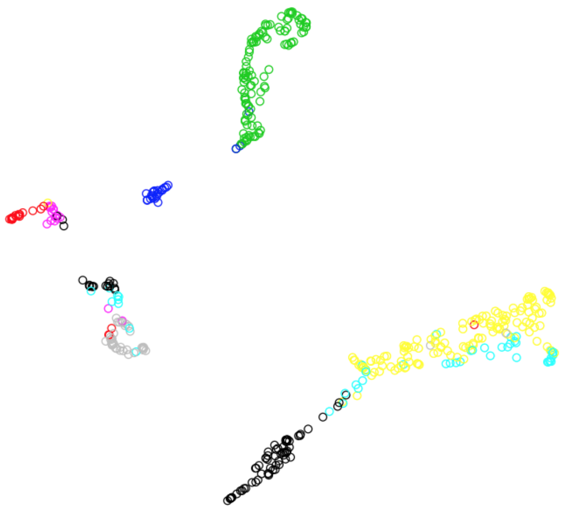
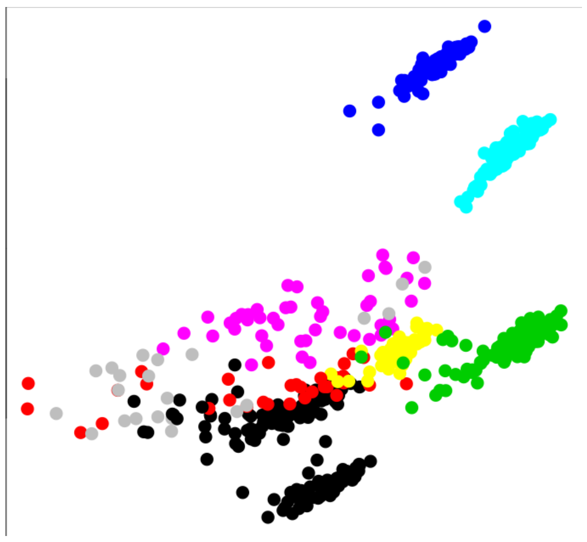
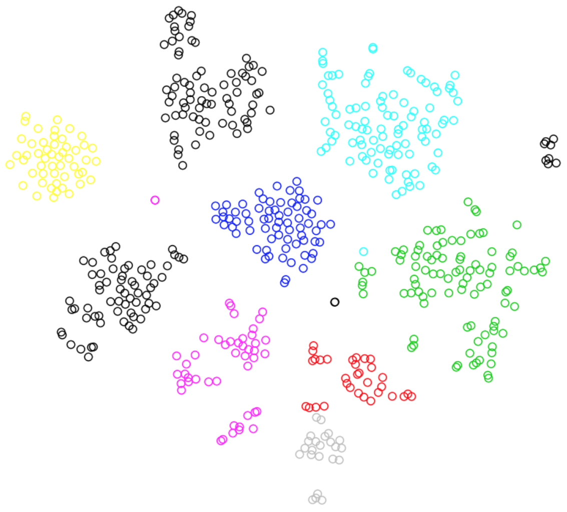

(c) Darmanis et al with UMAP

Figure A.2: A visual display of dimensionality reduction methods with a deep 2-layer DGMM-CFL, raw t-SNE and raw UMAP on the Darmanis et al Single-Cell Gene Expression datasets. This follows the SC3 protocol exactly, with the addition of the Phi-S measure (when compared to Figure 5.7)


(a) Li et al with DGMM-CFL


(b) Li et al with t-SNE

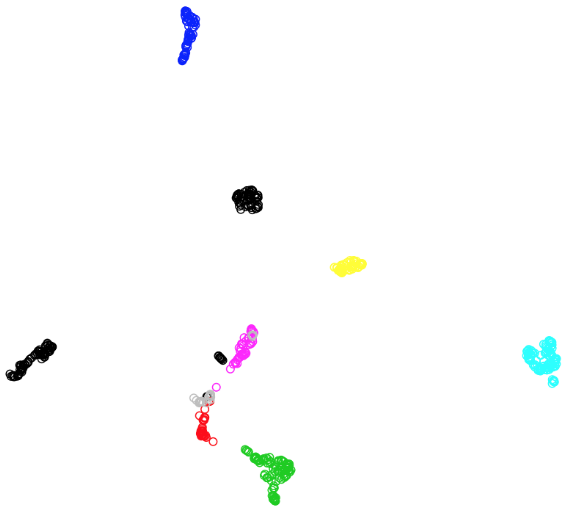
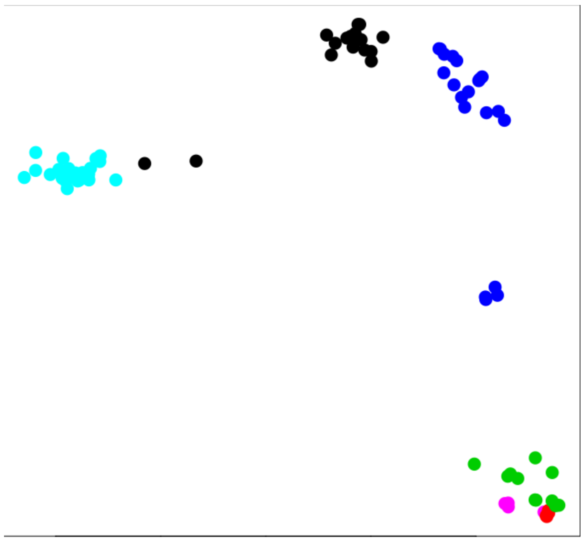
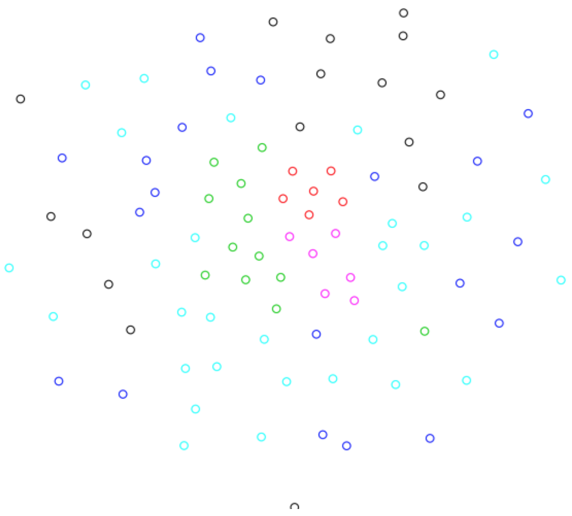

(c) Li et al with UMAP

Figure A.3: A visual display of dimensionality reduction methods with a deep 2-layer DGMM-CFL, raw t-SNE and raw UMAP on the Li et al Single-Cell Gene Expression datasets. This follows the SC3 protocol exactly, with the addition of the Phi-S measure (when compared to Figure 5.7)


(a) Yan et al with DGMM-CFL


(b) Yan et al with t-SNE

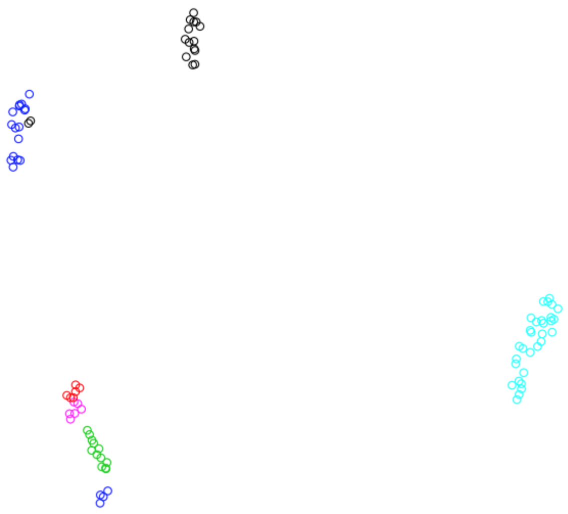
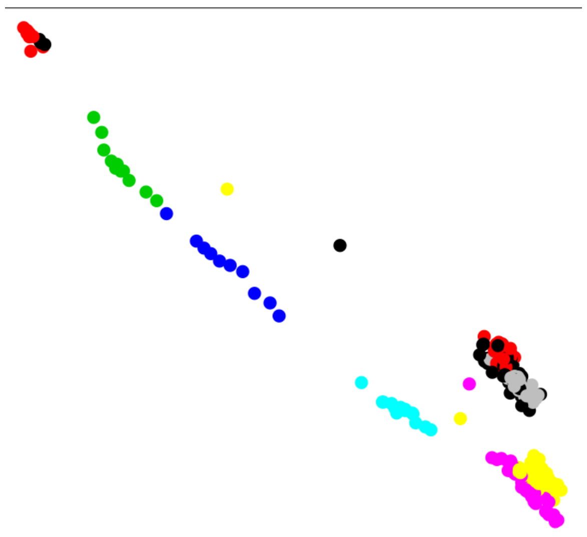
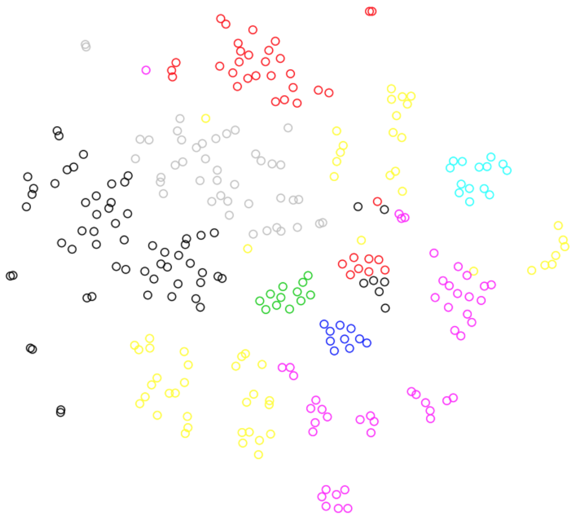

(c) Yan et al with UMAP

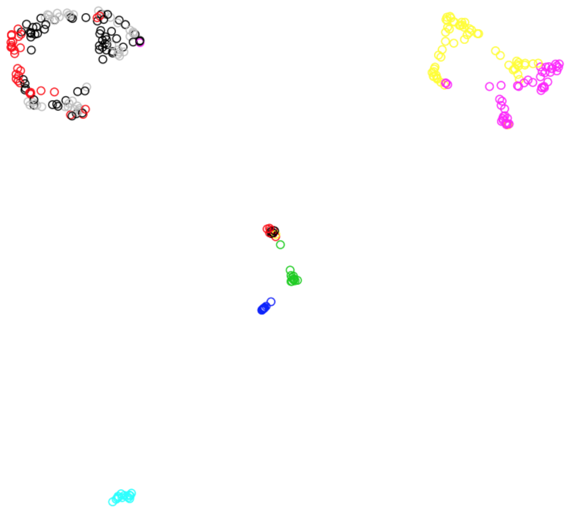
Figure A.4: A visual display of dimensionality reduction methods with a deep 2-layer DGMM-CFL, raw t-SNE and raw UMAP on the Yan et al Single-Cell Gene Expression datasets. This follows the SC3 protocol exactly, with the addition of the Phi-S measure (when compared to Figure 5.7)



(a) Deng et al with DGMM-CFL



(b) Deng et al with t-SNE



(c) Deng et al with UMAP

Figure A.5: A visual display of dimensionality reduction methods with a deep 2-layer DGMM-CFL, raw t-SNE and raw UMAP on the Deng et al Single-Cell Gene Expression datasets. This follows the SC3 protocol exactly, with the addition of the Phi-S measure (when compared to Figure 5.7)