Appendix: Gaussian Mixture Copulas for High-Dimensional Clustering and Dependency-based Subtyping

Siva Rajesh Kasa [1], **Sakyajit Bhattacharya** [2] **and Vaibhav Rajan** [1]
[1]Department of Information Systems and Analytics, School of Computing, National University of Singapore, Singapore and
[2]TCS Innovation Labs, Kolkata, India.

A Gaussian Mixture Copula Model

In this section, we provide the relevant background on multivariate distributions, copulas and mixture models required for a self-contained description of GMCM. We state the main results without proofs that can be found in previous literature on copulas (Trivedi et al., 2007; Joe, 2014) and GMCM (Tewari et al., 2011; Bilgrau et al., 2016; Bhattacharya and Rajan, 2014). We also compare GMCM with related mixture- and copula-based models.

Distributions and Densities

Consider n i.i.d. instances of p-dimensional data, $\mathbf{X} = [x_{ij}]_{n\times p} = (\mathbf{X}_1, \ldots, \mathbf{X}_p)$, where $\mathbf{X}_j$ denotes the j^{th} dimension of data, i denotes the observation ($i = 1, \ldots, n$), and j denotes the dimension ($j = 1, \ldots, p$). We use bold font for vectors and matrices to distinguish them from scalars. Single subscript denotes the index along dimension, unless specified otherwise.

The joint Cumulative Distribution Function (CDF) of $\mathbf{X}$ is defined by

$$F(\mathbf{x}) = P(\mathbf{X} \leq \mathbf{x}) = P(\mathbf{X}_1 \leq \mathbf{x}_1, \ldots, \mathbf{X}_p \leq \mathbf{x}_p).$$

For continuous-valued $\mathbf{X}$, a non-negative Probability Density Function (PDF), f, exists, such that

$$F(\mathbf{x}) = \int_{-\infty}^{\mathbf{x}} f(\mathbf{u})d\mathbf{u} \text{ and } \int_{-\infty}^{\infty} f(\mathbf{u})d\mathbf{u} = 1$$

where the integrals are multidimensional, i.e., $\int_{-\infty}^{\mathbf{x}} f(\mathbf{u})d\mathbf{u} = \int_{-\infty}^{x_1} \ldots \int_{-\infty}^{x_p} f(u_1, \ldots, u_p)du_1 \ldots du_p$.

The marginal CDF of $\mathbf{X}_j$, denoted by F_j, is defined by

$$F_j(\mathbf{x}) = P(\mathbf{X}_j \leq \mathbf{x}) = F(\infty, \ldots, \infty, \mathbf{X}_j \leq \mathbf{x}, \infty, \ldots, \infty).$$

The j^{th} marginal PDF, f_j, can be obtained by 'marginalizing out' all other dimensions from the joint PDF:

$$f_j(\mathbf{x}) = \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} f(u_1, \ldots, u_p)du_1 \ldots, du_{j-1}, du_{j+1}, \ldots, du_p.$$

Note that a CDF (including the marginal CDF) maps a random variable to a scalar that is uniformly distributed in the interval $[0, 1]$. Thus, the inverse CDF or quantile function is well-defined: $F_j^{-1}(v) = \inf\{X_{ij} : F_j(X_{ij}) \geq v\}$, where $v \sim unif(0, 1)$. This enables a transformation of a scalar, uniformly distributed between 0 and 1, to a unique value of the random variable X_{ij}, viz., the minimum amongst all those values whose marginal CDF value exceeds v.

Copulas

While the marginal CDF values, along each dimension, is uniformly distributed, the joint distribution of all p marginal CDFs is *not* uniform in a p-dimensional hypercube. This distribution is modeled by a copula. A p–dimensional copula is a multivariate distribution function $C : [0, 1]^p \to [0, 1]$, defined on random variables obtained through CDF transformations, $\mathbf{U}_j = F_j(\mathbf{X}_j)$.

A theorem by Sklar (1959) shows that copulas can uniquely characterize continuous joint distributions:

Theorem 1 (Sklar's Theorem) *Let F be a joint distribution function with marginals $F_1, \ldots, F_p$. Then there exists a copula $C : [0, 1]^p \to [0, 1]$ such that*

$$F(\mathbf{x}) = C(F_1(\mathbf{x}_1), \ldots, F_p(\mathbf{x}_p)).$$

If the marginal distributions are continuous, then this copula is unique. Conversely, if C is a copula and $F_1, \ldots, F_p$ are univariate distribution functions, then F as defined above is a multivariate distribution function with marginals $F_1, \ldots, F_p$.

It can be shown that the corresponding joint density is given by the product of the individual marginal densities f_j and the *copula density* c:

$$f(\mathbf{x}) = c(F_1(\mathbf{x}_1), \ldots, F_p(\mathbf{x}_p))\ \Pi_{j=1}^{p} f_j(\mathbf{x}_j). \quad \text{(A.1)}$$

Rearranging the terms above, we note that the copula density can be expressed in terms of the joint density and the marginals:

$$\begin{aligned} c(\mathbf{U}_1, \ldots, \mathbf{U}_p) &= c(F_1(\mathbf{x}_1), \ldots, F_p(\mathbf{x}_p)) \\ &= \frac{f(\mathbf{x})}{\Pi_{j=1}^{p} f_j(\mathbf{x}_j)}. \end{aligned} \quad \text{(A.2)}$$

Copula densities can be defined by suitable choices of the joint density f. For instance, the Gaussian copula density, c_ϕ is defined by choosing the multivariate normal density ϕ with normal marginal densities ϕ_j in equation A.2: $c_\phi = \frac{\phi(\mathbf{x})}{\Pi_{j=1}^{p}\phi_j(\mathbf{x}_j)}$. Note that using the Gaussian copula density, c_ϕ, we can construct various joint distributions, called *Meta-Gaussian distributions*, using different choices of marginal densities in equation A.1. Meta-Gaussian distributions can model non-Gaussian data but cannot model asymmetric and tail dependencies (see fig. 1).

Sklar's theorem and equation A.1 show how copulas enable flexible modeling of multivariate data by decoupling the specification of marginals and the dependence structure (the latter modeled by a parametric copula family). The modeler has the choice of choosing each marginal density and copula family independently from each other. Copulas are invariant to monotonic transformations of the random variables, i.e., if $\mathbf{X}$ is from a multivariate distribution with marginals f_j and copula c, then the dependence structure of monotonically transformed variable $t(\mathbf{X})$ is also given by c.

Copula-based models can also be viewed as generative models defined on the CDF-transformed data, $\mathbf{U}_j = F_j(\mathbf{X}_j)$. The generative model for the Gaussian copula is defined through a Gaussian distribution on the latent CDF transformations (Hoff et al., 2007):

$$\mathbf{X}_j = F_j^{-1}(\mathbf{U}_j);\ \ \mathbf{U}_j = \Phi_j(\mathbf{Y}_j);\ \ \mathbf{Y} \sim \Phi(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where Φ_j denotes the j^{th} marginal CDF of the multivariate normal distribution and Φ is the CDF of a multivariate normal with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

Parameter Inference. Standard Maximum Likelihood (ML) inference requires specifying a parametric family for each marginal density and inferring the parameters simultaneously with the copula parameters which is computationally expensive for even moderate dimensions. The two–step IFM procedure (Joe, 2014) requires fitting a marginal to each feature and then using the CDF transformation to obtain the input data (in the range $[0, 1]$) for an ML estimation of copula parameters. In reality, marginals are usually not known and may be difficult to estimate correctly. Also, we may be interested solely in the dependence structure, including in this work, and may want to circumvent marginal parameter estimation. In such cases, a *semiparametric approach* is to use rank–transformed scaled empirical marginals to estimate the copula parameters. In the rank transformation, each data element x_{ij}, of the feature vector X_j is transformed to $R(x_{ij}) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(X_{ij} \leq x_{ij})$, the scaled rank of x_{ij} in all the observations of the feature, X_j. The scaled rank transformed features lie in the range $[0, 1]$ and can be used directly to estimate the copula parameters. The resulting pseudo–likelihood estimator is consistent under the condition that the margins are continuous (Genest et al., 1995).

Finite Mixture Models

The PDF of a G-component finite mixture model is given by

$$f(\mathbf{x}) = \sum_{g=1}^{G} \pi_g f_g(\mathbf{x}; \theta_g) \tag{A.3}$$

with mixing proportions $\pi_g > 0$ such that $\sum_{g=1}^{G} \pi_g = 1$ and component densities f_g parameterized by θ_g.

A common choice of the component density is the Multivariate Normal, ϕ, that is parameterized by component-specific mean vectors, $\boldsymbol{\mu}_g$, and covariance matrices, $\boldsymbol{\Sigma}_g$. This yields the popular Gaussian Mixture Model (GMM) that we denote by $\mathcal{G}(\boldsymbol{\vartheta})$, where $\boldsymbol{\vartheta} = (\pi_1, ...\pi_G, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_G, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_G)$ denotes the complete set of parameters.

Model	Covariance Parameters
CCC	$[pq - q(q-1)/2] + 1$
CCU	$[pq - q(q-1)/2] + p$
CUC	$[pq - q(q-1)/2] + G$
CUU	$[pq - q(q-1)/2] + Gp$
UCC	$G[pq - q(q-1)/2] + 1$
UCU	$G[pq - q(q-1)/2] + p$
UUC	$G[pq - q(q-1)/2] + G$
UUU	$G\{pq - q(q-1)/2\} + Gp$

Table A.1: Parsimonious Gaussian Mixture Models

Parsimonious Gaussian Mixture Models

With p-dimensional data and with G components in a mixture there are exactly $(G-1) + Gp + Gp(p+1)/2$ free parameters. Unless p is very small, most of these parameters are in the component covariance matrices $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_G$. To reduce the computational cost of the estimation, special families of covariance structures have been introduced that impose constraints upon the constituent parts of the decomposition of $\boldsymbol{\Sigma}_g$. The Parsimonious Gaussian Mixture family, or PGMM (McNicholas and Murphy, 2008) are a family wherein the covariance structure is assumed to be of the form $\boldsymbol{\Sigma}_g = \boldsymbol{\Lambda}_g\boldsymbol{\Lambda}_g' + \boldsymbol{\Omega}_g$, where $\boldsymbol{\Omega}_g$ is a diagonal matrix of white noise, $\boldsymbol{\Lambda}_g$ is a p x q matrix of factor loadings and q is the number of latent factors. Generally $q < p$. The loading and noise terms can be constrained to be equal or unequal across groups to give a collection of eight parsimonious covariance structures shown in table A.1. The number of parameters in these eight families range from $pq - q(q-1)/2 + 1$ to $G(pq - q(q-1)/2 + p)$. See (McNicholas and Murphy, 2008) for more details.

GMCM

In GMCM, the dependence is obtained from a Gaussian Mixture Model. Let $\Psi_j(\boldsymbol{\vartheta})$ and $\psi_j(\boldsymbol{\vartheta})$ denote the j^{th} marginal CDF and PDF respectively, of $\mathcal{G}(\boldsymbol{\vartheta})$. Note that $\mathbf{U}_j = F_j(\mathbf{X}_j)$. From equation A.2, the GMCM copula density is given by:

$$c_{\mathcal{G}}(\mathbf{U}; \boldsymbol{\vartheta}) = \frac{\mathcal{G}(\Psi_j^{-1}(\mathbf{U}))}{\prod_{j=1}^{p} \psi_j(\Psi_j^{-1}(\mathbf{U}_j))} \tag{A.4}$$

The generative model of GMCM is specified by, $\forall j \in \{1, \ldots, p\}$:

$$\mathbf{X}_j = F_j^{-1}(\mathbf{U}_j);\ \ \mathbf{U}_j = \Psi_j(\mathbf{Y}_j);\ \ \mathbf{Y} \sim \mathcal{G}(\boldsymbol{\vartheta}). \tag{A.5}$$

Note that $\mathbf{U}_j = F_j(\mathbf{X}_j) = \Psi_j(\mathbf{Y}_j)$ and so, $\mathbf{Y}_j = \Psi_j^{-1}(\mathbf{U}_j)$. Thus, the likelihood of n i.i.d. samples from GMCM can be expressed in terms of the latent variable Y:

$$\prod_{i=1}^{n} \frac{\sum_{g=1}^{G} \pi_g \phi(\mathbf{y}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)}{\prod_{j=1}^{p} \psi_j(y_{ij} \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g))}.$$

For clustering, GMCM can be used to obtain cluster labels $l \in 1, \ldots, G$ through a semiparametric MAP estimate $\arg\max_l P(l = g|\boldsymbol{\vartheta}, \mathbf{X})$, from rank-transformed marginals in the data as estimates of $\mathbf{U}_j$ (Bhattacharya and Rajan, 2014). To find the number of clusters, the BIC-based model selection criterion was empirically found to be effective.

GMM, GMCM and Mixture of Copulas

The Gaussian Mixture Model (GMM) is a finite mixture model (equation A.3), where each component density is a multivariate normal distribution, $f_g = \phi$. A mixture of copulas (Kosmidis and Karlis, 2016) is also a finite mixture model where each component density is defined using a copula (equation A.1), i.e. $f_g = c(F_1(\mathbf{x}_1), \ldots, F_p(\mathbf{x}_p))\ \Pi_{j=1}^{p} f_j(\mathbf{x}_j)$. This is a very flexible model that encompasses all known mixture models and allows construction of new mixture models through the choice of arbitrary copula density and marginals in each component.

GMCM, is a specific copula model, wherein the distribution of the CDF-transformed data ($\mathbf{U}_j$) is modeled with a GMM. Like any copula model (including mixture of copulas), each marginal density can be independently specified. The higher modeling flexibility of mixture of copulas comes at the cost of more difficult parameter inference. The M-step of the EM algorithm proposed in (Kosmidis and Karlis, 2016) for inference does not use a closed-form expression and relies on numerical methods that are slow to converge. In contrast, our inference algorithm, HD-GMCM, uses closed-form expressions within an AECM algorithm.

The most important difference is with respect to application on high-dimensional data. Kosmidis and Karlis (2016) state that their method cannot be directly applied to high-dimensional data, since most copula families cannot be used to fit high-dimensional data. Our method has been specifically designed for high-dimensional data and empirically performs well for clustering.

B Proof of Equation 5

The CDF of a univariate Gaussian distribution $\mathcal{N}(\mu, \sigma)$ for a point y is given by

$$u = \int_{-\infty}^{y} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}} dy$$

$$u = 0.5 + \int_{\mu}^{y} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}} dy$$

$$\frac{du}{dy} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$$

$$\frac{d^2u}{dy^2} = \frac{1}{\sqrt{2\pi}\sigma} \left(-\frac{(y-\mu)^2}{2\sigma^2} \right) e^{-\frac{(y-\mu)^2}{2\sigma^2}}$$

Expanding u as a Taylor series around $y = \mu$, we get

$$u = 0.5 + \frac{du}{dy}\Big|_{x=\mu} (y-\mu) + \frac{1}{2}\frac{d^2u}{dy^2}\Big|_{x=\mu} (y-\mu)^2 +$$

(higher order terms)

Substituting the derivative terms in the above expansion,

$$u = 0.5 + \frac{1}{\sqrt{2\pi}\sigma_g}(y-\mu) +$$

(the second derivative goes to zero) + (higher order terms)

For a GMM, the above expression becomes

$$u \approx \sum_{g=1}^{G} \left(\pi_g \left(0.5 + \frac{1}{\sqrt{2\pi}\sigma}(y-\mu_g) \right) \right)$$

which yields

$$y \approx \left(\sum_{g=1}^{G} \frac{\pi_g}{\sigma_g\sqrt{2\pi}} \right)^{-1} \left[u - 0.5 + \left(\sum_{g=1}^{G} \frac{\pi_g \mu_g}{\sigma_g\sqrt{2\pi}} \right) \right].$$

C Mean Parameter Estimate in Algorithm 1

Proof The mean parameters are estimated in successive iterations. Denote by $\boldsymbol{\mu}^{(m)}$ the estimate of of $\boldsymbol{\mu}$ after m iterations. The penalty function $\mathcal{L}_{pen}$ is non-concave and singular at the origin. We locally approximate the penalty by a quadratic function by $\varphi(\boldsymbol{\mu}) \approx n\lambda_n \sum_{g=1}^{G} \pi_g \sum_{j=1}^{p_g} [|\mu_{gj}^{(m)} - c_j| + \frac{1}{2}\frac{\text{sign}\{\mu_{gj}^{(m)} - c_j\}}{\mu_{gj}^{(m)}}(\mu_{gj}^2 - {\mu^{(m)}}_{gj}^2)]$ as suggested in Khalili and Chen (2007). Here, c_j is the j^{th} marginal of the mean of all the data points. We can find the derivative of the penalization term $\frac{\partial\varphi}{\partial\mu_{gj}} = \text{sign}(\mu_{gj}^m - c_j)$ So, the mean estimate in our penalized likelihood case is obtained by setting

$$\hat{\boldsymbol{\Sigma}}_g^{-1} \sum_{i=1}^{n} \hat{z}_{ig}(\mathbf{y}_i - \boldsymbol{\mu}_g) - n\lambda_n \hat{\pi}_g \boldsymbol{\beta}_g = 0$$

where $\boldsymbol{\beta}_g$ is a vector with P elements, its jth element being $\text{sign}(\mu_{gj}^m - c_j)$. Solving the above equation for $\boldsymbol{\mu}_g$ yields

$$\boldsymbol{\mu}_g = \frac{\sum_{i=1}^{n} \hat{z}_{ig}\mathbf{y}_i}{\sum_{i=1}^{n} \hat{z}_{ig}} - n\lambda_n \hat{\pi}_g \boldsymbol{\Sigma}_g \boldsymbol{\beta}_g.$$

D Proof of Theorem 1

The complete likelihood $\vartheta^{(t)}$, given $y^{(t)}$ and $z^{(t)}$ is [1]

$$\mathcal{L}(\vartheta^{(t+1)}|y^{(t)}, z^{(t)}) = \prod_{i=1}^{N}\prod_{g=1}^{G} \left[\pi_{g^{t+1}} \Phi\left(\frac{\mathbf{y}_i^{(t)}}{\boldsymbol{\mu}_g^{(t+1)}, \boldsymbol{\Sigma}_g^{(t+1)}} \right) \right]^{z_{ig}^{(t)}}$$

Similarly, we can obtain $\mathcal{L}(\vartheta^{(t+1)}|y^{(t+1)}, z^{(t)})$

Now,

$$\log \mathcal{L}(\vartheta^{(t+1)}|\mathbf{y}^{(t+1)}, z^{(t)}) - \log \mathcal{L}(\vartheta^{(t+1)}|\mathbf{y}^{(t)}, z^{(t)}) =$$

$$- 0.5 \sum_{i=1}^{N}\sum_{g=1}^{G} \Bigg(z_{ig}^{(t)} \Bigg((\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)})^T \boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)}) \Bigg)$$

$$- z_{ig}^{(t)} \left((\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)})^T \boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)}) \right) \Bigg)$$

First, note that if A is 1×1, then $tr(A) = A$. Also, note that $tr(ABC) = tr(BAC)$. Using these facts, we can simplify the above expression to

[1] We have omitted *hat* on the estimates in this section to avoid clutter, e.g., we use $z^{(t)}$ instead of $\hat{z}^{(t)}$.

$$
\begin{aligned}
&- 0.5 \sum_{i=1}^{N} \sum_{g=1}^{G} \Bigg(z_{ig}^{(t)} \Bigg((\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)})^T \\
&\boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)}) \Bigg) \\
&- z_{ig}^{(t)} \Bigg((\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)})^T \boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)}) \Bigg) \Bigg) \\
&= - 0.5 \sum_{i=1}^{N} \sum_{g=1}^{G} \Bigg(z_{ig}^{(t)} tr \Bigg(\boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)})^T \\
&(\mathbf{y}_i^{(t+1)} - \boldsymbol{\mu}_g^{(t+1)}) \Bigg) \\
&- z_{ig}^{(t)} tr \left(\boldsymbol{\Sigma}_g^{(t+1)^{-1}} (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)})^T (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_g^{(t+1)}) \right) \Bigg) \\
&= -0.5 \sum_{i=1}^{N} \sum_{g=1}^{G} \Bigg(z_{ig}^{(t)} tr \left(\boldsymbol{\Sigma}_g^{(t+1)^{-1}} \right) \\
&\underbrace{(-2(\boldsymbol{\mu}_g^{(t+1)})^T (\mathbf{y}_i^{(t+1)} - \mathbf{y}_i^{(t)}) + (\mathbf{y}_i^{(t+1)^T} \mathbf{y}_i^{(t+1)} - \mathbf{y}_i^{(t)^T} \mathbf{y}_i^{(t)}))}_{\text{Say A}} \Bigg)
\end{aligned}
$$

Note that by construction, $z_{ig}^{(t)}$ is always non-negative. Also, since $\Sigma_g^{(t+1)^{-1}}$ is positive definite by construction, so its trace is always positive. Therefore, for the likelihood to increase the term A has to be negative.

Note that, for any two real numbers, $y^2 - x^2 = 2(y-x)x + (y-x)^2$

Now, let $\min_{j,g}(\mu_{jg}) = \kappa$. Substituting κ in A will make A independent of g.

Now, A can be written as

$$
\begin{aligned}
A &\leq \sum_{j=1}^{P} \left(-2\kappa (y_{ij}^{(t+1)} - y_{ij}^{(t)}) + \left({y_{ij}^{(t+1)}}^2 - {y_{ij}^{(t)}}^2 \right) \right) \\
&= \sum_{j=1}^{P} \Bigg(-2\kappa (y_{ij}^{(t+1)} - y_{ij}^{(t)}) + \\
&\left(2 \left(y_{ij}^{(t+1)} - y_{ij}^{(t)} \right) y_{ij}^{(t)} + \left(y_{ij}^{(t+1)} - y_{ij}^{(t)} \right)^2 \right) \Bigg)
\end{aligned}
$$

This is a quadratic equation in $(y_{ij}^{(t+1)} - y_{ij}^{(t)})$. If the above expression is ≤ 0, definitely A is ≤ 0. Therefore, upon setting the above expression to ≤ 0, yields

$$
|(y_{ij}^{(t+1)} - y_{ij}^{(t)})| \leq |2\kappa - 2y_{ij}^{(t)}| \tag{D.1}
$$

Now, the pseudo-loglikelihood $\mathcal{L}(\vartheta^{(t+1)}|y^{(t)}, z^{(t)})$ also increases monotonically because

Simulation		n	p	Marginals
A	1	100	100	All Gaussian
	2	100	500	
	3	250	500	
B	4	100	100	50% Gaussian, 50% F
	5	100	500	
	6	250	500	
C	7	100	100	50% Gaussian, 50% Gamma All σ-transformed
	8	100	500	
	9	250	500	

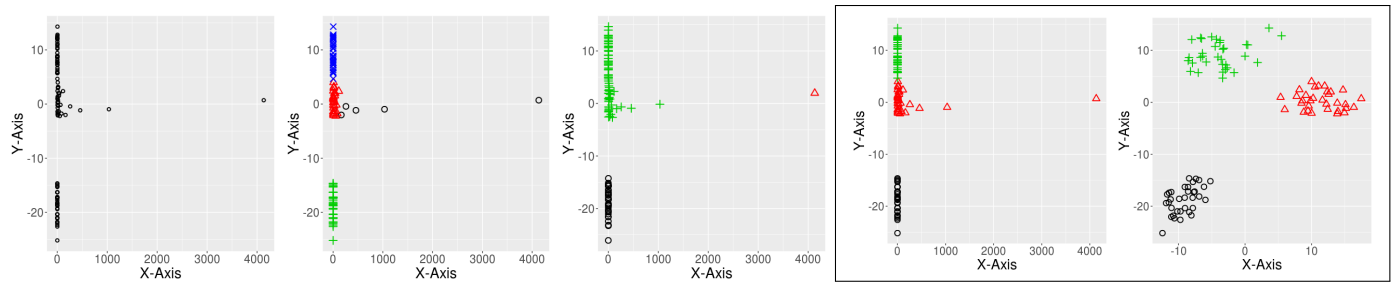
Table E.1: Simulated Datasets

$$
\begin{aligned}
\mathcal{L}(\vartheta^{(t+1)}|y^{(t+1)}) &= \int_0^1 \mathcal{L}(\vartheta^{(t+1)}|y^{(t+1)}, z^{(t)}) dz^{(t)} \\
&\geq \int_0^1 \mathcal{L}(\vartheta^{(t+1)}|y^{(t)}, z^{(t)}) dz^{(t)} \\
&= \mathcal{L}(\vartheta^{(t+1)}|y^{(t)})
\end{aligned}
$$

E Simulation Studies

Data Generation. We simulated three sets of data, A, B and C, with different choices of marginals, using a latent 10-component Gaussian mixture. In each set we vary the number of instances n and dimensions p as shown in table E.1. For each simulation setting (1 to 9), we simulated 15 different datasets. The mean and standard deviation values of the Gaussian components were randomly chosen in each dataset. In set A, i.e., simulations 1,2 and 3, we generated 10 component Gaussian mixtures using GMCM package. In set B, i.e., simulations 4,5 and 6, we initially generate 10 component Gaussian mixtures in a similar manner and then 50% of the marginals are randomly chosen to be Gaussian and the remaining are chosen to be from the F distribution with parameters 10 and 1. In simulations 7, 8 and 9, viz. set C, again we generate a 10 component Gaussian mixtures and again, 50% of the marginals are randomly chosen to be Gaussian and the remaining are chosen to be from the Gamma distribution with shape parameter 1 and scale parameter 2. However, finally, a sigmoid transformation is applied to all the dimensions. Thus set A has only Gaussian variables, set B has partially (50%) Gaussian variables and set C has non-Gaussian variables.

Results. Table E.2 shows the ARI obtained by HD-GMCM and baseline methods for each simulation, averaged over 15 datasets (with standard deviations in parentheses). With Gaussian data (set A), VarSelLCM, that assumes a Gaussian mixture model, performs the best. GMCM is second-best. At the highest p/n ratio, simulation 2, all three HD-GMCM, VarSelLCM and HDDC yield comparable results. HDDC fails for set B. The variance and skewness of F-distributed marginals are higher than Gaussian marginals. Some outlying datapoints are assigned a separate cluster that results in singularity during parameter inference, for HDDC. VarSelLCM continues to perform well, outperforming other methods while HD-GMCM is second-best. In set C, HD-



(a) Dataset (b) GMM (c) HDDC (d) HD-GMCM

Figure E.1: Clustering of non-Gaussian data with outliers. From left to right: (a) Data generated from 3 clusters, (b) Four clusters obtained by GMM, (c) 3 clusters obtained by HDDC (farthest outlier placed in a separate cluster) (d) 3 clusters obtained by HD-GMCM in data and latent space. Best viewed in color.

Set		HD-GMCM	K-means	VarSelLCM	HDDC
A	1	0.92 (0.08)	0.78 (0.12)	**1** (0.02)	0.76 (0.25)
	2	0.85 (0.08)	0.75 (0.1)	**0.99** (0.05)	0.64 (0.34)
	3	0.84 (0.09)	0.82 (0.09)	**0.99** (0.08)	0.81 (0.14)
B	4	0.57 (0.17)	0.06 (0.03)	**0.7** (0.18)	0.0 (0.0)
	5	0.59 (0.22)	0.28 (0.16)	**0.64** (0.18)	0.0 (0.0)
	6	0.53 (0.13)	0.04 (0.02)	**0.62** (0.2)	0.0 (0.0)
C	7	**0.88** (0.08)	0.79 (0.11)	0.0 (0.0)	0.74 (0.31)
	8	**0.85** (0.09)	0.74 (0.11)	0.0 (0.0)	0.73 (0.32)
	9	**0.75** (0.12)	0.74 (0.12)	0.0 (0.0)	0.85 (0.1)

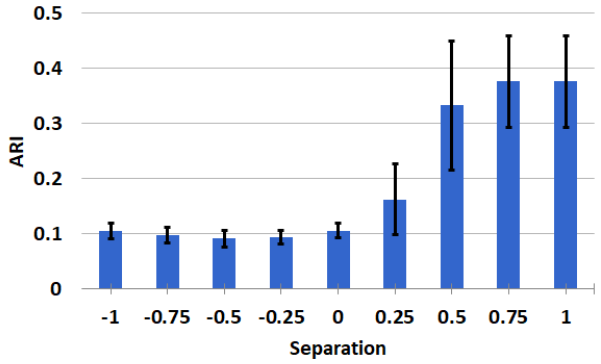
Table E.2: Average ARI and standard deviation on Simulated Datasets (Table E.1). Row-wise best results in bold.

GMCM outperforms all other methods, the difference in performance being the most at the highest p/n ratio, simulation 8. VarSelLCM fails, possibly due to outliers resulting in singularity during inference. HDDC and K-means give comparable results but remain inferior to HD-GMCM. In all 9 simulation settings, HD-GMCM outperforms K-means, GMM and GMCM. GMM and GMCM are not shown since they fail to run in all these high-dimensional datasets.

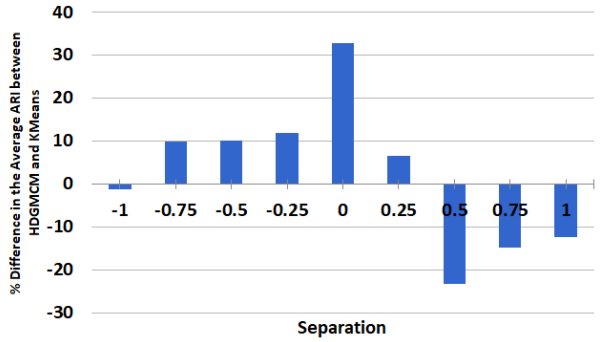
These results suggest that the performance of HD-GMCM is superior to state-of-the-art algorithms for high-dimensional non-Gaussian data, while being comparable for high-dimensional Gaussian data.

F Performance with Varying Cluster Signal

We evaluate the performance of HD-GMCM by varying the clustering signal in the data. To vary the signal, we first center the data separately for each cluster and then shift each cluster's mean value by a fixed magnitude (t). Centering the data collapses the clusters into a single group and with each shift of the mean, the clusters get more separated, thereby increasing the clustering signal in the data. We use $t = 0$ for the cluster separation observed in the data. Increasing t increases cluster separation up to $t = 1$ that indicates no overlap in the clusters. Decreasing t decreases cluster separation up to $t = -1$ that indicates complete overlap of the clusters.

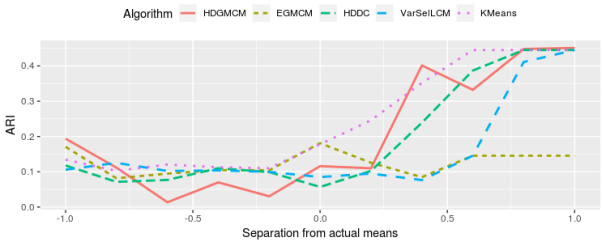


(a) Avg. ARI of K-means

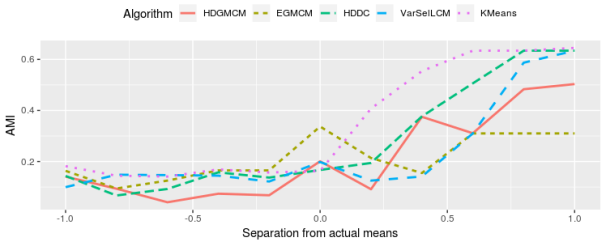


(b) % difference in Avg. ARI between HD-GMCM and K-means

Figure F.1: Performance of HD-GMCM and K-means on varying cluster separation.



(a) ARI at varying cluster separation



(b) AMI at varying cluster separation

Figure F.2: ARI (above) and AMI (below) of all baseline algorithms for a specific run.

We perform this experiment on the Khan500 dataset (James et al., 2017) using K-means and HD-GMCM for various values of t. For each t, we test with 10 random initializations of K-means and HD-GMCM, and evaluate clustering performance using ARI (averaged over 10 runs).

Figure F.1 shows the average ARI values achieved by K-means (a) and the difference between the average ARI of HD-GMCM and K-means (b). At $t = -1$, with complete overlap, there is almost no difference in performance. When the clusters are not well separated, i.e. $-0.75 \leq t \leq 0.25$, HD-GMCM outperforms K-means. When the clusters are close to being well separated $t \geq 0.5$, K-means outperforms HD-GMCM. At $t \geq 1$ as well (not shown), K-means outperforms HD-GMCM.

Figure F.2 shows the ARI and AMI over varying values of t, for a given initialization where HD-GMCM had the best LASSO-BIC at $t = 0$. We see that the performance of HD-GMCM is comparable to baselines EGMCM, HDDC and VarSelLCM. K-means outperforms all these methods when the clusters are well separated. MixGlasso is not shown since it did not run on many values of t.

G Model Selection

Empirically, we find that the LPBIC criterion works well for selecting the number of components except when the number of components is high. For instance, for Lusc-Methyl Dataset, as shown in Figure G.1, LPBIC criterion selects $g = 2$ as indicated in Table 1. However, for mixtures with large number of components, we find that both LPBIC for HD-GMCM and BIC for PGMM under-estimate the number of components as shown in figure G.2 for one of the datasets from simulation A1 ($n = 100$, $g = 10$ and $p = 100$) from E.1, We suspect this is due to the problem of isolated data points in the immensity of high-dimensional space: indeed, we find in our simulations, that some clusters have just five datapoints. LPBIC has been shown to match or outperform BIC for mixture model selection in high-dimensional settings (Bhattacharya and McNicholas, 2014). However, the local approximation of the penalty function results in shortcomings: for shrunken estimators, it stays at 0; it also fails when the posterior mode is at the boundary or outside of the initial domain of the estimates (Bhattacharya and McNicholas, 2014). More work is required to design a consistent model selection criterion using the pseudo-likelihood or a criterion based on the copula likelihood.

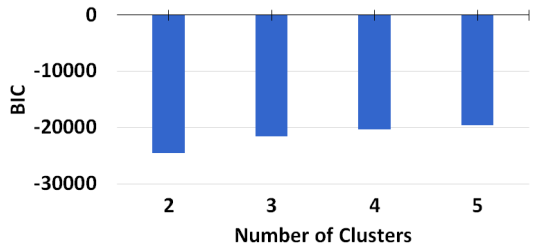
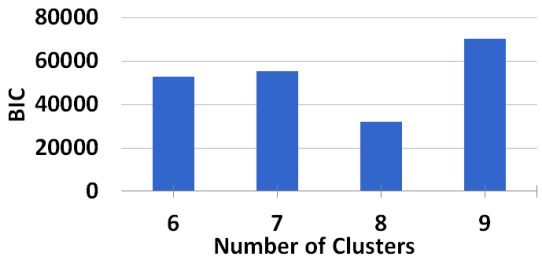
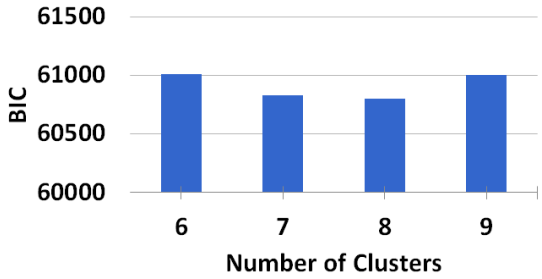


Figure G.1: LPBIC vs Number of components for Lusc Methyl Dataset



(a) HDGMCM - LPBIC



(b) PGMM - BIC

Figure G.2: BIC vs Number of clusters for simulated 10-component dataset A1 (see table E.1)

H Monotonic Increase in Likelihood

The AECM algorithm guarantees monotonic increase in likelihood if in each iteration (1) the input data is not modified for computing the updates and (2) exact updates are used for all the parameters. In the GMCM model, parameter estimates are inferred using the inverse CDF values

($y_{ij} = \Psi_j^{-1}(\hat{\mathbf{U}}_j; \boldsymbol{\vartheta})$) which changes in each iteration (thus violating condition 1 above). However, we prove in Theorem 1 that the introduction of our 'Reset y' step in HD-GMCM algorithm preserves the property of monotonic increase in pseudo-likelihood.

With respect to the second condition, exact updates are used for estimates of mean and covariance parameters, but not for the mixing proportions. This inexact update, $\hat{\pi}$, has been reported to work well for closely related LASSO-penalized models, such as the mixture of regression model (Khalili and Chen, 2007) and parsimonious Gaussian mixture model (Bhattacharya and McNicholas, 2014). Empirically we also observe monotonic increase in all the datasets we ran HD-GMCM on. For instance, in figure H.1, we show the increase in pseudo-likelihood $\mathcal{L}$ monotonically with every iteration for the simulated dataset used in Appendix G.

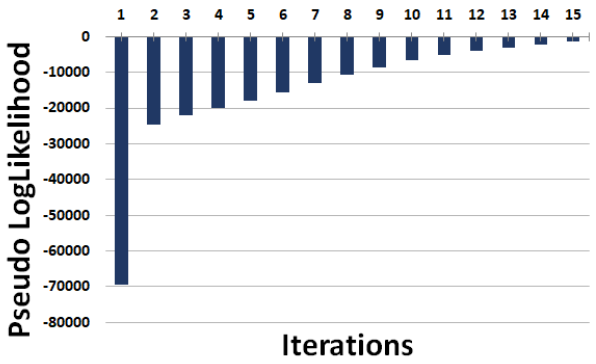


Figure H.1: Monotonic Increase in pseudo-likelihood

I Implementation

Baseline Algorithms

R packages – GMCM, VarSelLCM, HDClassif, MixGlasso, Mclust and Mixtools - were used for the baseline algorithms.

Feature Selection for Illustration

These features are genes 382 and 562 that are the variables with highest importance selected after fitting a random forest of 50 trees on the dataset (using caret package from Jed Wing et al. (2019)).

References

Sakyajit Bhattacharya and Paul D McNicholas. A lasso-penalized bic for mixture model selection. *Advances in Data Analysis and Classification*, 8(1):45–61, 2014.

Sakyajit Bhattacharya and Vaibhav Rajan. Unsupervised learning using Gaussian mixture copula model. In *21st International Conference on Computational Statistics (COMPSTAT)*. Geneva, Switzerland, 2014.

Anders E Bilgrau, Poul S Eriksen, Jakob G Rasmussen, Hans E Johnsen, Karen Dybkær, and Martin Bøgsted. GMCM: Unsupervised clustering and meta-analysis using Gaussian mixture copula models. *Journal of Statistical Software*, 70(2):1–23, 2016.

Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem, Luca Scrucca, Yuan Tang, Can Candan, and Tyler Hunt. *caret: Classification and Regression Training*, 2019. URL `https://CRAN.R-project.org/package=caret`. R package version 6.0-84.

Christian Genest, Kilani Ghoudi, and L-P Rivest. A semiparametric estimation procedure of dependence parameters in multivariate families of distributions. *Biometrika*, 82(3):543–552, 1995.

Peter D Hoff et al. Extending the rank likelihood for semiparametric copula estimation. *The Annals of Applied Statistics*, 1(1):265–283, 2007.

Gareth James, Daniela Witten, Trevor Hastie, and Rob Tibshirani. *ISLR: Data for an Introduction to Statistical Learning with Applications in R*, 2017. URL `https://CRAN.R-project.org/package=ISLR`. R package version 1.2.

Harry Joe. *Dependence Modeling with Copulas*. CRC Press, 2014.

Abbas Khalili and Jiahua Chen. Variable selection in finite mixture of regression models. *Journal of the American Statistical Association*, 102(479):1025–1038, 2007.

Ioannis Kosmidis and Dimitris Karlis. Model-based clustering using copulas with applications. *Statistics and Computing*, 26(5):1079–1099, 2016.

P. D. McNicholas and T. B. Murphy. Parsimonious Gaussian mixture models. *Statistics and Computing*, 18:285–296, 2008.

A. Sklar. Fonctions de rpartition n dimensions et leurs marges. *Publ. Inst. Statist. Univ. Paris*, 8:229–231, 1959.

Ashutosh Tewari, Michael J Giering, and Arvind Raghunathan. Parametric characterization of multimodal distributions with non-Gaussian modes. In *2011 11th IEEE International Conference on Data Mining Workshops*, pages 286–292. IEEE, 2011.

Pravin K Trivedi, David M Zimmer, et al. Copula modeling: an introduction for practitioners. *Foundations and Trends® in Econometrics*, 1(1):1–111, 2007.