How the Gaussian Mixture Model can Fail and How to Fix it

Adam Kessler, Quanzhi Bi, Xiao Han Xiong, Jordan Wang, Yanhao Miao, Wanyan Shao, Zhaohe Huang

Abstract—The Gaussian Mixture Model (GMM) can serve as both a clustering algorithm and a density estimation algorithm. This method is useful in many applications due to the natural ubiquity of the Gaussian distribution in data. However, as we explore in the paper, it is not applicable to all problems. First, we highlight the k-means clustering and GMM methods' inability to recognize clusters in the two moons dataset, where interleaving half circle clusters are not identified by the linear and quadratic boundaries of these methods respectively. The paper then details how GMM also poorly models S&P 500 returns, by failing to capture the fat tailed distribution of this financial dataset. Because GMM fails in these two examples, the paper then introduces a modified version of GMM, called the Laplacian Regularized Gaussian Mixture Model (LapGMM). The LapGMM algorithm expands upon the GMM method by introducing a regularization term to the log-likelihood function used in GMM. The regularization term uses the graph Laplacian in order to recognize the geometry of the underlying distribution. This method is then applied successfully to the two moons example.

1 Introduction

The Gaussian Mixture Model (GMM) is a probabilistic model that assumes data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters [1], resulting in an overall distribution different from a Gaussian distribution. GMM is often applied to model data that the single Gaussian model fails to model appropriately. On the one hand, GMM is sometimes applied to estimate the distribution of financial data [2], which are known to exhibit fat tails. On the other hand, one can think of mixture models as the sub-populations within an overall population. Thus this model is, by nature, widely applied to solve data clustering problems [3] [4] [5]. The goal of a clustering problem is to group a subset of data points of the full dataset into disjoint groups such that within each group the data points share some "similarities".

Careless application of GMM may introduce issues, as GMM is not useful for all data clustering problems. This paper introduces two examples where GMM may appear to work but is not actually appropriate for modeling the underlying data. The first example is two moons clustering. Sometimes similarities between data points are difficult to measure using only a Euclidean distance metric. For example, the two moon datatset from scikit-learn [6] is a dataset on an ambient submanifold. The distance between points in the same cluster can be large while points in different clusters can be relatively closer in Euclidean space. Therefore, many popular clustering algorithms like k-means and GMM fail to solve this clustering problem. The paper will show that the quadratic decision boundary of GMM largely limits its clustering capability in such examples. However, one can fix this issue by introducing an elegant submanifold regularization term based on graph Laplacian. This method is called the LapGMM model [7]. The second example we introduce is one with fat tails. The paper explains that the upper bound of tail probability of GMM is in the same order of magnitude as a Gaussian distribution, while the tail probability of financial data follows the power law. Thus, GMM again fails to accurately estimate the distribution of this financial dataset.

This document is the Group 2's term paper of MTH 9898, Barch College.

The paper is organized in the following way: Section 2 states the problem and lists the research questions the paper addresses. Section 3 contains a comprehensive review and theoretical analysis of the methodologies used, namely k-means, GMM, and LapGMM. Section 4 presents the results of the experiments on the two example datasets. Finally, Section 5 summarizes our findings.

2 Statement of the Problem

In identifying a data clustering method to model a given dataset, the appropriateness of the method must be assessed in accordance with the underlying distribution. In this paper we assess several data clustering methods on two different datasets to determine whether a given clustering method is appropriate, and when it fails, explain why it fails. Through experimentation on two distinct datasets, we address the following research questions;

RQ1 : Why does the k-means algorithm fail to detect the two clusters in the two moons example?

RQ2 : Why does the GMM algorithm fail to detect the two clusters in the two moons example?

RQ3 : Is the distribution of weekly returns of the S&P 500 index characterized by fat tails?

RQ4 : Why does the GMM algorithm fail to capture the distribution of weekly return of the S&P 500 index?

RQ5 : Why can LapGMM successfully detect the two clusters in the two moons example?

RQ6 : How do different parameters affect the performance of LapGMM?

3 Methodology Review

3.1 K-means Clustering

The k-means method attempts to divide data into clusters by minimizing the sum of squared Euclidean distances between the points of each cluster and their respective centers. One way to think about the k-means model is to regard each cluster as the center of a circle (or, in higher dimensions, a hyper-sphere), with its radius defined by the most distant point belonging to that cluster. This radius acts as a hard cutoff for cluster assignment within the training set: any point outside this circle is not considered a member of the cluster. Boundaries for out-of-sample data are then determined by lines or hyper-planes formed by the equal distance between cluster centers. Due to this property, the boundaries between k-means clusters are always linear, which indicates that it's not suitable for more complicated boundaries. This explains the answer to **RQ1**.

3.2 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) is a probabilistic model that seeks to fit all data points on a mixture of a finite number of Gaussian distributions with unknown parameters. It is classified as a soft clustering algorithm, in contrast with the k-means algorithm where the membership of a data point to a cluster is binary in nature. GMM assigns each data point to all clusters with a given probability.

The GMM is initialized by employing initial guesses of model parameters, including the number of Gaussian distributions, the parameters of those distributions, and a vector of probabilities indicating the likelihood of belonging to each cluster. The fitting of GMM is an iterative process composed of two main steps. The first is to calculate the probability of each data point's membership to each cluster, and the second, to find optimized parameters for each Gaussian distribution by maximizing the log-likelihood function over all data points. The process iterates until convergence.

The GMM method is slightly more complex than k-means in that it identifies clusters that fit a conditional multivariate Gaussian distribution with parameters unique to each cluster. Unlike the Voronoi cells identified by the k-means method, the GMM method imposes ellipses or ellipsoids (often overlapping) driven by each cluster's conditional distribution because the contour lines of conditional Gaussian likelihood are ellipses in 2-D and ellipsoids in 3-D space. One of the main differences between these two algorithms is that the k-means model has no intrinsic measure of probability or uncertainty of cluster assignments. In other words, k-means uses a "hard assignment" while GMM employs a "soft assignment": GMM contains a probabilistic model under the hood to determine the likelihood of cluster assignment.

To address **RQ2**, it is important to notice that the decision boundary of the GMM model for clustering is in general a quadratic one. Suppose the parameters of GMM are

$$\Theta = [\alpha_1, ..., \alpha_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K] \tag{1}$$

where α_i is the prior probability of belonging to cluster i, and μ_i and Σ_i are the parameters of i-th conditional Gaussian distribution. Then, the ratios of posteriors become

$$\begin{aligned}
\log \frac{\mathbb{P}(i|\mathbf{x})}{\mathbb{P}(j|\mathbf{x})} &= \log \frac{\alpha_i \mathbb{P}(\mathbf{x}|i)}{\alpha_j \mathbb{P}(\mathbf{x}|j)} \\
&= \log \frac{\alpha_i \frac{1}{|2\pi\Sigma_i|} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)\right]}{\alpha_j \frac{1}{|2\pi\Sigma_j|} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu_j)^T \Sigma_j^{-1}(\mathbf{x}-\mu_j)\right]} \\
&= \frac{1}{2}\mathbf{x}^T(\Sigma_j^{-1} - \Sigma_i^{-1})\mathbf{x} + (\Sigma_i^{-1}\mu_i - \Sigma_j^{-1}\mu_j)^T \mathbf{x} \\
&+ \frac{1}{2}\left(\mu_j^T \Sigma_j^{-1}\mu_j - \mu_i^T \Sigma_j^{-1}\mu_i\right) + \log \frac{\alpha_j \sqrt{|2\pi\Sigma_i|}}{\alpha_j \sqrt{|2\pi\Sigma_i|}} \\
&= \mathbf{x}^T \mathbf{W} \mathbf{x} + \omega^T \mathbf{x} + c
\end{aligned} \tag{2}$$

resulting in quadratic decision boundaries which would limit the capability of the GMM for certain underlying distributions. Examples such as the two moons dataset need a decision boundary with at least 2 inflection points while quadratic boundary can provide only one.

To address **RQ4**, we need to quantify the tail distribution of GMM. The bound of the GMM tail distribution is very similar to the Gaussian distribution. The tail of the Gaussian distribution can be bounded by the following inequality [8]:

$$\frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}}\left(\frac{1}{t} - \frac{1}{t^3}\right) \le \mathbb{P}(X > \mu + \sigma t) \le \frac{e^{\frac{-t^2}{2}}}{t\sqrt{2\pi}} \tag{3}$$

where $t > 0$.

For the GMM distribution:

$$\begin{aligned}
p(x) &= \sum_{k=1}^{K} \alpha_k \mathcal{N}(x|\mu_k, \sigma_k) \\
&= \sum_{k=1}^{K} \alpha_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right) \\
&= \sum_{k=1}^{K} \alpha_k p_k(x)
\end{aligned} \tag{4}$$

which naturally leads to the tail distribution:

$$\mathbb{P}(X > t) = \sum_{k=1}^{K} \alpha_k \mathbb{P}_k(x > t) \tag{5}$$

The tail distribution $\mathbb{P}_k(x > \mu_k + \sigma_k t)$ can be bounded by inequality (3). Thus, the tail distribution of $p(x)$, $\mathbb{P}(X > t)$ is within the region restricted by the following boundary:

$$\begin{aligned}
&\sum_{k=1}^{K} \frac{\alpha_k \sigma_k}{\sqrt{2\pi}} \left(\frac{1}{(t-\mu_k)} - \frac{1}{(t-\mu_k)^3}\right) \exp \frac{-(t-\mu_k)^2}{2\sigma_k^2} \\
\le &\mathbb{P}(X > t) \\
\le &\sum_{k=1}^{K} \frac{\alpha_k \sigma_k}{\sqrt{2\pi}(t-\mu_k)} \exp \frac{-(t-\mu_k)^2}{2\sigma_k^2}
\end{aligned} \tag{6}$$

as we can see if $t \to \infty$, we have

$$\mathbb{P}(X > t) = \Omega\left(\frac{\alpha_{k\max}\sigma_{k\max}}{\sqrt{2\pi}(t-\mu_{k\max})} \exp \frac{-(t-\mu_{k\max})^2}{2\sigma_{k\max}}\right) \tag{7}$$

so the asymptotic tail probability of GMM has the same order of magnitude as the Gaussian distribution.

```
Input  : dataset $\mathbf{X} \in \mathbb{R}^{m\times d}$, number of clusters $K$, number of nearest neighbors $p$, regularization parameter $\lambda$,
         termination value $\delta_{\mathrm{EM}}$, termination value $\delta_{\mathrm{NR}}$, maximum iteration $N_{\mathrm{EM}}$, maximum iteration $N_{\mathrm{NR}}$,
         maximum iteration $N_{\mathrm{M}}$
Output: the clustering result $\mathbf{y} \in [K]^m$
Construct the nearest neighbor matrix $S$
Calculate graph Laplacian $L = D - S$
Get K-means clustering $y_{\mathrm{init}} \in [K]^m$
Initialize $\alpha_k^0 = \frac{1}{m}\sum_{i=1}^m \mathbb{I}(y_{\mathrm{init},i} = k)$
Initialize $\mu_k^0 = \frac{1}{m}\sum_{i=1}^m \mathbf{x}_i\mathbb{I}(y_{\mathrm{init},i} = k)$
Initialize $\Sigma_k^0 = \frac{1}{m}\sum_{i=1}^m (\mathbf{x}_i - \mu_i)(\mathbf{x}_i - \mu_i)^T\mathbb{I}(y_{\mathrm{init},i} = k)$
for $n \leftarrow 1$ to $N_{\mathrm{EM}}$ do
    E-step: compute the posterior probabilities
        $$\mathbb{P}(k|\mathbf{x}_i) = \frac{\alpha_k^{n-1}\mathcal{N}(\mathbf{x}_i|\mu_k^{n-1}, \Sigma_k^{n-1})}{\sum_{j=1}^K \alpha_j^{n-1}\mathcal{N}(\mathbf{x}_i|\mu_j^{n-1}, \Sigma_j^{n-1})} \quad (8)$$
    M-step:
    $\gamma = 0.9$
    for $i \leftarrow 1$ to $N_{\mathrm{M}}$ do
        Smooth the posterior probabilities using Newton-Raphson method
        for $j \leftarrow 1$ to $N_{\mathrm{NR}}$ do
            $$\mathbb{P}(k|\mathbf{x}_i) = (1-\gamma)\mathbb{P}(k|\mathbf{x}_i) + \gamma\frac{\sum_{j=1}^m S_{ij}\mathbb{P}(k|\mathbf{x}_j)}{\sum_{j=1}^m S_{ij}} \quad (9)$$
            Evaluate the manifold regularization $R^j = \sum_{k=1}^K \mathbf{f}_k^T L\mathbf{f}$
            if $|R^j - R^{j-1}| < \delta_{NR}$ then
                break
            end
        end
        Compute the LapGMM estimates $\alpha^n, \mu^n, \sigma^n$
            $$\alpha_i^n = \frac{1}{m}\mathbb{P}(i|\mathbf{x}_j), \quad \mu_i^n = \frac{\sum_{j=1}^m \mathbf{x}_j\mathbb{P}(i|\mathbf{x}_j)}{\sum_{j=1}^m \mathbb{P}(i|\mathbf{x}_j)}, \quad \Sigma_i^n \frac{\sum_{j=1}^m \mathbb{P}(i|\mathbf{x}_j)(\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T}{\sum_{j=1}^m \mathbb{P}(i|\mathbf{x}_j)} \quad (10)$$
        Evaluate the regularized log likelihood
            $$\mathcal{L}(\Theta^n) = \sum_{i=1}^m \log\left(\sum_{j=1}^m \alpha_j\mathcal{N}(\mathbf{x}_j|\mu_j^n, \Sigma_j^n)\right) - \lambda\sum_{i=1}^K \mathcal{R}_k \quad (11)$$
        if $\mathcal{L}(\Theta^n) < \mathcal{L}(\Theta^{n-1})$ then
            $\gamma = 0.9\gamma$
        else
            break
        end
    end
    if $\mathcal{L}(\Theta^n) - \mathcal{L}(\Theta^{n-1}) \leq \delta_{\mathrm{M}}$ then
        break
    end
end
$y_i = \arg\max_k \mathbb{P}(k|\mathbf{x}_i)$
return y
```

Algorithm 1: The LapGMM Algorithm

3.3 Laplacian Regularized Gaussian Mixture Model

"Laplacian Regularized Gaussian Mixture Model for Data Clustering" introduces an adjustment to the standard GMM method that allows for the recognition of the underlying geometry of probability distributions from which data is sampled, when such geometry exists on a submanifold of a high dimensional ambient space. This method leverages the power of both the mixture model and the graph Laplacian regularization [7].

The aim is to fit a probabilistic model while respecting the manifold structure. Specifically, if two data points are sufficiently close on the manifold, then they are likely to be generated by sampling the same Gaussian component. In the real world when the data manifold is unknown, a nearest neighbor graph is constructed to discretely model the manifold structure. Using graph Laplacian, the manifold structure can be incorporated in the log-likelihood function of the standard GMM as a regularization term. This gives the clustering algorithm the ability to recognize a geodesic, which is a generalization of the idea of the "straight line" to "curved spaces".

After that, a Generalized EM algorithm is illustrated to fit the model. The paper [7] performs experiments on several datasets to show the effectiveness of LapGMM, including our two moons example, as well as USPS, COIL20, and TDT2 datasets. The LapGMM successfully identifies the two moons which both k-means and standard GMM fail to, because this method encodes more discriminating information compared to standard GMM by smoothing the conditional probability density functions. On all three empirical datasets, LapGMM outperforms the other four algorithms tested.

When adopting the Generalized EM algorithm, there are two essential parameters: the number of nearest neighbors p which defines the "locality" of the data manifold, and the regularization parameter λ that controls the smoothness of the conditional probability distributions. The paper [7] demonstrates that selection of parameters is not a very crucial problem in the LapGMM algorithm.

Algorithm 1 shows our implementation of the LapGMM method. We made some additional assumptions that the original paper didn't mention. First, we use the same initial $\gamma = 0.9$ every time we move from the E-step to the M-step. Otherwise γ will decrease to 0 exponentially, making the smoothing process hard to converge in the late stages of the EM iteration. Second, we choose to setup a maximum iteration number for all the while loops. Unfortunately, there's no mathematical proof that decreasing γ will result in a larger $\mathcal{L}(\Theta)$ so we introduce an early stopping mechanism to avoid endless loops.

4 EXPERIMENT

4.1 Experiment Setup

4.1.1 Implementation of GMM

The initial values of the GMM are generated using uniform distribution sampling, random sampling, and k-means. In uniform distribution sampling, mean and variance vectors are obtained from K equally sized clusters after shuffling the observations. The prior probability of each cluster is identical because of the same observations' membership to each cluster. In random sampling, the mean vector is generated by K randomly selected observations. Each observation is assigned to the nearest cluster with one mean value as the center. The prior probability and variance are calculated cluster by cluster based on the sample variance and the number of elements. In the k-means setting, k-means with given number of clusters is conducted before the calculation of three parameter vectors based on the elements in each cluster.

After initialization, EM process is performed in estimation of the parameters. The process iterates until either the log-likelihood or the number of iterations reaches the maximum. Note that BIC is used in selection of the optimal number of clusters.

4.1.2 Fitting GMM on S&P 500 Index Data

S&P 500 index data was downloaded from Yahoo Finance. We used adjusted price from 2000-01-01 to 2021-03-14. Weekly return was calculated as log return.

For the GMM model, we used different initialization methods and number of components to ensure the robustness of the result. Initialization methods included uniform distribution initialization, random initialization and K-means initialization. The parameters of the algorithm were listed below.

Symbol	Description	Value
$(m.d)$	Size of the dataset	1105×1
K	Number of clusters	$1 \sim 15$
δ_{EM}	termination value of EM iteration	10^{-2}
N_{EM}	max iteration of EM	100

TABLE 1: Parameters of GMM Algorithm

4.1.3 Implementation of LapGMM

We implemented LapGMM based on Algorithm 1 with the following parameters

Symbol	Description	Value
$(m.d)$	Size of the dataset	200×2
K	Number of clusters	2
p	Number of nearest neighbors	5
λ	Regularization parameter	100
δ_{EM}	termination value of EM iteration	10^{-6}
δ_{NR}	termination value of Newton method	10^{-6}
N_{EM}	max iteration of EM	50
N_{NR}	max iteration of Newton method	2000
N_{M}	max iteration of M-step	500

TABLE 2: Parameters of LapGMM Algorithm

4.2 Results and Analysis

4.2.1 K-means and GMM for Two Moons Example

For the two moons example shown in Figure 1a, we can see in Figure 1b to 1e that k-means, NMF and Gaussian Mixture Model (GMM) fail to identify the moon clusters. The linear decision boundary of k-means and quadratic decision boundary of GMM limited the clustering ability in two-moon dataset. Although NMF doesn't have a clear decision boundary, it models each cluster as a linear combination of the data points. So it's difficult to cluster the ambient space.

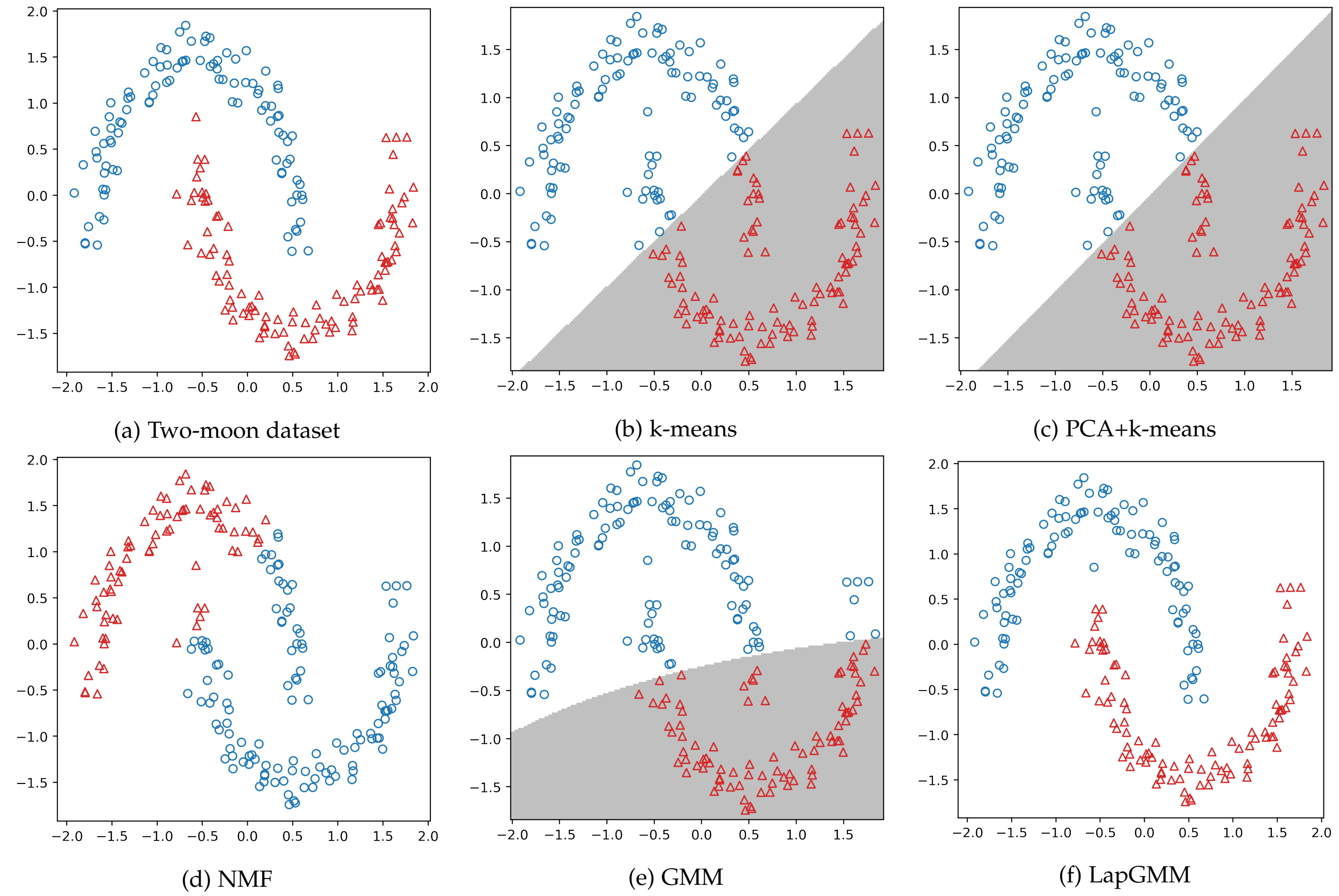

(a) Two-moon dataset
(b) k-means
(c) PCA+k-means
(d) NMF
(e) GMM
(f) LapGMM

Fig. 1: Clustering on the two moons pattern. (a) Original data set; (b) Clustering results given by k-means with decision boundary; (c) Clustering results given by PCA and k-means with decision boundary; (d) Clustering results given by NMF; (e) Clustering results given by GMM with decision boundary; (f) Clustering results given by LapGMM

Both k-means and GMM identify clusters based on "round" data in space; one using "balls" and the other using "ellipses". Although GMM makes some assumptions on the underlying distribution of input data, in the two moons example it still fails to consider the submanifolds on which the underlying moon distributions are sampled from. It is clear from a quick examination of figure 1a that the two moons example has an intrinsic "submanifold" of the ambient space. Neither K-means nor GMM incorporates this geometric structure, giving us an answer to **RQ1** and **RQ2**. In order to come up with a method that could work for the two moons example, the underlying geometry of the probability distributions needs to be considered.

4.2.2 GMM for S&P 500 index density estimation

GMM can be used not only as a clustering algorithm but also as a density estimation algorithm. Examining the density of weekly returns of the S&P 500 index, it has a heavier tail than normal distribution. The fat tail feature of index return is also referred to as tail risk, which is attributed to the occurrence of extreme events and abnormal returns. We fit a GMM model to estimate the distribution of weekly returns of the S&P 500 index, and investigated whether this model is able to reproduce correctly the tails of the distribution.

The tail feature of the obtained distribution was examined using three different approaches. In Figure 2, log plot shows the logarithmic scale of tail probability along the weekly return. When the weekly return is small and near the center of the distribution, the GMM distribution fits the true data much better than normal distribution. However, when the weekly return is large and at the tail of the distribution, it approaches normal distribution in an asymptotic manner. To verify GMM's failure in reproducing the tails of S&P, we quantitatively deduced the downward trend of S&P original data by fitting a linear model with no intercept. The R^2 of the fitted model is over 98%, indicating the S&P original data declines at a rate of $\exp(-at), a > 0$, which is slower than the declining rate of GMM and normal distribution. Both of these methods only capture a part of the tail risk and thus fail to reproduce correctly the tails of the true data.

Q-Q plot is another way to check the tail pattern. The Q-Q plot of S&P 500 index return (Figure 3) shows a downward deviation at the left and an upward deviation at the right, which conforms to the fat tail pattern (**RQ3**). As is shown in Figure 4, if we only used one component in the GMM model, the fitted distribution is also normal. It's trivial that they don't have fat tail as is shown in the first column. When we used more than one components, the Q-Q plots all show similar fat tail pattern no matter what initialization

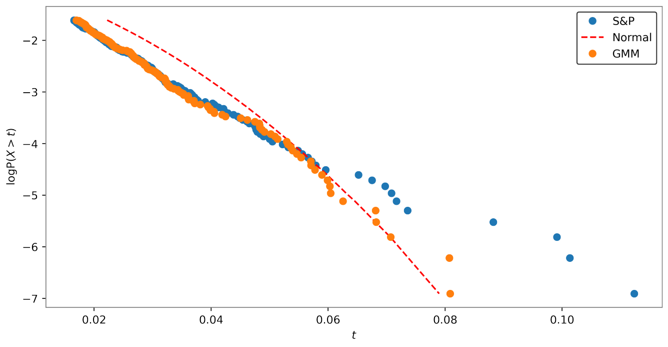


Fig. 2: Log plot of the tail. When the weekly return is small and near the center of the distribution, the GMM distribution fits the true data much better than normal distribution. When the weekly return is large and at the tail of the distribution, it approaches normal distribution in an asymptotic manner.

methods we used. The result indicates that the distribution fitted by GMM does capture at least a part of the fat tail characteristics.

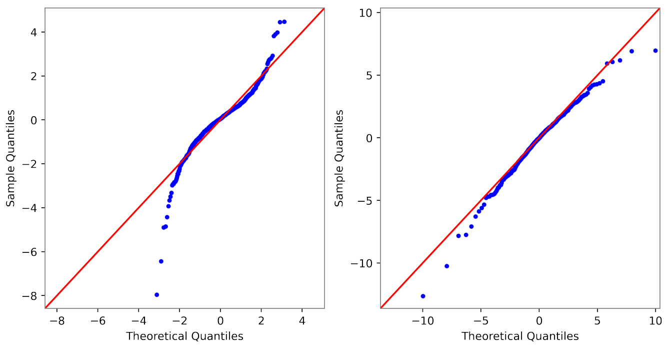


Fig. 3: Q-Q plots of S&P 500 index weekly return. The parametric curve is generated by different distributions (left: normal distribution, right: t-distribution with degree of freedom equals to 3). From the figure, we can conclude the fat tail characteristics of the S&P 500 index weekly return.

We also compared the histogram and the density functions fitted by kernel density estimation and GMM respectively in Figure 5. In the zoomed view of the tail, we can see that GMM can't fit the density well. The tail of the fitted GMM density is thinner than that of the kernel density.

All of this evidence shows that GMM can capture a part of tail risk near the tail part but fail to reproduce correctly the fat tail of S&P index returns. (**RQ4**)

4.2.3 LapGMM for Two Moons Example

From Figure 1f, we can see that LapGMM successfully identifies the moon clusters, which proves its ability to have complex decision boundaries, in contrast to linear decision boundaries of K-means, and quadratic decision boundaries of standard GMM. This is mainly contributed by the regularization term $\sum_k \mathcal{R}_k$, which can also answer **RQ5**. To further see how LapGMM improves its estimation throughout iterations, we plotted the evolution of objective

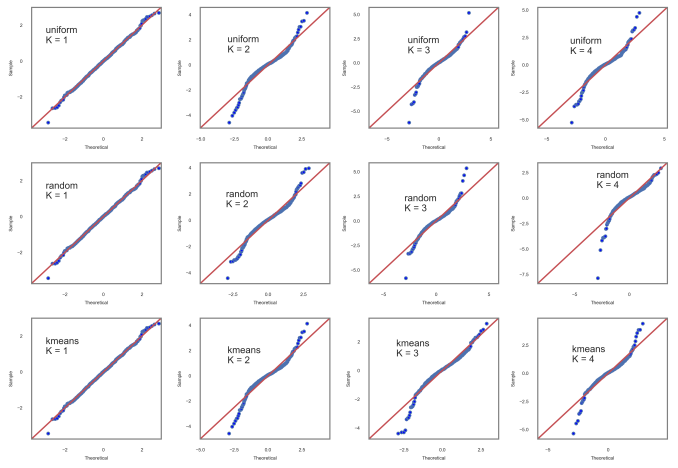


Fig. 4: Q-Q plots of GMM fitted S&P with different initialization methods (Uniform Distribution Sampling, Random Sampling, K-means) and number of components (1∼4). (horizontal: sample quantile, vertical: theoretical quantile). If we only used one component in the GMM model, the fitted distribution is also normal. When we used more than one components, the Q-Q plots all show similar fat tail pattern no matter what initialization methods we used.

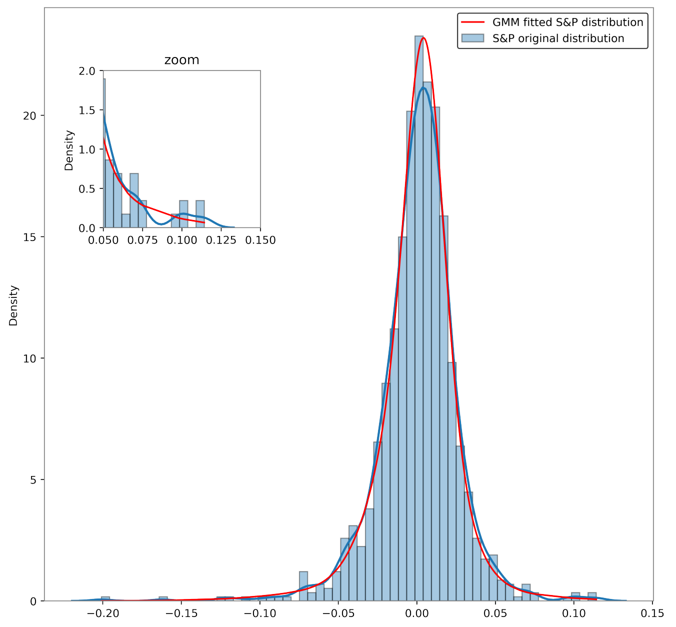
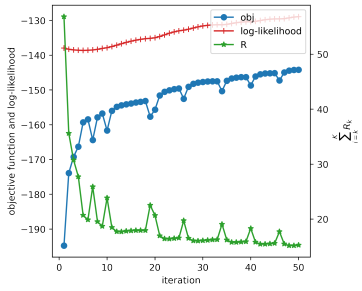
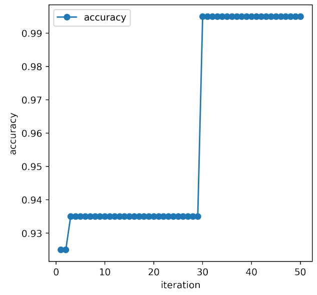


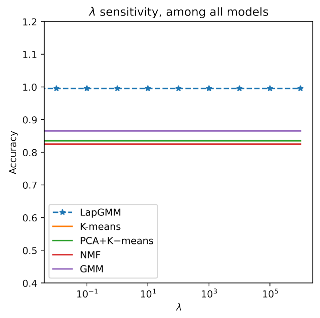
Fig. 5: GMM fitted results and S&P histogram with zoomed view on the tail distribution. In the zoomed view of the tail, we can see that the tail of the fitted GMM density is thinner than that of the kernel density.

function $\mathcal{L}(\Theta)$, log-likelihood function $\mathcal{Q}(\Theta, \Theta^{n-1})$, regularization term $\sum_k \mathcal{R}_k$ and accuracy, as shown in Figure 6a and 6b.

We could see that $\mathcal{Q}(\Theta, \Theta^{n-1})$ continuously increases, while $\sum_k \mathcal{R}_k$ does not. In each iteration, LapGMM maximizes log likelihood and minimizes manifold regularization respectively using Newton's method, to find a "locally better" fit for the parameters. Therefore, if stopping criteria of Newton's method are too strict in M-step, it will more likely


(a) Objective evolution


(b) Accuracy evolution


(c) Sensitivity w.r.t. λ

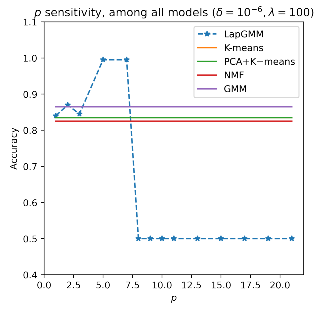

(d) Sensitivity w.r.t. p

Fig. 6: (a) – (b) Evolution of objective function $\mathcal{L}(\Theta)$, log-likelihood function $\mathcal{Q}(\Theta, \Theta^{n-1})$, regularization term $\sum_k \mathcal{R}_k$ and accuracy, throughout all iterations. As we can see once the regularization term is small enough, the accuracy will jump to a high level. (c)– (d) Performance sensitivity of LapGMM, K-means, PCA + K-means, NMF and standard GMM, w.r.t changes in λ and p.

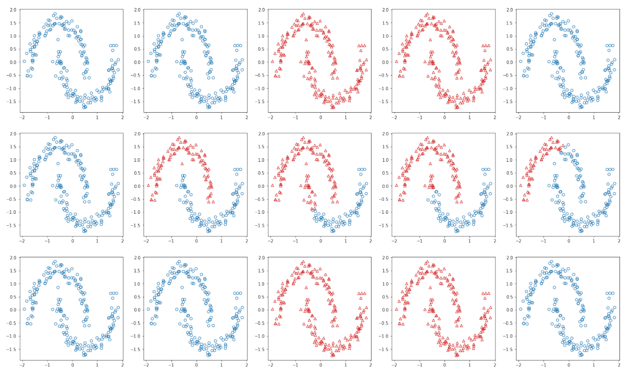


Fig. 7: Performance robustness of LapGMM, w.r.t S and p; row 1∼3 for $S \in \{'0-1', '\text{dot}-\text{product}', '\text{heat}'\}$; column 1∼5 for $p \in \{8, 7, 6, 5, 4\}$. As we can see if we change the weight matrix S, the performance of LapGMM will change by a lot.

find a globally optimal Θ, causing a jump in $|f_k(\mathbf{x}_i) - f_k(\mathbf{x}_j)|$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close", as well as a jump in $\sum_k \mathcal{R}_k$.

While tuning the definition of weight matrix S, the regularization coefficient λ, the number of nearest neighbors p, and the maximum number of iterations N, LapGMM's performance varies a lot. See Figure 7 for an example of LapGMM's performance robustness, w.r.t. different S and p respectively. Notice that sometimes the LapGMM clusters all data points into a single cluster. This is a typical overfitting phenomenon with respect to $\sum_k \mathcal{R}_k$. We can easily find that $\mathbb{P}(k|\mathbf{x}) = 1/K$ will make the regularization 0 which happens to be the global minimum. Thus if we over smooth the posterior in the Newton method step, we will get a very small regularization term. That corresponds to a posterior probability of 0.5 for all the data points. Thus we need to control the $\delta_{\text{NR}}, N_{\text{NR}}$ to avoid overfitting.

LapGMM outperforms other clustering algorithms, with "good" hyper-parameters. From Figure 6c and 6d, we can also see that LapGMM generally outperforms other clustering algorithms, including K-means, PCA + K-means, NMF and standard GMM. In addition, the performance of LapGMM is not affected by changes in λ, but largely affected by changes in p. We generated the two-moon dataset with $m = 200$. A rule of thumb is that $p = \sqrt{m} \approx 14$; while in practice, we found that $p = \lceil \sqrt{m}/3 \rceil = 5$ usually worked best. If p is too small, the weight matrix S becomes too sparse, then we'll update $\mathbb{P}(k|\mathbf{x}_i)$ too efficiently to quickly find the global minimum of $\sum_k \mathcal{R}_k$. The analysis above answers **RQ6**.

5 Conclusion

Both the k-means and GMM algorithms fail to detect the two clusters in the two moons example. This is because the two moons example has an intrinsic "submanifold" of the ambient space, but neither k-means nor GMM incorporates this geometric structure, mainly due to their boundary properties.

For the S&P 500 Index weekly return data, GMM can capture a part of tail risk near the tail but fails to correctly reproduce the fat tail. The logarithmic scale of tail probability, Q-Q plot, and kernel density estimation are employed to provide evidence. The reason is explained by quantifying the tail distribution of GMM: the asymptotic tail probability of GMM has the same order of magnitude as the Gaussian distribution.

Laplacian Regularized Gaussian Mixture model (LapGMM) boasts its enhanced ability to have complex decision boundaries and successfully detect the two clusters in the two moons example. Thanks to the innovative regularization term, LapGMM find a "locally-better" fit for the parameters at each iteration, which largely improves the estimation.

Moreover, tuning hyper-parameters are crucial for almost every clustering algorithms. We conclude that the performance of LapGMM is not affected by changes in λ, but largely affected by changes in p, with a rule of thumb proposed. In addition, the $\delta_{\text{NR}}, N_{\text{NR}}$ need to be properly controlled to avoid overfitting.

Thus, all the research questions have been carefully analyzed and answered. Although LapGMM is a significant improvement over standard GMM, it's far from perfect. There is no reason to believe that the nearest neighbor graph is the only or the most natural choice, and graph Laplacian is also not the only choice of smoothing operators. It still remains unclear how to do model selection in a principled manner.

References

[1] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[2] I. D. Dinov, "Expectation maximization and mixture modeling tutorial," 2008.

[3] X. Hou, T. Zhang, G. Xiong, Z. Lu, and K. Xie, "A novel steganalysis framework of heterogeneous images based on gmm clustering," *Signal Processing: Image Communication*, vol. 29, no. 3, pp. 385–399, 2014.

[4] Y. Lu, Z. Tian, P. Peng, J. Niu, W. Li, and H. Zhang, "Gmm clustering for heating load patterns in-depth identification and prediction model accuracy improvement of district heating system," *Energy and Buildings*, vol. 190, pp. 49–60, 2019.

[5] T. Su and J. G. Dy, "In search of deterministic methods for initializing k-means and gaussian mixture clustering," *Intelligent Data Analysis*, vol. 11, no. 4, pp. 319–338, 2007.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[7] X. He, D. Cai, Y. Shao, H. Bao, and J. Han, "Laplacian regularized gaussian mixture model for data clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1406–1418, 2010.

[8] W. Feller, *An introduction to probability theory and its applications, vol 2*. John Wiley & Sons, 2008.