Two Applications of EM algorithm and its expansion

Haochen Jiang

Abstract

This article mainly analyzes the effectiveness of EM algorithm in handling log-likelihood functions of some classic models like Hidden Markov Model and Gaussian Mixture Model, which are very difficult to optimize when using other techniques. Besides, another methodology which sets up constrains on tail behavior will be introduced to fit the parameters in Gaussian Mixture model, where CVaR-distance will be mentioned. Finally, evaluations about these techniques will be carried out.
In the part of empirical study, 1min and 30min frequency returns of Apple stock in the dynamic time period of 2020 will be used to fit these three types of model respectively and then the results and conclusion will be shown.

1 Introduction

In statistics, an expectation–maximization (EM) algorithm is an iterative method to find (local) maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. The EM iteration alternates between performing an expectation (E) step, which creates a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step.[10]

2 EM Algorithm[2]

This part is mainly translated from the textbook of Li (2019).

As a convention, we use Y to represent our observations, while Z is the latent or missing data, so that (Y;Z) is called complete-data. And observations are also called incomplete-data.
Assume we are given the observations Y with probabilistic distribution $P(Y|\theta)$, where θ is the model parameters to be estimated. Then the likelihood function of incomplete data Y is $P(Y|\theta)$, whose log-likelihood function $LL(\theta) = \log P(Y|\theta)$; And assume the joint distribution of (Y,Z) is $P(Y, Z|\theta)$, then the complete data log-likelihood function is $CDLL(\theta) = \log P(Y, Z|\theta)$.
EM algorithm uses iterations to estimate the maximum likelihood estimation of $LL(\theta) = logP(Y|\theta)$.

Algorithm 2.1 (EM Algorithm)**.**
Input: observations Y, latent data Z, joint distribution $P(Y, Z|\theta)$, conditional distribution $P(Z|Y, \theta)$;
Output: model parameters θ.

1. *Start with initial guesses for the parameters $\theta^{(0)}$;*
2. *Expectation Step: In (i+1)th step, compute:*

$$\begin{aligned} Q\left(\theta, \theta^{(i)}\right) &= E_Z\left[\log P(Y, Z|\theta)|Y, \theta^{(i)}\right] \\ &= \sum_Z \log P(Y, Z|\theta) \cdot P(Z|Y, \theta^{(i)}) \end{aligned} \tag{1}$$

where $\theta^{(i)}$ is the estimate of the ith step, and $P(Z|Y,\theta^{(i)})$ is the conditional distribution of latent data Z given observations Y and current estimate of parameters $\theta^{(i)}$.

3. Maximization Step: determine the maximizer of $Q\left(\theta,\theta^{(i)}\right)$ over θ as the estimate of (i+1)th step of model parameters $\theta^{(i+1)}$.

$$\theta^{(i+1)} = \arg\max_{\theta} Q\left(\theta,\theta^{(i)}\right) \tag{2}$$

4. Iterate steps 2 and 3 until convergence

Function $Q\left(\theta,\theta^{(i)}\right)$ is the core concept in the EM algorithm, which is called Q function.

Definition 2.1 (Q Function)**.** *The expectation of complete-data log-likelihood $CCDL(\theta) = \log P(Y,Z|\theta)$ given the conditional probabilistic distribution $P(Z|Y,\theta^{(i)})$ of unobserved data Z conditioning on the observations Y and current estimate of parameters $\theta^{(i)}$ is called **Q Function**, i.e.,*

$$Q\left(\theta,\theta^{(i)}\right) = E_Z\left[\log P(Y,Z|\theta)|Y,\theta^{(i)}\right] \tag{3}$$

Here are some remarks for EM algorithm:

1. The initial values can be randomly chose, but we have to notice that EM algorithm is sensitive to initial values, which means that it is likely to reach the local maxima finally.
2. In the maximization in M-step, when $\theta^{(i+1)}$ is estimated, we finish one step. It will be illustrated that log-likelihood function will increase or reach local maxima after each step.
3. As for the condition that we stop iterating, we commonly use

$$||\theta^{(i+1)} - \theta^{(i)}|| < \epsilon_1 \quad or \quad ||Q\left(\theta^{(i+1)},\theta^{(i)}\right) - Q\left(\theta^{(i)},\theta^{(i)}\right)|| < \epsilon_2$$

where ϵ_1 and ϵ_2 are very small positive numbers. When the inequality is satisfied, we will stop iterating.

Next, I will illustrate why EM algorithm makes sense in terms of maximizing the log-likelihood function of the observations, i.e., $LL(\theta) = \log P(Y|\theta)$.

Our goal is to maximize log-likelihood function of observations (incomplete-data) conditioning on parameter θ, i.e.,

$$\begin{aligned} LL(\theta) = \log P(Y|\theta) &= \log \sum_Z P(Y,Z|\theta) \\ &= \log\left(\sum_Z P(Y|Z,\theta)\cdot P(Z|\theta)\right) \end{aligned}$$

We can notice that the main difficulty to maximize this term is that it includes the logarithm of sum (or integral). In fact, EM algorithm takes steps to approximately maximize $LL(\theta)$.
Assume the estimate of θ after ith step is $\theta^{(i)}$. We hope that new estimate θ could yield larger values in $LL(\theta)$, i.e., $LL(\theta) > LL(\theta^{(i)})$, and could reach local maxima eventually.
Consider two difference of these two terms:

$$\begin{aligned} LL(\theta) - LL(\theta^{(i)}) &= \log\left(\sum_Z P(Z|Y,\theta^{(i)})\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Z|Y,\theta^{(i)})}\right) - \log P(Y|\theta^{(i)}) \\ &\geq \sum_Z P(Z|Y,\theta^{(i)})\log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Z|Y,\theta^{(i)})} - \log P(Y|\theta^{(i)}) \\ &= \sum_Z P(Z|Y,\theta^{(i)})\log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Z|Y,\theta^{(i)})} - \sum_Z P(Z|Y,\theta^{(i)})\log P(Y|\theta^{(i)}) \\ &= \sum_Z P(Z|Y,\theta^{(i)})\log\frac{P(Y|Z,\theta)P(Z|\theta)}{P(Z|Y,\theta^{(i)})P(Y|\theta^{(i)})} \end{aligned} \tag{4}$$

Notice that we use a special case of Jensen inequality in the derivation, which is

$$\log \sum_j \lambda_j y_j \geq \sum_j \lambda_j \log y_j \ where\, \lambda_j \geq 0 \ and \sum_j \lambda_j = 1 \tag{5}$$

Then the lower bound of $LL(\theta)$ can be got, which is denoted as $B(\theta, \theta^{(i)})$, where

$$B(\theta, \theta^{(i)}) := LL(\theta^{(i)}) + \sum_Z P(Z|Y, \theta^{(i)}) \log \frac{P(Y|Z, \theta)P(Z|\theta)}{P(Z|Y, \theta^{(i)})P(Y|\theta^{(i)})} \leq LL(\theta) \tag{6}$$

And we can also know that $LL(\theta^{(i)}) = B(\theta^{(i)}, \theta^{(i)})$ from (4).
Therefore, any value of θ which can make $B(\theta, \theta^{(i)})$ larger could make $LL(\theta)$ larger too. So we can maximize $B(\theta, \theta^{(i)})$ to reach the same goal as maximizing $LL(\theta)$, and choose the value $\theta^{(i+1)}$ to make $B(\theta, \theta^{(i)})$ reach the maxima, i.e.,

$$\theta^{(i+1)} = \arg\max_\theta B(\theta, \theta^{(i)}) \tag{7}$$

From (1), (4), (7), we have

$$\begin{aligned}
\theta^{(i+1)} &= \arg\max_\theta \left(LL(\theta^{(i)}) + \sum_Z P(Z|Y, \theta^{(i)}) \log \frac{P(Y|Z, \theta)P(Z|\theta)}{P(Z|Y, \theta^{(i)})P(Y|\theta^{(i)})} \right) \\
&= \arg\max_\theta \left(\sum_Z P(Z|Y, \theta^{(i)}) \log P(Y|Z, \theta)P(Z|\theta) \right) \\
&= \arg\max_\theta \left(\sum_Z \log P(Y, Z|\theta)P(Z|Y, \theta^{(i)}) \right) \\
&= Q\left(\theta, \theta^{(i)}\right)
\end{aligned} \tag{8}$$

It is obvious that this equation is equivalent to one step in EM algorithm. We can conclude that the essence of EM algorithm is to take steps to maximize its lower bound to approximate the maxima of itself. Figure[2] illustrates this idea. And we can also know that EM algorithm is not guaranteed to find the global maxima. Thus, the choice of initial values is very important.

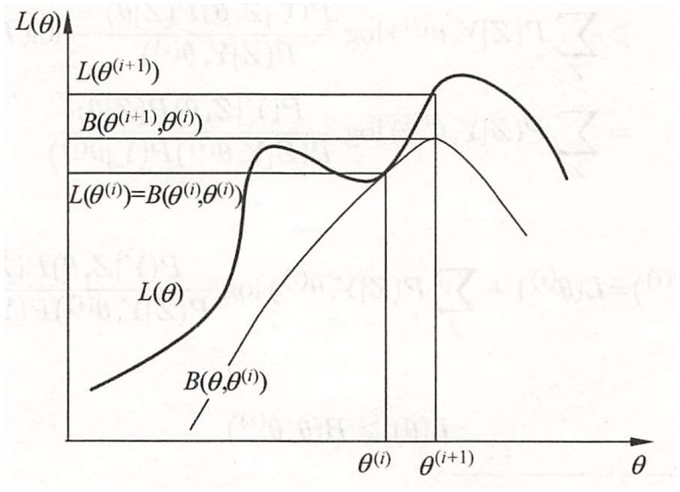

Figure 1: Geometric Explanation of EM Algorithm

Finally, I will give two theorems about the monotonicity and convergence of EM algorithm.

Theorem 2.1. *If $P(Z|\theta)$ is the likelihood function of observations, $\{\theta^{(i)}\}(i=1,2,...)$ is the parameter sequence estimated from EM algorithm, and $\{P(Z|\theta^{(i)})\}(i=1,2,...)$ is the corresponding likelihood function sequence. Then $P(Z|\theta^{(i)})$ is monotonically increasing, i.e.,*

$$P(Z|\theta^{(i+1)}) > P(Z|\theta^{(i)})$$

Proof. See Li (2019) [2]. ■

Theorem 2.2. *If $LL(\theta) = \log P(Y|\theta)$ is the likelihood function of observations, $\{\theta^{(i)}\}(i=1,2,...)$ is the parameter sequence estimated from EM algorithm, and $\{LL(\theta^{(i)})\}(i=1,2,...)$ is the corresponding log-likelihood function sequence.*

(1) If $P(Z|\theta)$ has a upper bound, then $LL(\theta) = \log P(Y|\theta)$ will converge to a value LL^*;

(2) When the functions $Q\left(\theta, \theta^{(i)}\right)$ and $LL(\theta)$ are under certain circumstances, convergence value θ^* of parameter sequence estimated from EM algorithm $\{\theta^{(i)}\}(i=1,2,...)$ is a stationary point of $LL(\theta)$.

Proof. See Wu (1983) [11]. ■

3 Problem Formulation

3.1 Hidden Markov Model (HMM) for Gaussian distribution[9]

This part is mainly from the website of Stan Uryasev (2017b).

Notations

$\{O\}$ = sequence of observations $\{O_1, ..., O_T\}$, where $O_t \in O$;
T = number of consequence time moments in sequence $\{O\}$;
m = number of hidden states to be considered in a model;
n = number of possible observations for discrete distribution;
$\vec{\lambda}$ = vector of decision variables consisting of ($\vec{\pi}$, A, B), where
 $\vec{\pi}$ = vector of probabilities of initial latent node such that $\sum_{i=1}^{m} \pi_i = 1$;
 A = state transition matrix with probabilities such that $\sum_{j=1}^{m} a_{ij} = 1$ for every $i = 1, ..., m$;
 B = observation symbol matrix with probabilities such that $\sum_{j=1}^{n} b_{ij} = 1$ for every $i = 1, ..., n$;
 (μ, σ) = parameters matrix of Gaussian distribution in each state for every $i = 1, ..., m$;
$LL_{HMM}\left(\vec{\lambda}, \{O\}\right) = \log\left(\vec{\pi}^T \cdot D_1 \left(\prod_{t=2}^{T} A \cdot D_t\right) \cdot \vec{\mathbf{1}}\right)$, where
 D_t= diagonal matrix with densities of probabilities $d_{ii}^t = \dfrac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\dfrac{(O_t-\mu_i)^2}{2\sigma_i^2}\right)$.

Optimization Problem

$$\max_{\vec{\lambda}} LL_{HMM}\left(\vec{\lambda}, \{O\}\right)$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i=1}^{m} \pi_i = 1 \text{ (Constraints on probabilities of initial states)} \\
& \sum_{j=1}^{m} a_{ij} = 1 \text{ (Constraints on probabilities of transitions in every state)} \\
& \pi_i \geq 0,\ a_{ij} \geq 0,\ \sigma_i \geq \epsilon \text{ (Box constraints on variables)} \\
& i, j = 1, ..., m.
\end{aligned}$$

Main Difficulty

It is actually a problem in the field of sequential learning in machine learning.

And we can see that there are tons of parameters ($m^2 + 3m$) in HMM. And the parameters are connected in a certain way. It is extremely difficult and computationally expensive to optimize the log-likelihood function directly. We must resort to some other technique, which is EM Algorithm.

3.2 Gaussian Mixture Model (GMM)[2]

This part is mainly from the textbook of Li (2019).

Notations

O = set of observations $(O_1, ..., O_T)$, where $O_t \in O$;
T = number of observations in empirical data O;
m = number of Gaussian distributions in the mixture model;
$\vec{\lambda}$ = vector of decision variables consisting of $(\vec{\alpha}, \vec{\theta})$, where
  α_j = weight of the jth Gaussian mixture, such that $\sum_{j=1}^{m} p_j = 1$;
  $\vec{\alpha}$ = vector of weights $(\alpha_1, ..., \alpha_m)$;
  $\vec{\theta}$ = emission matrix with parameters (μ_i, σ_i) of Gaussian mixtures for every $i = 1, ..., m$;

$$LL_{GMM}\left(\vec{\lambda}, O\right) = \sum_{t=1}^{T} \log\left(\sum_{k=1}^{m} \alpha_k \phi(O_t|\theta_k)\right)$$

Optimization Problem

$$\max_{\vec{\lambda}} LL_{GMM}\left(\vec{\lambda}, O\right)$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i = 1 \text{ (Constraints on probabilities of Gaussian mixtures)} \\ & \alpha_i \geq 0 \text{ (Box constraints on variables)} \\ & i = 1, ..., m. \end{aligned}$$

Main Difficulty

We can see that the log-likelihood function of GMM has even more severe problem than HMM, since there are summations inside the log() function, which means we can not benefit from taking derivatives of it. Similarly, we will see that EM Algorithm is able to handle this problem very efficiently.

Note

On the one hand, it is actually a problem in the field of unsupervised learning (clustering methods) in machine learning, which can be used to infer the latent distribution of a single observation.

On the other hand, we can replicate the empirical distribution by maximizing the log-likelihood function, where the result is unpredictable to some extent. In other words, in some circumstances, we mainly focus on the tail behavior, this method is not likely to control the tail behavior at our will. To solve the problem, we can introduce the following model.

3.3 Gaussian Mixture Model with CVaR distance and CVaR constraints[5][8]

This part is mainly from the paper of Pertaia and Uryasev (2018).

Notations

O = set of observations $(O_1, ..., O_T)$, where $O_t \in O$;
T = number of observations in empirical data O;

m = number of Gaussian distributions in the mixture model;
y_t = locations of atoms of empirical distribution, $t = 1, ..., T$;
$F_n(y)$ = empirical CDF at point y($\in R$), where

$$F_n(y) = \frac{1}{n}\sum_{t=1}^{T} \mathbf{1}_{\{y \geq y_t\}} \text{ will be used in the part of numerical experiment;}$$

$F_{\vec{p}}(y)$ = CDF of mixture model with weights $\vec{p}$ at point y($\in R$), where
p_j = weight of the jth CDF in the mixture, such that $\sum_{j=1}^{m} p_j = 1$;
$\vec{p}$ = vector of weights $(p_1, ..., p_m)$;
u = number of CVaR constraints;
k = index of CVaR constraints;
$\alpha(k)$ = confidence level, where $\alpha(k) \in (0,1)$, $k = 1, ..., u$;
$d_\alpha(F_n, F_{\vec{p}}) = CVaR_\alpha$ distance between $F_n(y)$ and $F_{\vec{p}}(y)$;
$CVaR_{\alpha(k)}(X_{\vec{p}}) = CVaR$ of mixture distribution with confidence level $\alpha(k)$;
$CVaR_{\alpha(k)}(Y) = CVaR$ of empirical distribution with confidence level $\alpha(k)$.
(Optional - for cardinality constraints)
M = maximum number of nonzero weights in the mixture.

Optimization Problem

$$\min_{\vec{p}} d_\alpha(F_n, F_{\vec{p}})$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i=1}^{m} p_i = 1 \text{ (Constraints on probabilities of Gaussian mixtures)} \\
& CVaR_{\alpha(k)}(X_{\vec{p}}) \geq CVaR_{\alpha(k)}(Y) \text{ (Constraints on tail distribution of mixtures)} \\
& p_i \geq 0 \text{ (Box constraints on variables)} \\
& i = 1, ..., m, \; k = 1, ..., u. \\
& \textbf{(Optional - for cardinality constraints)} \\
& \sum_{i=1}^{m} r_i \leq M \text{ and } p_j \leq r_j, j = 1, ..., m.
\end{aligned}$$

Note

As what I've mentioned before, the aim for introducing this model is mainly for adding some constraints on the tail behavior when we replicate the empirical distribution.

And the reason why we introduce CVaR distance is as follows. In general, EM algorithm solves a nonconvex optimization problem with respect to parameters of a mixture, which does not allow for additional constraints in the problem (Pertaia and Uryasev (2018)). In short, we want to introduce some convexity to our model to make tail constraints possible. And we will see later, this problem can be simplify to a convex optimization problem, while some other troubles are caused.

4 Problem Transformation

4.1 Hidden Markov Model (HMM) for Gaussian distribution[2]

This part is mainly from the textbook of Li (2019). For more details, see Chapter 10 of it.

Inspired by EM algorithm, first, we transform the log-likelihood function

$$LL_{HMM}\left(\vec{\lambda}, \{O\}\right) = \log\left(\vec{\pi}^T \cdot D_1 \left(\prod_{t=2}^{T} A \cdot D_t\right) \cdot \vec{\mathbf{1}}\right) = \log\left(\vec{\pi}_{m_1} b_{m_1}(O_1) a_{m_1 m_2} b_{m_2}(O_2) ... a_{m_{T-1} m_T} b_{m_T}(O_T)\right)$$

to complete-data log-likelihood function, and then to the Q-function

$$Q\left(\vec{\lambda}, \widehat{\vec{\lambda}}\right) = \sum_m \log \vec{\pi}_{m_1} LL_{HMM}\left(\widehat{\vec{\lambda}}, \{O\}\right) + \sum_m \left(\sum_{t=1}^{T-1} \log a_{m_t m_{t+1}}\right) LL_{HMM}\left(\widehat{\vec{\lambda}}, \{O\}\right)$$
$$+ \sum_m \left(\sum_{t=1}^{T} \log b_{m_t}(O_t)\right) LL_{HMM}\left(\widehat{\vec{\lambda}}, \{O\}\right)$$

where $\widehat{\vec{\lambda}}$ is the current estimation.

In this way, it is possible to estimate the three elements in $\vec{\lambda}$ separately, which is much simpler. And finally we can get the recurrence relation and repeat the formulas until convergence (the time at $n+1$), where the initial guess of $\vec{\lambda}^{(0)}$ is required.

$$a_{ij}^{(n+1)} = \frac{\sum\limits_{t=1}^{T-1} \xi_t(i,j)}{\sum\limits_{t=1}^{T-1} \gamma_t(i)}; \quad b_j(k)^{(n+1)} = \frac{\sum\limits_{t=1, o_t=v_k}^{T} \gamma_t(j)}{\sum\limits_{t=1}^{T} \gamma_t(j)}; \quad \pi_i^{(n+1)} = \gamma_1(i)$$

where $\alpha_t(i)$ denotes the probability of the partial observation sequence $\{O_1, O_2 \cdots O_t\}$(until time t) and state m_i at time t given the model $\vec{\lambda}$; $\beta_t(i)$ denotes the probability of the partial observation sequence from $t+1$ to the end, given state m_i at time t and the model $\vec{\lambda}$; $\gamma_t(i)$ denotes the probability of being in state m_i at time t, given the observation sequence $\{O\}$ and model $\vec{\lambda}$; and $\xi_t(i,j)$ denotes the probability of being in state m_i at time t, and state m_j at time $t+1$, given the observation sequence $\{O\}$ and model $\vec{\lambda}$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum\limits_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$
$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}$$

According to the assumption, the observations O_t at time t depend on the latent states m_j, and follows the distribution $N\left(\mu_j, \sigma_j^2\right)$. Thus, the last step is, to get all of Gaussian parameters μ_j's and σ_j^2's, which control the distributions at certain states, where $j = 1, 2, \cdots, m$. As we mentioned before, $(b_j(1), b_j(2), \cdots, b_j(n))$, and the frequency numbers of $(v(1), v(2), \cdots, v(n))$, all together reveal the characteristics of the underlying distribution (Gaussian distribution) in state j, where $\sum_{k=1}^{M} b_j(k) = 1$.

Finally, we want to derive $\mu_j^{(n+1)}$ and $\sigma_j^{2(n+1)}$ from them, and we can use the MLE estimators for Gaussian distribution, which is very straightforward

$$\widehat{\mu_j^{(n+1)}} = \frac{\sum\limits_{t=1, o_t=v_k}^{T} v(k) \cdot \gamma_t^{(n+1)}(j)}{\sum\limits_{t=1}^{T} \gamma_t^{(n+1)}(j)}$$
$$\widehat{\sigma_j^{2(n+1)}} = \frac{\sum\limits_{t=1, o_t=v_k}^{T} \left(v(k) - \mu_j^{(n+1)}\right)^2 \cdot \gamma_t^{(n+1)}(j)}{\sum\limits_{t=1}^{T} \gamma_t^{(n+1)}(j)}$$

4.2 Gaussian Mixture Model (GMM)[2]

This part is mainly from the textbook of Li (2019). For more details, see Chapter 9 of it.

Inspired by EM algorithm, we introduce latent indicator variables γ's, where $\gamma_{jk} = 1$, when observation O_j is from the kth Gaussian distribution and otherwise, $\gamma_{jk} = 0$, $j = 1, ..., T,\ k = 1, ..., m$.
And then we transform the log-likelihood function

$$LL_{GMM}\left(\vec{\lambda}, O\right) = \sum_{t=1}^{T} \log\left(\sum_{k=1}^{m} \alpha_k \phi(O_t|\theta_k)\right)$$

to complete-data log-likelihood function

$$\begin{aligned}
\log\left(P(O,\gamma|\vec{\lambda})\right) &= \log\left(\prod_{j=1}^{T} P(O_j, \gamma_{j1}, \gamma_{j2}, \ldots, \gamma_{jm}|\vec{\lambda})\right)\\
&= \log\left(\prod_{k=1}^{m}\prod_{j=1}^{T}\left[\alpha_k \phi(O_j|\vec{\lambda}_k)\right]^{\gamma_{jk}}\right)\\
&= \log\left(\prod_{k=1}^{m}\alpha_k^{n_k}\prod_{j=1}^{T}\left[\phi(O_j|\vec{\lambda}_k)\right]^{\gamma_{jk}}\right)\\
&= \log\left(\prod_{k=1}^{m}\alpha_k^{n_k}\prod_{j=1}^{T}\left[\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(O_j-\mu_k)^2}{2\sigma_k^2}\right)\right]^{\gamma_{jk}}\right)
\end{aligned}$$

where $n_k = \sum_{j=1}^{T}\gamma_{jk},\ \sum_{k=1}^{m} n_k = T$. And then transform it to Q-function

$$\begin{aligned}
Q\left(\vec{\lambda}, \hat{\vec{\lambda}}\right) &= E\left[\log\left(P(O,\gamma|\vec{\lambda})\right)\right]\\
&= \sum_{k=1}^{m}\sum_{j=1}^{T}(E\gamma_{jk})\log\alpha_k + \sum_{j=1}^{N}(E\gamma_{jk})\left[\log(\frac{1}{\sqrt{2\pi}}) - \log\sigma_k - \frac{1}{2\sigma_k^2(O_j-\mu_k)^2}\right]
\end{aligned}$$

In this way, the summations are eliminated by introducing latent indicator variables and transforming to Q-function. Then we can take the partial derivatives with regard to μ_k, σ_k^2 and α_k with constraint that the summation of α_k's is 1. And finally we can get the recurrence relation and repeat the formulas until convergence, where the initial guess of $\vec{\lambda}^{(0)}$ is required too

$$\hat{\gamma}_{jk} = \frac{\alpha_k \phi(O_j|\vec{\lambda}_k)}{\sum_{k=1}^{m}\alpha_k\phi(O_j|\vec{\lambda}_k)};\quad \hat{\mu}_k = \frac{\sum_{j=1}^{T}\hat{\gamma}_{jk}O_j}{\sum_{j=1}^{T}\hat{\gamma}_{jk}};\quad \hat{\sigma}_k^2 = \frac{\sum_{j=1}^{T}\hat{\gamma}_{jk}\left(O_j-\mu_k\right)^2}{\sum_{j=1}^{T}\hat{\gamma}_{jk}};\quad \hat{\alpha}_k = \frac{\sum_{j=1}^{T}\hat{\gamma}_{jk}}{T}$$

where all of the notations have the same meanings as the last subsection.

4.3 CVaR distance and CVaR constraints[5]

This part is mainly from the paper of Pertaia and Uryasev (2018).

We denote the CVaR of a random variable (r.v.) X at the confidence level $\alpha \in [0, 1)$ by $CVaR_\alpha(X)$.

$$CVaR_\alpha(X) = \min_C\left(C + \frac{1}{1-\alpha}E[X-C]^+\right)$$

In the last formula $[x]^+ = max(x, 0)$. It can be shown that $CVaR_0(X) = E(X)$. For a comprehensive analysis of the $CVaR_\alpha(X)$ risk-measure, see Rockafellar and Uryasev (2002[6], 2000[7]).

We denote by $||X||_\alpha$ the $CVaR_\alpha$-norm of X at the confidence level $\alpha \in [0, 1)$, as defined by Mafusalov and Uryasev (2016[3]):

$$||X||_\alpha = CVaR_\alpha(|X|)$$

Then the concept of $CVaR_\alpha$-distance between distributions can be introduced. The $CVaR_\alpha$-distance was defined by Pavlikov and Uryasev (2018[4]) in the context of discrete distributions.

Assume that there are two r.v.s Y and Z, with corresponding CDFs, $F(x)$ and $G(x)$. Assume also that there is some auxiliary r.v. H with CDF $W(x)$. We define a new r.v. X^W, representing the difference between $F(x)$ and $G(x)$, as

$$X^W(F, G) = F(H) - G(H)$$

Note that the auxiliary r.v. H may coincide with one of the r.v.s Y and Z, i.e., $W(x)$ may be equal to $F(x)$ or $G(x)$.

Definition 4.1 ($CVaR_\alpha$-distance)**.** *$CVaR_\alpha$-distance at some confidence level $\alpha \in [0, 1)$, between distributions of two r.v.s Y and Z with corresponding CDFs F_Y and G_Z is defined as*

$$d_\alpha^W(F, G) = ||X^W(F, G)||_\alpha$$

where H is an auxiliary r.v. with CDF W_H.

Knowing these basics, then we can introduce two important propositions.

Proposition 4.1. *$Q(\vec{p}) = d_\alpha^W(F, F_{\vec{p}})$ is a convex function of $\vec{p}$.*

Proof. See Pertaia and Uryasev (2018) [5]. ■

Proposition 4.2. *$CVaR_{\alpha(k)}(X_{\vec{p}})$ is a concave function of $\vec{p}$.*

Proof. See Pertaia and Uryasev (2018) [5]. ■

With $CVaR_\alpha$ constraints we ensure a specified fatness of the tail. For example, if we approximate some portfolio loss distribution by a mixture, we can guarantee that the $CVaR_\alpha$ of the mixture will be greater than or equal to some specified threshold.

Then the objective function is convex and the feasible region is the intersection of convex sets, thus it is a convex optimization problem.

Furthermore, we can add cardinality constraints to this model, but this part of content is trivial which can be transformed to a mixed integer programming problem (MIP).

5 Empirical Study

5.1 Coding Environment

Windows 10 + Python 3.8 + JupyterLab + PSG (for convex optimization).

5.2 Data

The original data is from TAQ database and data pre-processing procedure is in done in AMS 520, which is now in the form of tick data. And Apple stock is selected for analysis, then some methods in Python are used to get the data which are aggregated in 1min and 30min. And Figure[2] is the 1min returns for illustration.

5.3 Implementation of HMM

Among all of the models I have introduced, hidden markov model is the most popular model that can be applied to quantitative trading, where the hidden states are empirically used as a digital filter of financial sequences without delayed frequency response unlike MACD and many of the other technical indicators. And it can track trends in a much more flexible way by encoding non-linear relationship between the states. This allows for sudden changes to a new trend, whereas digital filters inevitably have some delay in response depending upon their frequency response (Christensen et al. (2020)).

And for HMM, it is more often to be used to fit high frequency data and develop some strategies in high frequency sense. Since in low frequency case, it will always yield some problems such as overfitting, distribution overlapping and so on.

In the following part, I will fit Apple stock data in 2020 sampled at 1min and 30min frequency with the models, which are on dynamic time period (9:30 AM to 4:00 PM). In this part of implementing, returns sampled at 1min frequency will be used for analysis.

Firstly, we are going to determine the number of hidden states, we can use penalized likelihood criteria to choose, such as Bayesian information criterion (BIC) (Schwarz, 1978) and Akaike information criterion (AIC) (Akaike, 1974). These criteria penalize the maximized likelihood function by the number of model parameters and data. The disadvantage, is that they do not provide any measure of confidence in the selected model (Christensen et al. (2020)).

Figure[3] shows the results and 4 is the optimal number of hidden states when we want a small number. And intuitively, there should not be so much hidden states in the model, which would make developing strategies very difficult. And 4 is already a very large number, while 2 or 3 are always better states number, which corresponds to bear, bull and fluctuating market.

Next, we can extract some of the fitted parameters for analysis, and use this best model to determine the states of historical data to check behavior of it as shown in Figure[4].

From the graph, we can see that the mean values of the distributions are very closed to zero and each other, while the standard deviations differ a lot. This is a not bad result. However, a very large proportion of time is occupied by state 1 and only three points belong to state 0 and 2, which means that we are impossible to use such a model to develop strategies.

Even though this fitting is far from satisfactory, we have to complete the whole procedure. Figure[5] is the transition matrix. And we can get probabilities for each state given the data, take the average to find the proportion of time in that state, then construct a Gaussian mixture to get its probability density function as shown in Figure[6].

Finally, there are two things to remark.

One is that I introduce frequency in log scale to the second graph of Figure[6], since all of the data are clustering in a very small range, which is very difficult to see what happened here. It can help us to see the tail behavior more clearly in these cases, though the graph looks a little bit strange.

The second is that this figure does not make a lot of sense, since it drops some very important information contained in the sequence, and similar thing could be done with other simpler model such as GMM. However, Figure[6] shows that the distribution is well-fitted to the empirical distribution of return.

5.4 Implementation of GMM

In the following parts, I will use returns sampled at 30min for illustration since there are no strict frequency requirements in these models.

Similarly, the first thing to do is to find the best number of states in this model. Results are shown in Figure[7], which states that 9 might be a good choice. However, we can notice that the curves are very flat when number of states exceeds 2, thus is hard to say which one is definitely the best.

In this part, we considered standard Gaussian mixture model, where the parameters are determined by maximizing the Log-likelihood function. And standard Gaussian mixture is a weighted sum of Gaussian distributions,

$$F_p(x) = \sum_{j=1}^{m} p_j \Phi(x, \mu_i, \sigma_i)$$

where $\Phi(x, \mu_i, \sigma_i)$ is a Gaussian CDF with mean μ_i and standard deviation σ_i. We estimated μ_i and σ_i with EM algorithm. The estimated parameters of the mixture are presented in the Table[1].

For the mixture with parameters in Table[1] and the empirical distribution, we have calculated $CVaR_{0.90}$, $CVaR_{0.95}$, $CVaR_{0.99}$, $CVaR_{0.995}$ and $CVaR_{0.999}$, see Table[2].

From Table 2, we know that the standard Gaussian model underestimate the behavior of the empirical distribution, which means it tends to overestimate the loss and thus gives a inaccurate estimation. Furthermore, when we try GMM model with 5 hidden states, a completely different result will be seen, which tends to underestimate of loss a lot. These facts shows that the tail behavior of GMM model is hard to control and predict.

5.5 Implementation of GMM with CVaR distance and CVaR constraints

Further we minimized CVaR distance. We constrained CVaRs of the mixture to be greater or equal to the empirical CVaRs, which makes sure that at least we are not going to underestimate the loss.

Optimal weights of the mixture are updated in Table[3]. And the CVaRs for the mixture with CVaR constraints are in Table[4].

Table[4] shows that all of the CVaR constraints are satisfied except the last one when $\alpha_k = 0.999$. It is mainly because that the CVaR computation error and it is actually exactly satisfied. Figure[8] will illustrate that CVaR distance and constraints actually control the tail behavior of the target distribution very well compared to standard GMM.

6 Conclusions and Future works

In the last section, HMM has got very poor performance. I think it is partial caused by that I have fitted too much data to the model, which is almost 100000 samples. In this case, this model is hard to detect the local violation of returns, since most of the returns are around the mean value. Therefore, the current trend is compensated by the future information. Maybe we can try a smaller number of date.

On the other hand, maybe we can use some more advanced HMM model, such as Markov-Switching model, which allows to switch in different models (Zucchini and MacDonald (2009)). And as for the model with CVaR distance and CVaR constraints, it can efficiently control the tail behavior of the replicated distribution while GMM is likely to yield some inaccurate estimation of the tail behavior.

However, the choice of means, variances and number of hidden states in this model might be a big headache if we want to completely rely on this method to optimize.

Additionally, we can see that in Figure[8], the other side of tail behaves much more poor, which deviates a lot from the empirical distribution. But we do not have to worry a lot about it, since in risk management, we always only focus on the tail of loss.

References

[1] Christensen, H., Godsill, S., and Turner, R. E. (2020). Hidden markov models applied to intraday momentum trading with side information.

[2] Li, H. (2019). *Statistical Learning Methods*. Tsinghua University Press, 2 edition.

[3] Mafusalov, A. and Uryasev, S. (2016). Cvar (superquantile) norm: Stochastic case. *European Journal of Operational Research*, 249(1):200–208.

[4] Pavlikov, K. and Uryasev, S. (2018). CVaR distance between univariate probability distributions and approximation problems. *Annals of Operations Research*, 262(1):67–88.

[5] Pertaia, G. and Uryasev, S. (2018). Fitting mixture models with cvar constraints.

[6] Rockafellar, R. and Uryasev, S. (2002). Conditional value-at-risk for general loss distributions. *Journal of Banking Finance*, 26(7):1443–1471.

[7] Rockafellar, R. T. and Uryasev, S. (2000). Optimization of conditional value-at risk. *Journal of Risk*, 3:21–41.

[8] Stan Uryasev (2017a). Case study: Fitting mixture models with cvar constraints. http://uryasev.ams.stonybrook.edu/index.php/research/testproblems/advanced-statistics/fitting-mixture-models-with-cvar/.

[9] Stan Uryasev (2017b). Case study: Maximization of log-lokelihood in hidden markov model. http://uryasev.ams.stonybrook.edu/index.php/research/testproblems/financial_engineering/case-study-maximization-of-log-lokelihood-in-hidden-markov-model-hmm_discrete-hmm_normal-linear-linearmulti/.

[10] Wikipedia contributors (2022). Expectation–maximization algorithm — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Expectation%E2%80%93maximization_algorithm&oldid=1116151599.

[11] Wu, C. J. (1983). On the convergence properties of the em algorithm. *The Annals of statistics*, pages 95–103.

[12] Zucchini, W. and MacDonald, I. L. (2009). *Hidden Markov models for time series: an introduction using R*. Chapman and Hall/CRC.

Appendix A Tables

j	μ_j	σ_j	p_j
1	0.000172	0.002354	0.490991
2	-0.003086	0.002789	0.239113
3	0.004719	0.003106	0.151332
4	-0.009875	0.004006	0.050093
5	0.011826	0.004541	0.039690
6	-0.022249	0.005588	0.015767
7	0.032169	0.011852	0.008657
8	-0.046155	0.011268	0.004055
9	0.101885	0.001000	0.000303

Table 1: Parameters of standard Gaussian distributions in the mixtures

k	$\alpha(k)$	$CVaR_{\alpha(k)}(X_p)$	$CVaR_{\alpha(k)}(Y)$	**Difference**
1	0.900	0.012053	0.011898	0.000155
2	0.950	0.016786	0.016611	0.000175
3	0.990	0.034029	0.033533	0.000497
4	0.995	0.044520	0.042891	0.001630
5	0.999	0.068586	0.066246	0.002339

Table 2: CVaRs of empirical distribution and Gaussian mixture fitted by the EM algorithm. $CVaR_{\alpha(k)}(X_p)$ is the CVaR of mixture with confidence $\alpha(k)$ and $CVaR_{\alpha(k)}(Y)$ is the CVaR of empirical distribution. The entries in "Difference" column are $CVaR_{\alpha(k)}(X_p) - CVaR_{\alpha(k)}(Y)$.

j	μ_j	σ_j	p_j
1	0.000172	0.002354	0.616539
2	0.004719	0.003106	0.141907
3	-0.009875	0.004006	0.102450
4	-0.003086	0.002789	0.090835
5	0.011826	0.004541	0.032120
6	0.032169	0.011852	0.015937
7	0.101885	0.001000	0.000212
8	-0.022249	0.005588	0.000000
9	-0.046155	0.011268	0.000000

Table 3: Parameters of Gaussian distributions in the mixtures with CVaR distance and CVaR constraints

k	$\alpha(k)$	$CVaR_{\alpha(k)}(X_p)$	$CVaR_{\alpha(k)}(Y)$	**Difference**
1	0.900	0.013347	0.011898	0.001449
2	0.950	0.019450	0.016611	0.002839
3	0.990	0.040869	0.033533	0.007337
4	0.995	0.048253	0.042891	0.005362
5	0.999	0.066226	0.066246	-0.000020

Table 4: CVaRs of empirical distribution and Gaussian mixture fitted by minimizing CVaR distance with CVaR constraints. $CVaR_{\alpha(k)}(X_p)$ is the CVaR of mixture with confidence $\alpha(k)$ and $CVaR_{\alpha(k)}(Y)$ is the CVaR of empirical distribution. The entries in "Difference" column are $CVaR_{\alpha(k)}(X_p) - CVaR_{\alpha(k)}(Y)$.

Appendix B Graphs

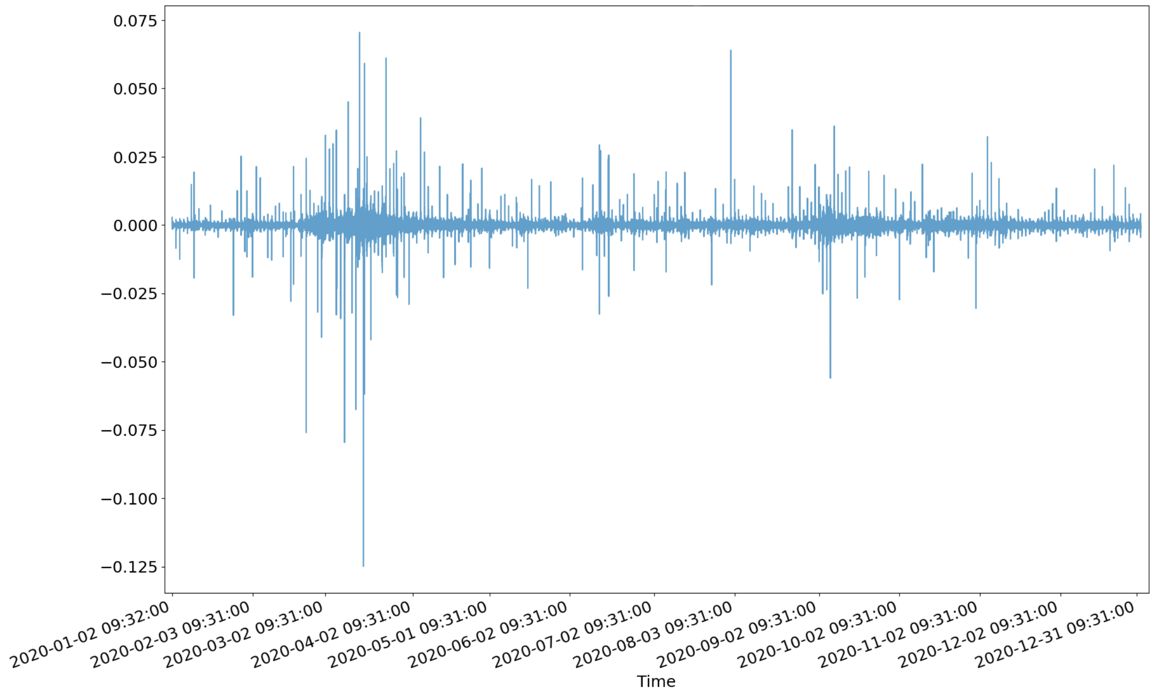

Figure 2: AAPL stock return - 1min frequency (whole year)

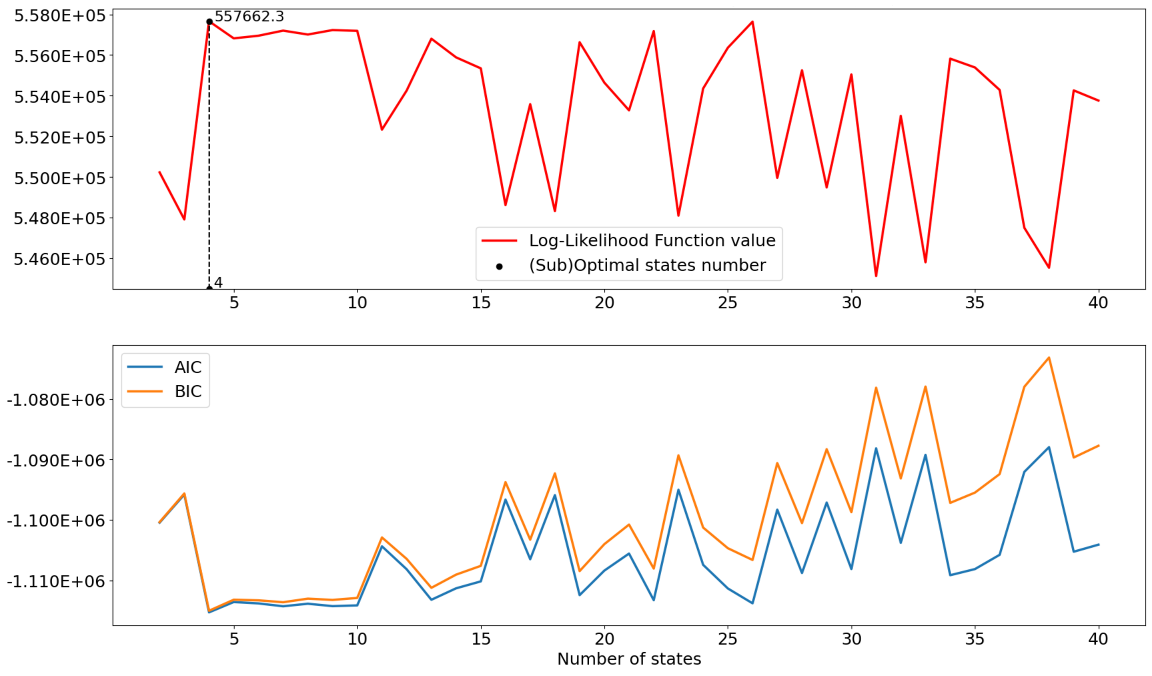

Figure 3: AAPL stock HMM model selection - 1min frequency (whole year)

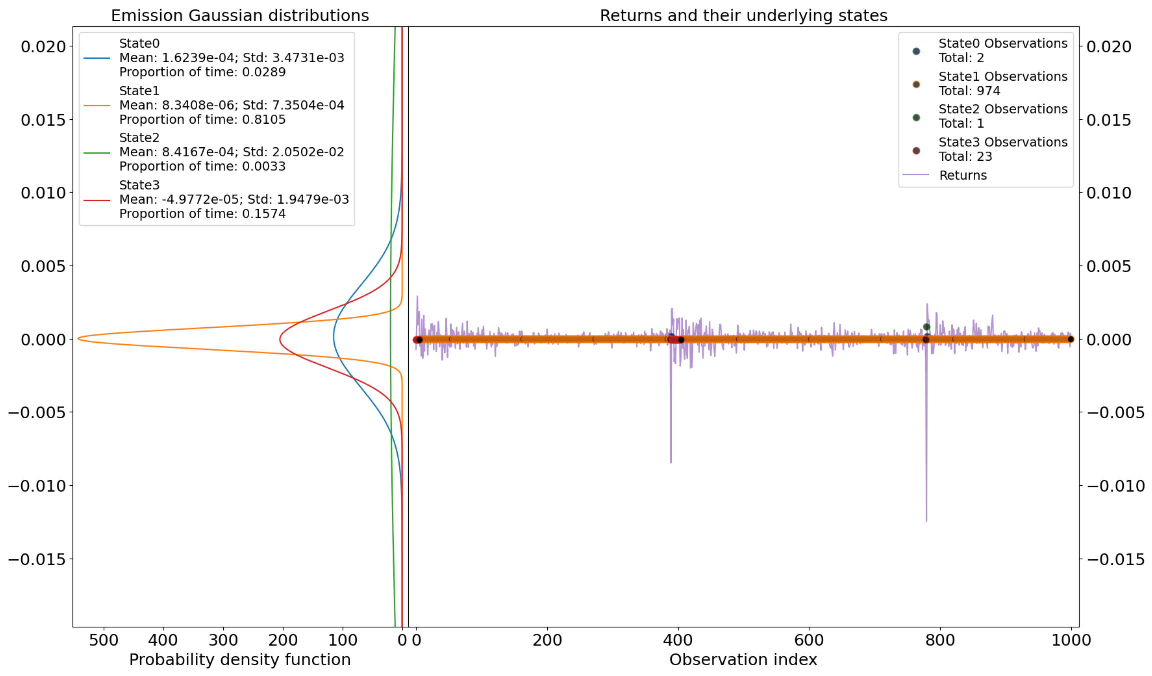

Figure 4: Results of HMM fitting

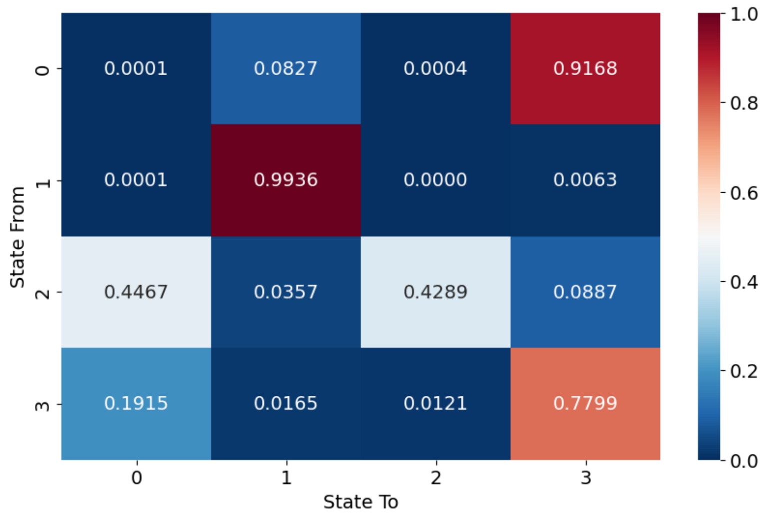

Figure 5: Transition Matrix of HMM

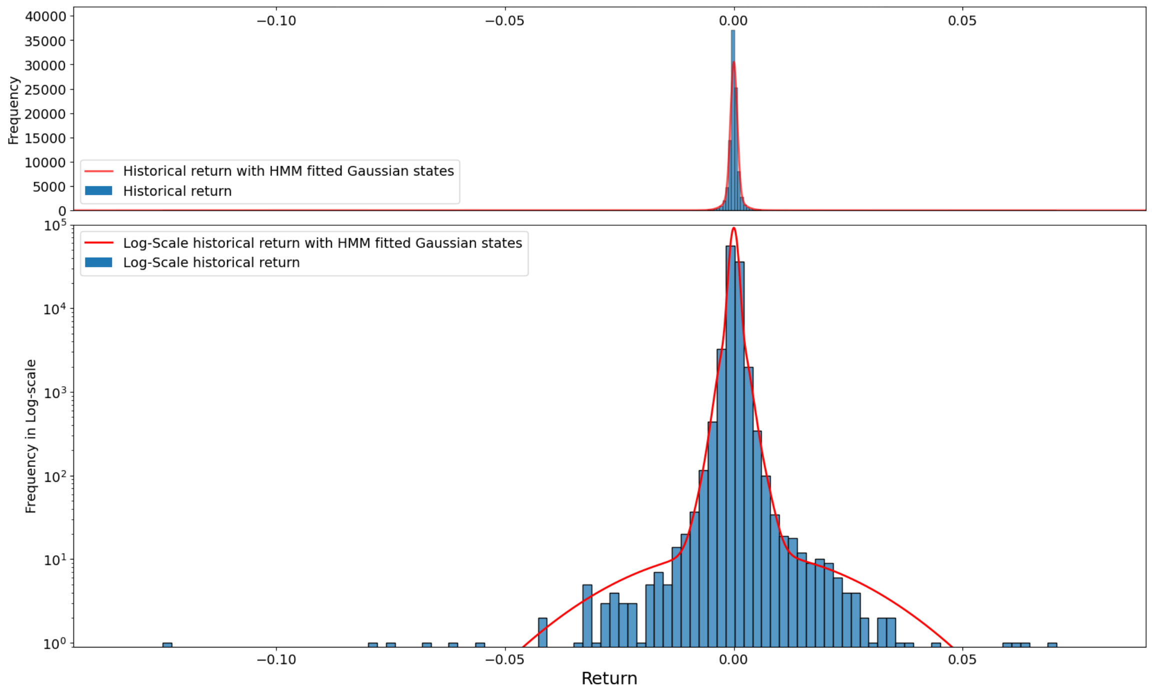

Figure 6: Histogram of return and return with HMM fitted Gaussian states

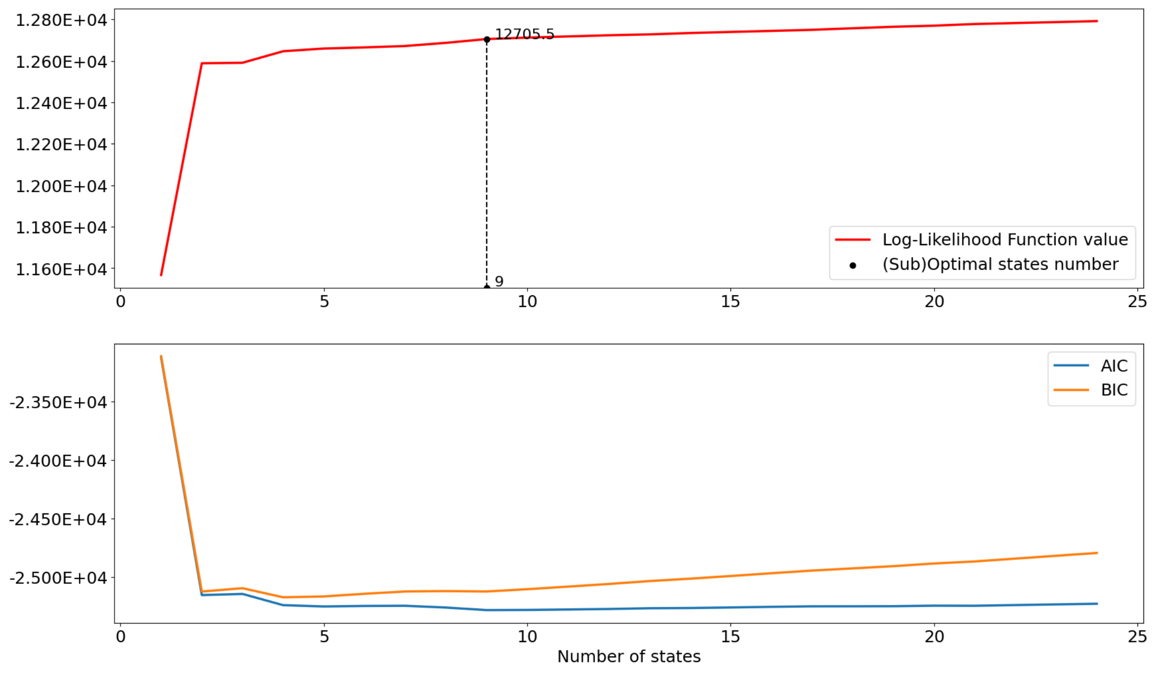

Figure 7: AAPL stock GMM model selection - 30min frequency (whole year)

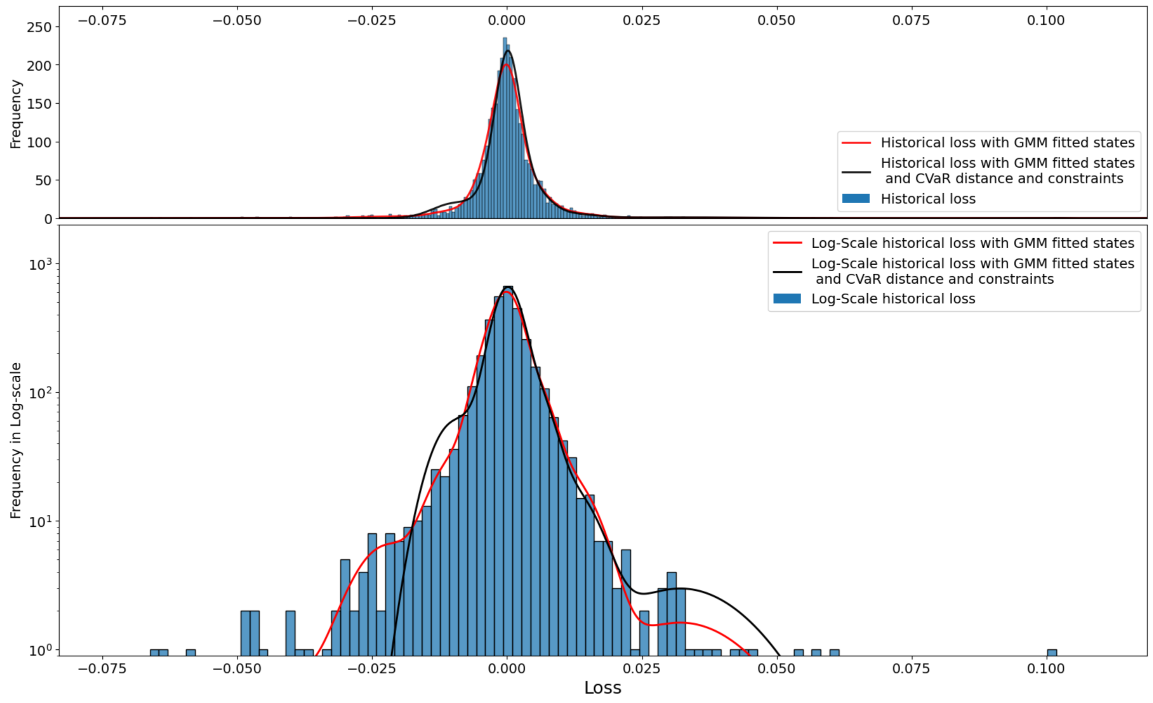

Figure 8: Histogram of loss and loss with GMM fitted states / loss with GMM fitted states and CVaR distance and constraints