Intelligent Automation And Soft Computing, 2020
Copyright © 2020, TSI® Press
Vol. 26, no. 2, 227–236
https://doi.org/10.31209/2019.100000068

Self-Organizing Gaussian Mixture Map Based on Adaptive Recursive Bayesian Estimation

He Ni[1], Yongqiao Wang[1], Buyun Xu[2]

[1]. School of Finance, Zhejiang Gongshang University, Hangzhou, China
[2]. Hangzhou College of Commerce, Zhejiang Gongshang University, Hangzhou, Tonglu, China

ABSTRACT
The paper presents a probabilistic clustering approach based on self-organizing learning algorithm and recursive Bayesian estimation. The model is built upon the principle that the market data space is multimodal and can be described by a mixture of Gaussian distributions. The model parameters are approximated by a stochastic recursive Bayesian learning: searches for the maximum a posterior solution at each step, stochastically updates model parameters using a "dual-neighbourhood" function with adaptive simulated annealing, and applies profile likelihood confidence interval to avoid prolonged learning. The proposed model is based on a number of pioneer works, such as Mixture Gaussian Autoregressive Model, Self-Organizing Mixture Map, and have some favoured attributes on its robust convergence and good generalization. The experimental results on both artificial and real market data show that the algorithm is a good alternative in measuring multimodal distribution.

KEY WORDS: Adaptive simulated annealing, Density approximation, Mixture distribution, Profile likelihood confidence interval, Self-Organizing Map

1 INTRODUCTION

THE underlying distribution of many real time market data are usually assumed to be uni-modal. However the assumption does not consider those exceptional market shocks which are always influential in measuring market impacts. Fong and Wong (2008) argued that the probability of these shocks can be underestimated if the multimodal components are neglected. As a matter of fact, the fat tail pattern, which can be treated as a multimodal distribution, has been empirically and commonly encountered. Market events/shocks do not follow central limit theorem, but are usually not mathematically well-behaved. The well known "80-20" rule (Also known as Pareto principle that is for many events, roughly 80% of the effects come from 20% of the causes) is a manifestation of a fat tail distribution. People thus start to assume such heterogeneous economic market data to be generated from a mixture of Gaussian distribution (i.e. Wong and Li (2008), Nguyen and Wu (2017)) in a sense that a normal Gaussian distribution with a fat tail can be treated as a minor Gaussian distribution centered on the tail of the major Gaussian distribution. Instead of estimated the parameters of a Gaussian with fat tail, it would be more sensible to estimate a few normal Gaussians that can be used to form that fat tail Gaussian. The idea can also be regarded as local model approach.

The purpose of this study is to explicitly model the density of a number of possible observations by a probabilistic vector quantization algorithm. One of such algorithms is known as Gaussian Mixture Model, which has been widely used to describe a mixed distributions that represents the probability distribution of a number of sub-populations within an overall population, such as Seo and Thorson (2016), McLachlan, et al. (2004), Zhuang, et al. (1996), Jacobs, et al. (1991). The observed density is represented by a weighted sum of component densities, while the weighting factors are all non-negative and sum to one. By using such a weighting scheme, the mixture model can be used to model a rich class of densities by assuming a number of sources that generate the data each with a probability equal to its mixing weight. The model can generate an entire distribution of the parameters of interests, leaving users to decide which quantile to report. In

CONTACT He Ni nihe@zjgsu.edu.cn

prediction of the return of a financial instrument, the mixture model can be a comprehensive statistic inference of underlying investment risks. Moon (1996), Xu and Jordan (1996) suggested that a conventional approach being used to estimate the parameters of a mixture model is the Expectation Maximization (EM) algorithm. The EM algorithm is an iterative procedure that can find the local maximum of a certain log-likelihood function. The popularity of EM algorithm comes from its easy implementation and its robustness in finding a local maximum of the log-likelihood function. However the EM algorithm may not always be able to reach the global minimum, as sometimes it can be sensitive to the initial status of the parameters. Besides, the EM algorithm, on its original form, is a batch algorithm which can be inefficient in processing online data and is not always computationally efficient as being argued by Figueiredo and Jain (2002) and Verbeek, et. al. (2003).

Successful applications of Self-Organizing Map (SOM) by Heskes (2001), Verbeek, et. al. (2003) on iteratively estimating parameters of mixture model have appeared on literature for ages. Yin (2006) proved that SOM is a simplified Gaussian, while the neighborhood is functioning as the variance of Gaussian. In general, SOM enables a representation of a higher dimensional data space by a lower dimensional latent variable space with a spatial organizing algorithm. The neighborhood function enables SOM to preserve topology by keeping similar clusters “nearby”. Heskes (2001) showed that SOM is a popular clustering method applied to many engineering problems. However, SOM on its original form suffers from inherent deficiencies, such as the absence of a cost function, the lack of a theoretical basis for choosing learning rate and neighborhood parameters, and no definition of a probability density.

In this paper, we propose a SOM alike learning algorithm using probabilistic mixture model. The proposed algorithm uses a recursive Bayesian approach which can yield a posterior distribution based on both the informative prior (if there is any) and data samples. Since we aim to analysis patterns buried in market data of large size, the proposed algorithm uses iterative learning process, that is, the data is processed online rather than in a batch. The posterior distribution at each step of learning is a revision of the previous prior. The magnitude of revision reflects the information contains in samples. In order to avoid the improper initialization/learning, such as trapped in a local minima, the adaptive simulated annealing technique is applied to “distort” the originally monotonically decreasing learning rate. Meanwhile, we further impose an additional topological constraint on the parameter updating. That is although the updating Gaussian variance is well functioning as the neighborhood function of SOM, the topology order/shape of a certain choice of lattice can apply a further control on the learning process at a little cost of computational efficiency. The structure of lattice should be carefully chosen based on the prior. Finally in order to prevent unnecessary learning, a profile likelihood confidence interval has been used to cut down the maximum number of iterations. The proposed algorithm can be useful in other applications provided the data has clustering nature, i.e. cross-sectional analysis of temporal data and can reach the solutions that are similar to or even better than those obtained by batch EM operations.

2 METHODOLOGY

One of the key challenges of mixture model estimation is to find a number of appropriate density parameters $\theta_k = \{\mu_k, \Sigma_k\}$ of the model/density k. There is a trade-off between the precision of estimated parameters and the computation efficiency. When Σ_k is assumed to be a diagonal covariance matrix, the distribution of sub-models is of an ellipse shape with its horizontal axes either parallel or orthogonal to data ordinates. If Σ_k is a spherical covariance matrix, the sub-model distributions are circle-alike with different radiuses. In this paper, we define Σ_k as a full covariance matrix, the distribution of sub-models can be any tilted ellipse shape. In order to make the parameters estimation not only precise but also efficient in a sense of robustness in its convergency, attempts have been made to integrate a SOM alike general learning algorithm that can be applied to any probabilistic mixture model. Under such probabilistic framework, the neighborhood function of Kohonen SOM can be interpreted as a distribution over the component. Yin (2002) and Verbeek, et. al. (2005) re-derived the original SOM algorithm through defining a lower bound which consists of the data log-likelihood and a non-negative Kullback-Leiler divergence. Since Cover (1991) proved the Kullback-Leiler divergence has a form of entropy, it can be regarded as a penalty which is high when the true posterior is far from its estimation.

2.1 Gaussian Mixture with classic EM approach

Probability distribution of a process $f(X)$ can be represented by a Gaussian mixture if it is defined as a convex combination of Gaussian densities. A Gaussian density k defined in a d-dimensional space can be defined as

$$\phi(X,\theta_k) = (2\pi)^{-d/2} \det(\Sigma_k) \exp\left(\frac{-(X-\mu_k)^T (X-\mu_k)}{\Sigma_k} \right) \quad (1)$$

and a Gaussian mixture is then defined as

$$F(X \mid \Theta) = \sum_{k=1}^{K} \pi_k \phi(X, \theta_k) \quad (2)$$

where π_k , which is introduced previously as a mixture probability, can be treated as a prior probability as well as a self-organizing adaptive parameter or weight.

The classic approach used to estimate time invariant parameter Θ is to maximize the joint likelihood $P(X \mid \Theta) = \prod_t F(X(t) \mid \Theta)$ from a group of independent (ideally) samples X_t . The Expectation-Maximization (EM) algorithm can be used to update the parameters Θ iteratively.

For each input data sample X_t , a discrete latent variable (The latent variable is always a binary sequence. The number of elements in the sequence is equal to the number of mixture density K), c_t is defined to indicate which Gaussian component might generate (by how much in percentage) the input data sample X_t . The mixture probability π_k can be thus generated by,

$$\pi_k = \frac{1}{T}\sum_{t=1}^{T} P(c_t \mid X_t, \theta_k) \tag{3}$$

The density parameters $\theta_k = \{\mu_k, \Sigma_k\}$ of the sub-model or sub-density k can be generated by,

$$\mu_k = \frac{\sum_{t=1}^{T} P(c_t \mid X_t, \theta_k) X(t)}{\sum_{t=1}^{T} P(c_t \mid X_t, \theta_k)} \tag{4}$$

$$\Sigma_k = \frac{\sum_{t=1}^{T} P(c_t \mid X_t, \theta_k)(X(t)-\mu_k)(X(t)-\mu_k)^T}{\sum_{t=1}^{T} P(c_t \mid X_t, \theta_k)} \tag{5}$$

where $k \in 1,\ldots,K$ and the probability of the data sample X_t generated by Gaussian component k is denoted by

$$P(c_t \mid X_t, \theta_k) = p_k(X_t) \tag{6}$$

Thus, each sub-model k has an associate probability $p_k(X_t)$.

However, the EM algorithm can not guarantee the best solution, which implies that the likelihood may not be bounded and the solution yielding maximal joint likelihood may not be found. In fact, Lindsay (1983) found that the classic EM approach may lead to a singular estimation of parameters.

2.2 Self-organizing Estimation

The proposed model is defined by a mixture of K Gaussian densities. A bit different from the updating mechanism of conventional finite Gaussian mixture model, at each time step t, the model is presented by a data sample $X(t)$, then a winning Gaussian component and its neighbors (equal or less than $K-1$) are updated in a self-organizing manner.

Only one Gaussian component, which is defined as the winning component of the mixture model, is suppose to have better chance (i.e. with maximum likelihood or highest posterior probability) to generate the input data sample X_t . The probabilities of other Gaussian components that may generate input data sample X_t decrease (proportionally) according to the topological distance (i.e. similarity) to that winning component. The topology can be defined by a rectangular lattice, a hexagonal lattice or any other geometrical structures that meets the needs of the sample data.

Heskes (2001) had concluded that algorithm of self-organizing map for vector quantization is similar to mixture modeling in many applications. Self-organizing has a neighborhood structure which can be used to regulate the learning process. Parameters of each Gaussian distribution (similar as the neuron of SOM) can be estimated iteratively following a SOM manner - a guided EM process. Giving a piece of input data sample X_t , a good quantizer W_k with a low quantization error should have high probability to find the input X_t . Hence, we may use a confusion probability $g_k(i)$ which means the input sample is generated by mixture component i instead of k .

$$g_k(i) = p_k(c_t = i \mid g_k) \tag{7}$$

the probability $g_k(i)$ is a distribution around the center unit k . The probability $g_k(k) \geq g_k(i)$ which means the sub-model is uni-modal, while $g_k(i_1) \geq g_k(i_2)$ if i_1 is topologically close to k than i_2 in the lattice. This definition which is originally derived by Verbeek (2005) is inspired by the neighborhood function of self-organizing map. The probability $g_k(i)$ can be approximated by a Gaussian kernel / neighborhood function.

$$g_k(i) = \lambda(t)\exp\left(-\frac{\delta_{i,k}^2}{\sigma_k(t)}\right) \tag{8}$$

where $\delta_{i,k}^2$ represents a topological distance (i.e. a similarity measure) which is commonly used in the original SOM algorithm, $\sigma_k(t)$ and $\lambda(t)$ are two positive decreasing learning rates shrinking with time t which are quit sensitive and need to be estimated empirically. The quantization error can thus be defined by using the confusion probability $g_k(i)$,

$$e = \sum_{t=1}^{N}\sum_{k=1}^{K} p_k(X_t)\sum_{\eta_i} g_k(i)\left\|X_t - W_t\right\|^2 \tag{9}$$

where η_i is a neighbor of k and W_t is the reference weight vector of neuron i in the SOM.

$p_k(X_t)$ denotes the probability that data sample X_t is assigned to neuron k. Since the confusion probability $g_k(i)$ corresponds to the lateral-interaction strength, it is a decreasing function of the distance between neurons k and i. The definition of $g_k(i)$ enforces the self-organization of the network. It assigns a decreasing probability to neuron i as their topological distance to neuron k increases, and the value of $g_k(i)$ can indicate whether the neuron i would be updated ($g_k(i)=0$ means not update) and by how much (in case when $g_k(i)>0$). The confusion probability $g_k(i)$ can thus be taken as the neighborhood function which is also proportional to the rate of update in the SOM.

Meanwhile, $p_k(X_t)$ is an updated probability calculated by the updated parameters of neuron k which corresponding to a maximum likelihood in the previous step of parameters learning. Thus for certain input X_t, there will be a neuron ω with $p_\omega(X_t) \geq p_k(X_t)$ where $k \neq \omega$. If we define ω as the winning neuron with the highest posterior probability, neurons have relatively lower posterior probabilities forms a "neighborhood" of winner ω. However, this neighborhood doesn't include any topological information. The topology thus can be automatically formed by the distribution of posterior probabilities rather than being forced to follow a certain lattice structure (i.e. introduction of confusion probability $g_k(i)$).

In order to maintain the topological preserving property given some useful prior information, we include both pre-defined neighborhood which is the confusion probability $g_k(i)$ and a posterior probability $p_k(X_t)$ during the learning process. Thus we could have "double" neighborhood functions to ensure the convergence of the proposed SOM alike EM algorithm.

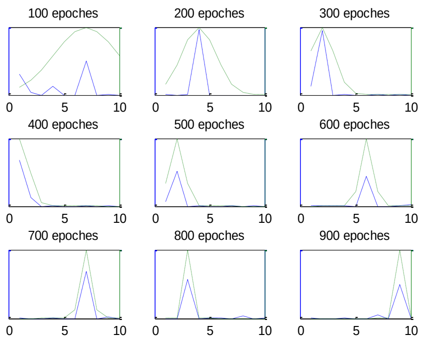

Figure 1. "Shape" of the neighbourhood functions and the posterior probabilities at different learning epochs (from 100 epochs in the upper left to 900 epochs in the lower right)

In order to show the effects of "double" neighborhood functions as well as other features of the proposed algorithm, we implement the algorithm with a set of artificial data (i.e. with five mixed centers illustrated by the upper-half of Figure2). The evolution process of neighborhood function of one of the mixed centers is shown on Figure1. It shows that both the $g_k(i)$ and $p_k(X_t)$ can gradually catch the center (i.e. location of the winning unit) and its neighbors in general. The evolution process (compromising) of neighborhood in Figure1 shows that the "shape" of the neighborhood are not similar at the beginning, but will be able to gradually get converged.

The use of double neighborhood functions makes the algorithm more capable in accommodating not only the "nature" distribution $p_k(X_t)$, but also the "man-made" topological conservation $g_k(i)$. Thus more robust when handling data with difference properties. The double neighborhood is one of major inventions of this study and is the most important connection between SOM algorithm and Gaussian mixture model. Successful implementations of the proposed algorithm on both artificial data and real world data verify the double neighborhood setting is a good attempt in incorporating self-organizing parameter estimation into the proposed algorithm.

2.3 Recursive Bayesian Approach

The Bayesian approach can improve the SOM's classification ability in dealing with Gaussian mixture problem. In a finite SOM lattice with K units, each unit (or sub-model) can be parameterized by a Gaussian kernel $\theta_k = \{\mu_k, \Sigma_k\}$ and a mixture weight π_k. The mixture weight is used to be initialized by a random value or equals to $1/K$ and being updated accordingly.

In a Bayesian approach, the past π_k (or $P(c)$ whose average over all samples is π_k) is treated as a prior. At each time step, the posterior probability of each Gaussian unit is updated/estimated by

$$P(c_k \mid X_t, \theta_k) = \frac{P(X_t \mid c_k, \theta_k) P(c_k)}{\sum_{i=1}^{K} P(X_t \mid c_k, \theta_k) P(c_k)} \quad (10)$$

The winning unit is defined as the one with highest posterior probability at each epoch. It shows that the way to find the best matched unit on the self-organizing map is coincident with the Maximize a Posterior (MAP) approach.

In order to update the parameters of network iteratively and efficiently, a popular approach is to define the "direction" of learning by minimizing the Kullback-Leibler information metric which has been detailed in Runnalls (2007). The partial derivatives of the information metric, with respect to unknown model parameters can be used to form a recursive stochastic approximation method, e.g. the Robbins-Monro method (utilized by Byrd (2016) and Banks (2017)), which can be used to find these parameters. The following updating equations are then obtained.

$$\Delta\pi_k(t) = \alpha\left(\frac{g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)}{\sum_{i=1}^{K} g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)} - \pi_k(t)\right) \quad (11)$$

$$\Delta\mu_k(t) = \alpha(t)\frac{g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)}{\sum_{i=1}^{K} g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)}\left(\mathbf{x}(t) - \mu_k(t)\right) \quad (12)$$

$$\Delta\Sigma_k(t) = \alpha(t)\frac{g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)}{\sum_{i=1}^{K} g_k(i)P(c_t \mid \mathbf{x}_t, \theta_k)}\Omega \quad (13)$$

$$\Omega = \left(\mathbf{x}(t) - \mu_k(t)\right)^T\left(\mathbf{x}(t) - \mu_k(t)\right) - n\Sigma_k(t)$$

where $\alpha(t)$ is a step size or learning rate of the Robbins-Monro algorithm which is always monotonically decreasing and must satisfy some mild constrains. It used to be in a form of $1/(at+b)$ or $1/(at+1/a)$ and $0 < a < 1$, with higher values of α bring more variability to the estimations.

The learning process sometime is, however, likely to trap into local optima, due to the effects of "double" neighborhood. Although these two neighborhood functions might have different topological structure, but both of them could very much likely have a same wining neuron with its Gaussian kernel $\tilde{\theta}_k$ neighborhoods, as being depicted in Figure 1.

$$\tilde{\theta}_k = \arg\max_{\theta} P(c_t \mid \mathbf{x}_t, \theta_k) \quad (14)$$

By integrating the topological constraint, the neighborhood function $g_k(i)$ depresses the learning rates for those neighbor neurons. Whilst the winning neuron is updated relatively faster than its neighbors by imposed $g_k(i)$. It will consequently weaken the neighborhood effects which aim to reduce the possibility that the learning process may trap into local minima.

2.4 Adaptive Learning Process

In order to avoid the side effect of "dual-neighborhood" function, we should extend the original learning process or make it adaptive to the learning speed. The value of likelihood based on the updating parameters should increase during the learning process. However, if the value of likelihood is gaining too quick, the learning process may also trap into a local minimum. To make the algorithm adaptive to the adverse effects of "dual-neighborhood" function and achieves a good balance, we propose an annealing algorithm which can slow down or re-start the learning process from an earlier point (during the learning process) when necessary.

Annealing is a mechanism/process during which the temperature of some metal being dynamically cooled down to make it reaches a state of low energy. Shown by Fan (2015), simulated annealing is a process starting from a random search at high temperature and gradually changes to a greedy descent as the temperature approaches zero. In our case, if the newly learnt parameters can generate a higher likelihood value (i.e. better fit to the sample data) then we can accept the assignment of this new parameters. Otherwise, if in a case of lower likelihood value, we may still accept the parameters, but with some relatively small probability, depending on the temperature (i.e. stage of learning) and how much it is worse than the previous parameters. Simulated annealing requires an annealing schedule, which specifies how the temperature is reduced as the search progresses and there is no guideline to choose an appropriate schedule. Thus the parameter setting is done by trial-and-error.

In this paper, we imposed the annealing process on the learning rate $\alpha(t)$ of the Robbins-Monro algorithm. Instead of using a monotonically decreasing variable, we made $\alpha(t)$ adaptive to the posterior probability $P(\mathbf{x}_t \mid c_k, \theta_k)$ as being shown by the Pseudo-code of Annealing Schedule as follows,

Pseudocode of Annealing Schedule

```
Initialization: $L_1^t, L_2^t, L_3^t, L_1^T, L_2^T, L_3^T$
Require: $t \geq L_1^t; L_1^t < L_2^t < L_3^t < \max(t), 0.5 < L_1^T < L_2^T < L_3^T < 1$
 1: $T \leftarrow t$
 2: if $\sum_k \log P(\mathbf{x}_t|c_k, \theta_k) > \sum_k \log P(\mathbf{x}_{t-1}|c_k, \theta_k)$ then
 3:    $T \leftarrow T + 1$
 4: else if $\exp \frac{\sum_k \log P(\mathbf{x}_t|c_k,\theta_k) - \sum_k \log P(\mathbf{x}_{t-1}|c_k,\theta_k)}{Tmax/t} > Rand(t)$ then
 5:    if $t > L_1^t$ and $t < L_2^t$ then
 6:       $T = T * L_1^T$
 7:    else if $t < L_3^t$ then
 8:       $T = T * L_2^T$
 9:    else
10:       $T = T * L_3^T$
11:    end if
12: end if
13: $\alpha(t) \leftarrow \alpha(T)$
```

The annealing effects is shown in Figure 2. It can be found that the annealing schedule works well with SOM mixture recursive Bayesian model. With the help of annealing, represented by the blue curve in the lower part of Figure2, the learning process which has been "interrupted" is more sustainable and adaptable, however without annealing effects, the red curve represents a less efficient learning: the learning process doesn't improve the model performance since approximately 600 epochs. The clustering performance shown by the sum of log-likelihood in the lower figure and the estimated means (date centers) as well as the standard deviation (circles) also confirms that annealing schedule can improve the quality of learning outcome.

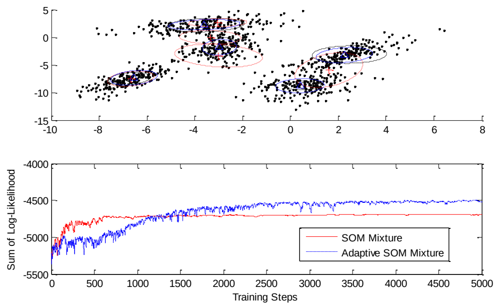

Figure 2. The figure above shows the learning outcomes of SOM mixture recursive model with and without adaptive annealing. The black dots represent original data points which consist of three groups of points and the black circles represent the centres and standard deviations of three groups. The red dot lines and crosses illustrate the outcomes of SOM mixture recursive model without proposed annealing schedule, while the blue dashed lines and triangles illustrate the outcomes of those without annealing schedule. The figure below shows the learning processes of the proposed model both with (red) and without (blue) adaptive annealing.

The parameters of annealing process is not as crucial as the learning rates in the self-organizing stage. Value of parameters would be mostly acceptable as long as the learning process can be somewhat slowing down. The annealing process is an adaptive adjuster in the proposed algorithm, so it should be purposely designed according to the nature and needs of its application.

2.5 Model Assessment

Recursive model estimation is always associated with a huge amount of computation. In order to dynamically assess the quality of those model parameters as well as optimize the learning process, we proposed to use the likelihood ratio instead of likelihood to judge whether the learning process should be terminated during the learning process. The traditional maximum likelihood estimation approach seeks the maximum of a likelihood function. In the case of recursive learning, as the data arrives randomly, so does the update of model parameters, changes of likelihood thus may not be always going towards a maximum. Consequently the maximum likelihood estimation, though generally applicable to many applications, may not be universally suitable, especially in the case when process control is needed. Likelihood ratio, on the other hand, assesses the development of likelihood and can signal when updating of parameters can not make any significant difference.

If we denote the parameter set θ^* as the estimated parameters for the less informative (or previous) model, while the learning parameter $\hat{\theta}$ denotes the estimated parameters for the informative (or updated) model, a likelihood ratio test can be constructed as follows,

$$LR = 2\log\left(\frac{L(\hat{\theta})}{L(\theta^*)}\right) = 2\left(\ell(\hat{\theta}) - \ell(\theta^*)\right) \quad (15)$$

where $L(.)$ denotes a likelihood function, LR represents the value of likelihood ratio and is believed to follow the chi-square distribution ($LR \sim \chi_p^2$), p is the degree of freedom. In our case, we aim to measure the learning efficiency by testing whether consecutive learning steps are making significant difference, that is with a large LR where θ^* and $\hat{\theta}$ represent consecutive estimated parameters.

As being shown, the value of LR shows the difference between two log-likelihood functions $\ell(\hat{\theta})$, $\ell(\theta^*)$. The magnitude of the difference depends on the "curvature" of the likelihood function $L(.)$ which is used to be defined by Jiang (2016) as a so-called information matrix,

$$I(\theta) = \frac{\dfrac{d^2\ell(\theta)}{d\theta^2}}{\left[1 + \left(\dfrac{d\ell(\theta)}{d\theta}\right)^2\right]^{3/2}} \quad (16)$$

since $\ell(\theta)$ is a function of θ with its maximum at $\ell(\theta^*)$, it has $d\ell(\theta^*)/d\theta^* = 0$. The curvature that is represented by the information matrix $I(\theta)$ can be rewritten as $I(\theta^*) = d^2\ell(\theta^*)/d\theta^{*2}$ which equals to the Hessian of $\ell(\theta)$ at θ^*. $\ell(\theta)$ with large curvature, or more information, will have relatively large LR comparing with that of with smaller curvature.

Chen (2016) argues that the profile likelihood confidence interval derives from the asymptotic Chi-squared distribution of the likelihood ratio statistic. It is believed to be more accurate than the Wald test when sample size is small. The null hypothesis $\hat{\theta} = \theta^*$ is rejected at $\alpha = 0.05$ if $LR > \chi_1^2(0.95)$ where $\chi_1^2(0.95)$ is the 0.95 quantile of a Chi-squared distribution with one degree of freedom. If the $\hat{\theta} = \theta^*$ can not be rejected, we can then conclude that update from θ^* to $\hat{\theta}$ is insignificant. It can serve as a criterion whether to stop the training / parameters updating.

In the proposed mixture model, we have more than one parameter to estimate. Thus the marginal profile confidence interval is applied, where all but one parameter has been fixed to a specified value, so that the log likelihood becomes a function of just one parameter. If suppose we would like to take a hypothesis test on μ, where the null hypothesis is $H_0 : \hat{\mu} = \mu^*$. The likelihood ratio statistic should be,

$$LR_{\mu|\Sigma} = 2\log\left(\frac{L(\hat{\mu},\Sigma^*)}{L(\mu^*,\Sigma^*)}\right) \sim \chi_1^2 \qquad (17)$$

We can reject the null hypothesis $\hat{\mu} = \mu^*$ at $\alpha = 0.05$ if $LR_{\mu|\Sigma} < \chi_1^2(0.95)$. Likewise, we then hold μ for some specific value and test whether the null hypothesis $\Sigma = \Sigma^*$ at $\alpha = 0.05$ can be rejected by checking if $LR_{\Sigma|\mu} < \chi_1^2(0.95)$.

Since the proposed training process, which incorporates the SOM learning mechanism, is not very sensitive to the number of iterations and can always converge to a local optima, we only stop the learning process when both μ and Σ can not effectively change the corresponding likelihood ratio, meanwhile, as the raw sample data is fed in one by one in the learning process, the corruptive noise will be very likely to generate misleading LR (either too large or too small) and then mistakenly interrupt the training. Therefore we need to use a compromised solution which measures an averaged LR over a certain time window and compare with a threshold value. Any averaged LR below that threshold should be the stop signal of the training process.

3 EXPERIMENT

IN order to justify the usefulness of the proposed method upon time series analysis, we apply the method on both artificial data and real data. For the artificial data, we generate both one-dimensional and two-dimensional time series data. For the real data, we collect the wine data which are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars (The data is retrievable from Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/Wine), and foreign exchange rate data (USD/EUR) as well as the Chinese stock prices data which is collected from CIS300 component stocks (blue chip stocks), finance related stocks in A share board (Chinese main board), Small and Medium-size Enterprise board and Growth Enterprise Market board (known as Chinese NASDAQ).

Firstly we create a set of "peaked" data, which is assumed to follow a multi-modal distribution. The data points are randomly generated with pre-allocated two centers (μ) and pre-specified distributions (Σ). The proposed method has been used to model the density distribution of the randomly generated data. Figure 3 depicts the learning outcome.

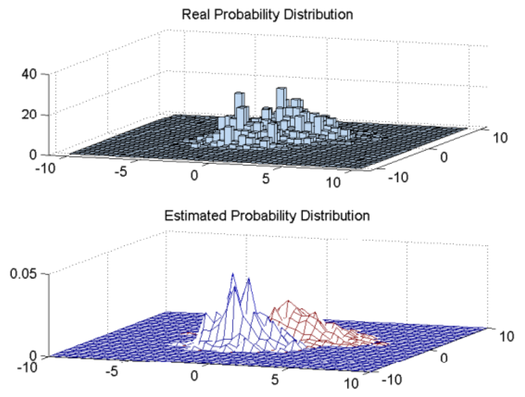

Figure 3. The figure above is a histogram which shows the original data distribution while the figure blow shows the learnt distribution by a 3-D mesh using proposed adaptive recursive self-organizing mixture map.

The estimated distribution can visually well match the density (e.g. the shapes and locations of two peaks) of generated data. In order to verify the robustness of the algorithm, we repeated independent estimation for 100 times while Figure 3 shows only one of the results. It clearly shows that on average the estimated distribution can well catch the real distribution. The generating parameters, the averaged estimated parameters as well as the bias in percentage are shown in the Table 1. It shows that quantitatively the estimation of both the locations (represented by μ) of data centers and the "shapes" (represented by Σ) of distributions are trustworthy, though some of estimated Σ are not very stable, representing by relatively large bias. Such instability is highly correlated with the level of corruptive noise. The percentage of bias though cannot tell that the estimated parameters are "significantly" equal to the original parameters, but is within a satisfying level.

Table 1. The Parameters Estimation on Figure 3

	Original		Estimated		Bias	
μ_1	1.00	2.00	1.11	2.31	12.3%	15.4%
Σ_1	3.00	0.20	2.71	0.28	22.4%	23.2%
	0.20	2.00	0.28	2.53	23.2%	27.2%
π_1	0.50		0.42		18%	
μ_2	-1.00	-2.00	-0.79	-1.67	28.0%	27.2%
Σ_2	2.00	0.00	2.39	-0.22	22.3%	21.5%
	0.00	1.00	-0.22	1.02	21.5%	2.5%
π_1	0.50		0.57		21%	

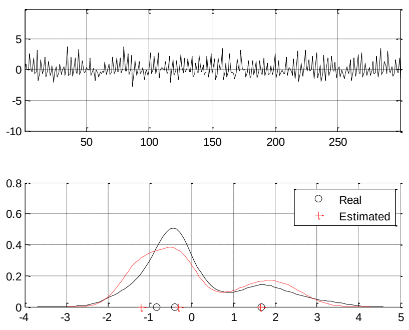

Figure 4. **The upper figure shows the artificial data generated by three pre-allocated centres as well as associated standard deviations. The lower part shows the estimated centres (red cross) and combined distribution (red dashed line) against the original distribution (black circles and black solid line).**

Figure 4 shows the capacity of the proposed algorithm on identifying the generating parameters (μ and Σ) given the "noisy" one-dimensional data. When estimating the parameters of one dimensional Gaussian models, we replace the covariance matrix by a single variance which simplifies the learning process. The estimated results shown by the red crosses and dashed line are very close to the desired black circles and solid line. Together with the comparative results in Table 2, we can generally conclude that the proposed adaptive recursive SOMM algorithm is able to outperform general EM in both estimating accuracy and computing efficiency(in terms of less number of epochs before the learning outcome reaches a satisfying level of accuracy) in the case of one dimensional sample data.

Table 2. **The Parameters Estimation on Figure 4**

	Original	Adaptive Recursive SOMM			EM		
		Est.	Bias	Epoch	Est.	Bias	Epoch
μ_1	-0.90	-1.21	29%		-1.30	34%	
Σ_1	0.20	0.17	14%		0.25	20%	
π_1	0.30	0.26	10%		0.29	8%	
μ_2	-0.30	-0.38	21%		-0.35	18%	
Σ_2	0.80	0.75	9%	420	0.87	11%	600
π_2	0.40	0.45	15%		0.47	16%	
μ_3	1.80	1.72	18%		1.69	22%	
Σ_3	1.00	1.02	4%		0.94	5%	
π_3	0.30	0.35	18%		0.38	20%	

Figure 5 shows the clustering performance of the proposed adaptive recursive SOMM on the classic wine data (These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The data can be retrieved from UCI database.). In order to visualize the clustering results more straightly, we draw four scatter plots each with 2 different attributes only. It shows that the algorithm can not only catch the peaks of data densities but also the structure of their distributions.

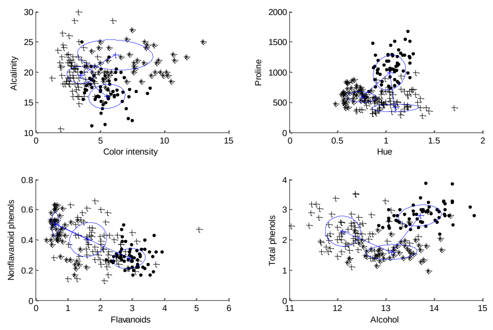

Figure 5. **Four figures show the clustering effects of the proposed adaptive recursive SOMM algorithm on 8 (out of 13) attributes of the three types of wines. We indicate the three different types of wine by cross, star and dot, and the clustering result is shown by three blue circles with their shape determined by the covariance matrix Σ.**

Figure 6 indicates that the algorithm is also able to deal with stock data with satisfying accuracy. Daily closing prices of 240 stocks, in the group of 60, are randomly collected from four different sources: Chinese Growth Enterprise Market board, Small and Medium-size Enterprise board, non-finance blue chip shares and finance related shares during two years' interval since 1 March 2013 to 1 March 2015. It can be seen that each of the four groups performs quite differently. The proposed algorithm can well catch the distribution of the stock data.

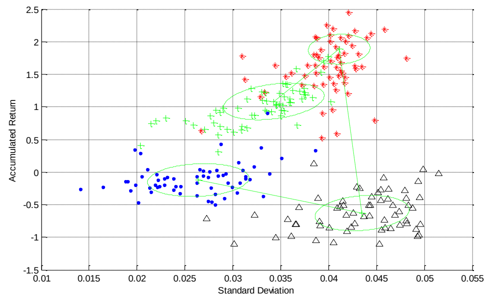

Figure 6. **Stock prices (in terms of average daily return and standard deviation from 1 March 2013 to 1 March 2015) collected from four different categories. The red stars represent stocks in Growth Enterprise Market board; the green crosses represent finance related stocks; the blue dots represent blue chip stocks; the black triangles represent stocks in Small and Medium-size Enterprise board. The proposed model can generally discover the difference as well as boundaries of those four categories.**

4 CONCLUSION AND FUTURE WORKS

THE research proposed a self-organizing adaptive Bayesian mixture model which enables traditional Gaussian Mixture Model to have a SOM style updating mechanism. The similarity measure of best matching unit of self-organizing map learning is performed by using the maximize a posterior approach. The neighborhood function of the SOM learning is consists of an additive confusion probability with a Gaussian kernel and an inherent posterior probability of winning unit and its neighbors. Such a double neighborhood structure ensures that two neighborhood functions are somewhat similar at beginning and can gradually converge during the learning process. In order to further fine tuning the algorithm, we implement a pair of operations: an annealing schedule which aims to slow down the updating speed and an early stop criterion triggered by the profile likelihood confidence interval. The algorithm is purposely designed to be theoretically efficient in identifying the grouping properties of target data. It can theoretically reach a good balance of both efficiency and accuracy. The proposed model is examined by both artificial and real market data. The clustering performance is satisfying in comparing with traditional approaches. We therefore believe the algorithm we have proposed can be a good alternative to the traditional EM approach which has been widely used for Gaussian Mixture Model parameter estimation.

The model, though has integrated several useful functions and with its better comprehensive performance over traditional approaches, is still far from prefect. We would further explore model's ability in adapting to multiple data sets especially real market financial data, which has been believed to be more noise corrupted.

Weakness of the proposed algorithm lies in its robustness in convergence. The “dual-neighbourhood” function uses both confusion probability and posterior probability. The learning process thus would be speed up and more likely to trap into local minima. Performances on different applications might be sensitive to learning parameters, which is also an inherent challenge of SOM algorithm. This is also why annealing has been used to prevent learning process being trapped into local optima. The model includes quite a few useful adjustable parameters, such as adjustable neighborhood shape, adaptive learning rate, purposely designed early stop criterion etc. The flexibility or accuracy always goes with the risk of over-fitting. Therefore, further efforts are needed in how to balance the model flexibility and complexity.

5 ACKNOWLEDGMENT

THIS work was supported by in part by Chinese National Science Council under grant (71571163, 71373236), Zhejiang Nature Science Foundation (LY20G010002, LY19G010001) and Social Sciences Foundation of the Ministry of Education (17YJA630015), Department of Education of Zhejiang Province Fund and the finance research centre of Zhejiang Gongshang University.

6 REFERENCES

Banks, H.T., Hu, S., & Joyner, M. (2017). A comparison of computational efficiencies of stochastic algorithms in terms of two infection models, Mathematical Biosciences & Engineering. 9(3): 487-526.

Byrd, R.H., Hansen, S.L., & Nocedal, J. (2016). A Stochastic Quasi-Newton Method for Large-Scale Optimization, Siam Journal on Optimization. 26(2): 1-31.

Chen, S., Li, Y.X., & Shin, J.Y. (2016). Constructing confidence intervals of extreme rainfall quantiles using Bayesian, bootstrap, and profile likelihood approaches, Science China Technological Sciences. 59(4):573-585.

Cover, T.M. & Thomas, J.A. (1991). Elements of information theory, Wiley.

Fan, W. & Machemehl, R.B. (2015). Optimal Transit Route Network Design Problem with Variable Transit Demand: Genetic Algorithm Approach, Journal of Transportation Engineering. 132(1):40-51.

Figueiredo, M. & Jain, A.K. (2002). Unsupervised learning of finite mixture models. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 24(3):381–396.

Fong, T.P. & Wong, C. (2008). Stress testing banks’ credit risk using mixture vector autoregressive models. Hong Kong Monetary Authority, Working Paper.

Heskes, T. (2001). Self-organizing maps, vector quantization, and mixture modelling, Neural Networks, IEEE Transactions on, 12(6):1299–1305, 2001.

Jacobs, R.A., Jordan, M., Nowlan, S., & Hinton, G.E. (1991). Adaptive mixtures of local experts, Neural computation.3(1):79–87.

Jiang, T. & Qi, Y. (2016). Likelihood Ratio Tests for High Dimensional Normal Distributions, Scandinavian Journal of Statistics. 42(4):988-1009.

Lindsay, B.G. (1983). The geometry of mixture likelihoods: A general theory. Annals of Statistics. 11(3):783–792.

McLachlan, G. & Peel, D. (2004). Finite mixture models. John Wiley & Sons.

Mohajer, A., Barari, M., & Zarrabi, H. (2018) Big Data based Self-Optimization Networking: A Novel Approach Beyond Cognition, Intelligent Automation and Soft Computing. 24(2): 413-421.

Moon, T.K. (1996). The expectation-maximization algorithm. Signal processing magazine IEEE. 13(6):47–60.

Mousavi, S.M. & Zandieh, M. (2018) An Efficient Hybrid Algorithm for a Bi-objectives Hybrid Flow Shop Scheduling. Intelligent Automation and Soft Computing. 24(1): 9-16.

Nguyen TM, & Wu QM. (2017). Bounded asymmetrical student's-t mixture model. IEEE Transactions on Cybernetics, 44(6), 857-869.

Rajabalinejad, M., & Mahdi, T.F. (2010). The inclusive and simplified forms of Bayesian interpolation for general and monotonic models using Gaussian and generalized beta distributions with application to Monte-Carlo simulations. Natural Hazards, 55(1), 29-49.

Rajabalinejad, M., Meester, L. E., van Gelder, P. H. A. J. M., & Vrijling, J. K. (2011). Dynamic bounds coupled with Monte Carlo simulations. Reliability Engineering & System Safety, 96(2), 278-285.

Runnalls, A.R. (2007). Kullback-leibler approach to gaussian mixture reduction. Aerospace and Electronic Systems, IEEE Transactions on, 43(3):989–999.

Seo, H., & Thorson, S. (2016). A mixture model of global internet capacity distributions. Journal of the Association for Information Science & Technology, 67(8), 2032-2044.

Teichmoeller, J. (1971). A note on the distribution of stock price changes. Journal of the American Statistical Association. 66(334):282–284.

Verbeek, J.J., Vlassis, N. & Krose, B. (2003). Efficient greedy learning of Gaussian mixture models, Neural Computation. 15(2):469–485.

Verbeek, J.J., Vlassis, N. & Krose, B. (2005). Self-organizing mixture models, Neurocomputing. 63:99–123.

Wong, C.S. & Li, W.K. (2000). On a mixture autoregressive model. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 62(1):95–115.

Xu, L. & Jordan, M. (1996). On convergence properties of the EM algorithm for Gaussian mixtures. Neural computation. 8(1):129–151.

Yin, H. (2006). On the equivalence between kernel self-organising maps and self-organising mixture density networks, Neural Networks. 19(6):780–784.

Yin, H. & Allinson, N. (2001). Self-organizing mixture networks for probability density estimation, Neural Networks, IEEE Transactions on. 12(2):405–411.

Zhuang, S., Huang, Y., Palaniappan, K. & Zhao, Y. (1996). Gaussian mixture density modeling, decomposition, and applications. IEEE Transactions on Image Processing. 5(9):1293–1302.

7 NOTE ON CONTRIBUTORS

He Ni received his PhD degree in Electrical Engineering and Electronics from the University of Manchester, England. He is a professor in School of Finance Zhejiang Gongshang University. His research interests include data mining and machine learning and has published more than 10 peer-reviewed journals including Neurocomputing, Applied Soft Computing, International Journal of Technology Management.

Yongqiao Wang received the Ph.D. degree in Management Sciences and Engineering from Academy of Mathematics and Systems Science, Chinese Academy of Sciences (CAS), Beijing in 2005. He is currently a full professor in School of Finance, Zhejiang Gongshang University. His research interests include machine learning, data mining and financial risk management. He has co-authored more than 10 papers in journals including European Journal of Operational Research, IEEE Transactions on Neural Networks and Learning Systems and IEEE Transactions on Fuzzy Systems.

Buyun Xu, an assistant professor in the Department of finance, Zhejiang Gongshang University Hangzhou College of Commerce. His main research interests includes data mining techniques in capital market applications.