Time Series Clustering with ARMA Mixtures

Yimin Xiong Dit-Yan Yeung

Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

Abstract

Clustering problems are central to many knowledge discovery and data mining tasks. However, most existing clustering methods can only work with fixed-dimensional representations of data patterns. In this paper, we study the clustering of data patterns that are represented as sequences or time series possibly of different lengths. We propose a model-based approach to this problem using mixtures of *autoregressive moving average* (ARMA) models. We derive an *expectation-maximization* (EM) algorithm for learning the mixing coefficients as well as the parameters of the component models. To address the model selection problem, we use the Bayesian information criterion (BIC) to determine the number of clusters in the data. Experiments are conducted on a number of simulated and real datasets. Results from the experiments show that our method compares favorably with other methods proposed previously by others for similar time series clustering tasks.

Key words: ARMA model, EM algorithm, Mixture model, Model-based clustering, Time series analysis

1 Introduction

Clustering is the unsupervised process of grouping data patterns into clusters so that patterns within a cluster bear strong similarity to one another but are very dissimilar to patterns in other clusters. Clustering problems are central to many knowledge discovery and data mining tasks. While many clustering techniques have been studied by researchers in statistics, pattern recognition,

and machine learning, most of these techniques are based on the assumption that data patterns can be represented as points in multidimensional spaces of fixed dimensionality. Unfortunately this assumption does not always hold. Temporal patterns involving sequences or time series are one important class of such problems that are found in many applications from scientific, medical, sociological, financial and other domains.

Distance-based methods (e.g., [1]) and *model-based* methods (e.g., [2]) are two major classes of clustering methods. They are analogous to other nonparametric and parametric methods, respectively, in that the former category (i.e., distance-based or nonparametric methods) assumes only some weak structure of the data, but the latter category (i.e., model-based or parametric methods) assumes some strong structure. *Partitional* distance-based methods, such as the well-known k-means algorithm, usually require the number of clusters to be known *a priori*. *Hierarchical* distance-based methods, on the other hand, do not require the number of clusters to be known. However, they have to search exhaustively for the number of clusters, either by starting from one (for *divisive* methods) or from the total number of all data patterns available (for *agglomerative* methods). Moreover, since typically pairwise distances have to be computed, hierarchical distance-based methods tend to have computational complexity that is quadratic in the number of data patterns and hence becomes prohibitive for large data sets.

Unlike distance-based methods, however, model-based methods can incorporate prior knowledge more naturally in finding the correct number of clusters. Also, for time series data, they provide a principled approach for handling the problem of modeling and clustering time series of different lengths. In this paper, we focus on model-based time series clustering methods. In particular, *mixture models* [2] are used.

The remainder of this paper is organized as follows. Some related work on model-based clustering of time series is reviewed in Section 2. In Section 3, we present our model-based time series clustering method based on mixtures of time series models. The learning algorithm is outlined in Section 4, with details left to Appendix A. We address the model selection problem in Sec-

tion 5. Experimental results for simulated and real datasets are presented in Sections 6 and 7, respectively. Finally, Section 8 concludes the paper with discussions of some possible future work.

2 Related Work

2.1 Markov Chains

Finite mixtures of Markov chains [3–8] have been proposed for clustering time series. The *expectation-maximization* (EM) algorithm [9] is used to learn the mixing coefficients as well as the parameters of the component models. The number of clusters can be determined by comparing different choices of the number based on some scoring scheme. One possibility, used by Cadez *et al.* [7,8], is related to minimizing the description length.

Another approach to the clustering of time series modeled by Markov chains is called Bayesian clustering by dynamics (BCD) [10–12]. Strictly speaking, this method is not a purely model-based approach. Rather, it can best be seen as a hybrid approach with both model-based and distance-based flavors. The BCD method first transforms each time series into a Markov chain, with its dynamics represented simply by a transition probability matrix. Next, it goes through an agglomerative procedure by trying to merge the two closest Markov chains at each step, using the Kullback-Leibler divergence [13] as the dissimilarity (cf. distance) measure between transition probability matrices. Based on a greedy heuristic search approach, this procedure continues until the resulting model is found to be less probable than the model before merging. Thus the number of clusters can be determined automatically.

2.2 Hidden Markov Models

While simple Markov chains are good enough for some applications, some time series can be modeled better using *hidden Markov models* (HMM) [14]

due to their ability of handling uncertainty in temporal and spatial dimensions simultaneously. For example, HMMs have been very successfully used for speech recognition, handwriting recognition, and bioinformatics. The idea of HMM-based clustering was first studied by Rabiner *et al.* [15] for speech recognition applications. Multiple HMMs are created through a splitting procedure to group speech data into clusters for more accurate modeling. The HMM-based clustering method studied in [16] follows the same spirit in that HMMs are introduced one at a time to model the time series data more accurately. It also shows that this method can perform better if dynamic time warping (DTW) is used as a distance metric to form an initial partitioning of the time series for the subsequent HMM-based clustering procedure.

Finite mixtures of HMMs have also been studied by a number of researchers. Similar to mixtures of Markov chains, the EM algorithm can also be used for HMM mixtures [17–20]. To trade accuracy for efficiency, the k-means algorithm (used in [21]) and the rival penalized competitive learning (RPCL) algorithm (used in [22]) have also been used in place of EM. The number of clusters can be determined using Monte-Carlo cross-validation [19] or information criteria such as the Bayesian information criterion (BIC) [20].

2.3 Regression Models

Regression models, *mixtures of regression models* or *regression mixtures*, and their extensions [23–25] are another type of models that can be used for time series modeling and clustering. Typically, a regression model provides a projection from the baseline status to some relevant demographic variables. Curve-type time series data are quite common examples of these kinds of variables. For example, mixtures of standard regression models and the accompanying EM algorithm have been used by Gaffney and Smyth [24] for the clustering of trajectory data. Recently in [25], they combined linear random effects models with standard regression mixtures and extended mixtures of regressions to the so-called random effects regression mixtures to more effectively solve similar clustering problems. However, the problem of automatically determining the number of clusters or the model size was not studied by them.

2.4 Autoregressive Moving Average Models

In addition to Markov chains, HMMs and regression models, *autoregressive moving average* (ARMA) and *autoregressive integrated moving average* (ARIMA) models have also been used extensively for time series analysis [26,27]. Kwok *et al.* [28] applied mixtures of ARMA models as well as their special cases, mixtures of autoregressive (AR) models, for time series modeling and forecasting. However, clustering applications based on such mixture models were not studied by them.

Similar to the BCD method for Markov chains, Ramoni *et al.* [29] extended their BCD method for mixtures of AR models. They also introduced a goodness of fit measure for the resulting clustering model to evaluate the goodness of the fitted model. Their method hierarchically searches for the AR models with the highest marginal likelihood values, and selects the model order with the highest goodness of fit score.

Recently, a method was proposed by Kalpakis *et al.* [30] for clustering ARIMA time series. This method is similar to the BCD method for Markov chains in that it is a hybrid method with both model-based and distance-based characteristics. For each time series in the data set, a differencing operation is applied to remove the nonstationarity in the time series and a separate ARMA model is created by estimating the model parameters from the time series after nonstationarity removal. Afterwards, a partitional distance-based clustering method called the partitioning around medoids (PAM) method is applied to group the ARMA models into a prespecified number of clusters. The distance measure used is the Euclidean distance between the linear predictive coding (LPC) cepstral coefficients computed from two ARMA models.

In the next section, we propose a new time series clustering method based on mixtures of ARMA models [31].

3 Model-Based Clustering with ARMA Mixtures

3.1 Standard ARMA Models

The ARIMA model introduced by Box and Jenkins [26] is a combination of three types of time series data processes, namely, autoregressive, integrated, and moving average processes. A stationary ARIMA model with autoregressive order p and moving average order q is commonly denoted as ARMA(p, q). If the ARMA(p, q) model is used on time series data integrated to order d, then the model for the original time series is denoted as ARIMA(p, d, q). Given a time series $\mathbf{x} = \{x_t\}_{t=1}^n$, the fitted ARMA$(p, q)$ model takes the form

$$x_t = \phi_0 + \sum_{j=1}^{p} \phi_j x_{t-j} + \sum_{j=1}^{q} \theta_j e_{t-j} + e_t, \qquad t = 1, 2, \ldots, n,$$

where n is the length of the time series, ϕ_0 is a constant term, $\{\phi_1, \phi_2, \ldots, \phi_p, \theta_1, \theta_2, \ldots, \theta_q\}$ is the set of AR(p) and MA(q) coefficients, and $\{e_t\}_{t=1}^n$ is a sequence of independent and identically distributed (i.i.d.) Gaussian white noise terms with variance σ^2. From [27], we can express the natural logarithm of the conditional likelihood function as

$$\ln P(\mathbf{x}|\mathbf{\Phi}) = -\frac{n}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{t=1}^{n} e_t^2,$$

where $\mathbf{\Phi} = \{\phi_0, \phi_1, \phi_2, \ldots, \phi_p, \theta_1, \theta_2, \ldots, \theta_q, \sigma^2\}$ is the set of all model parameters and e_t must be estimated recursively, i.e.,

$$e_t = x_t - \phi_0 - \sum_{j=1}^{p} \phi_j x_{t-j} - \sum_{j=1}^{q} \theta_j e_{t-j}, \qquad t = 1, 2, \ldots, n.$$

3.2 ARMA Mixtures

We now extend standard ARMA models to mixtures of ARMA models, or simply called ARMA mixtures, for time series clustering. Let us assume that the time series data are generated by M different ARMA models, which correspond to the M clusters of interest denoted as $\omega_1, \omega_2, \ldots, \omega_M$. Let $P(\mathbf{x}|\omega_k, \mathbf{\Phi}_k)$ denote the conditional likelihood function or density function of component

model k, with $\mathbf{\Phi}_k$ being the set of parameters for the model. Let $P(\omega_k)$ be the prior probability that a time series comes from model k. The conditional likelihood function of the mixture model can be expressed in the form of a mixture density as

$$P(\mathbf{x}|\mathbf{\Theta}) = \sum_{k=1}^{M} P(\mathbf{x}|\omega_k, \mathbf{\Phi}_k)P(\omega_k),$$

where $\mathbf{\Theta} = \Big\{\mathbf{\Phi}_1, \mathbf{\Phi}_2, \ldots, \mathbf{\Phi}_M, P(\omega_1), P(\omega_2), \ldots, P(\omega_M)\Big\}$ represents the set of all model parameters for the mixture model. For a time series $\mathbf{x}$, it is assigned to cluster ω_k with posterior probability $P(\omega_k|\mathbf{x})$, where $\sum_{k=1}^{M} P(\omega_k|\mathbf{x}) = 1$.

Suppose we are given a set $\mathbf{D} = \Big\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\Big\}$ of N time series. Under the usual assumption that different time series are conditionally independent given the underlying model parameters, we can express the likelihood of $\mathbf{D}$ as

$$P(\mathbf{D}|\mathbf{\Theta}) = \prod_{i=1}^{N} P(\mathbf{x}_i|\mathbf{\Theta}) \tag{1}$$

Model parameter learning amounts to finding the *maximum a posteriori* (MAP) parameter estimate given the data set $\mathbf{D}$, i.e.,

$$\widehat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta}} \Big[P(\mathbf{D}|\mathbf{\Theta})P(\mathbf{\Theta})\Big].$$

If we take a noninformative prior on $\mathbf{\Theta}$, learning degenerates to *maximum likelihood estimation* (MLE), i.e.,

$$\widehat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta}} P(\mathbf{D}|\mathbf{\Theta}).$$

This MLE problem can be solved efficiently using the EM algorithm, which is discussed in detail in the next section.

4 EM Learning Algorithm

4.1 Standard EM Algorithm

The EM algorithm is an iterative approach to MLE or MAP estimation problems in the presence of incomplete data. It has been widely used for many

applications, including clustering and mixture density estimation problems [32].

Let us rewrite the likelihood in Equation (1) as a function of the parameter vector $\mathbf{\Theta}$ for a given data set $\mathbf{D}$:

$$L(\mathbf{\Theta}; \mathbf{D}) = P(\mathbf{D}|\mathbf{\Theta}) = \prod_{i=1}^{N} P(\mathbf{x}_i|\mathbf{\Theta}).$$

Assuming a noninformative prior on $\mathbf{\Theta}$, the goal of the EM algorithm is to find $\mathbf{\Theta}$ that maximizes the likelihood $L(\mathbf{\Theta}; \mathbf{D})$ or the log-likelihood

$$\ell(\mathbf{\Theta}; \mathbf{D}) = \sum_{i=1}^{N} \ln P(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{i=1}^{N} \ln \left[\sum_{k=1}^{M} P(\mathbf{x}_i|\omega_k, \mathbf{\Phi}_k) P(\omega_k) \right].$$

Since $\mathbf{D}$ is the incomplete data, we assume the missing data to be $\mathbf{Z} = \Big\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\Big\}$, such that $\mathbf{D}$ and $\mathbf{Z}$ form the complete data $(\mathbf{D}, \mathbf{Z})$. Thus the complete-data log-likelihood function is $\ln P(\mathbf{D}, \mathbf{Z}|\mathbf{\Theta})$. If we know the missing data (and hence the complete data), parameter estimation would be straightforward. Without knowing the missing data, however, the EM algorithm has to iterate between the Expectation step (E-step) and the Maximization step (M-step). In the E-step, we calculate the expected value $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ of the complete-data log-likelihood with respect to the unknown data $\mathbf{Z}$ given the observed data $\mathbf{D}$ and the current parameter estimate $\mathbf{\Theta}(t)$, i.e.,

$$Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big) = E\Big[\ln P(\mathbf{D}, \mathbf{Z}|\mathbf{\Theta}) \,|\, \mathbf{D}, \mathbf{\Theta}(t)\Big].$$

In the M-step, we try to maximize $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ with respect to $\mathbf{\Theta}$ to find the new parameter estimate $\mathbf{\Theta}(t+1)$. If the M-step is carried out to increase $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ only but there is no guarantee that $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ is maximized, this generalized version of the EM algorithm is often called a generalized EM (GEM) algorithm.

4.2 EM Algorithm for ARMA Mixtures

In the context of using ARMA mixtures for clustering, the missing data correspond to the unknown cluster or group membership of each time series $\mathbf{x}_i$.

The log-likelihood $\ell(\mathbf{\Theta}; \mathbf{D})$ can thus be expressed as

$$\ell(\mathbf{\Theta}; \mathbf{D}) = \sum_{i=1}^{N} \ln P(\mathbf{x}_i|\omega_{\mathbf{z}_i}, \mathbf{\Phi}_{\mathbf{z}_i}) + \sum_{i=1}^{N} \ln P(\omega_{\mathbf{z}_i}).$$

Given the observed data $\mathbf{D}$ and the current parameter estimate $\mathbf{\Theta}(t)$, the expectation of the complete-data log-likelihood becomes

$$Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big) = \sum_{i=1}^{N}\sum_{k=1}^{M} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big) \ln P(\mathbf{x}_i|\omega_k, \mathbf{\Phi}_k) + \sum_{i=1}^{N}\sum_{k=1}^{M} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big) \ln P(\omega_k), \tag{2}$$

where the posterior probabilities $P(\omega_k|\mathbf{x}_i, \mathbf{\Theta})$ can be computed using the Bayes rule as

$$P(\omega_k|\mathbf{x}_i, \mathbf{\Theta}) = \frac{P(\mathbf{x}_i|\omega_k, \mathbf{\Phi}_k)P(\omega_k)}{\sum_{u=1}^{M} P(\mathbf{x}_i|\omega_u, \mathbf{\Phi}_u)P(\omega_u)}, \qquad i = 1, 2, \ldots, N \text{ and } k = 1, 2, \ldots, M. \tag{3}$$

The EM algorithm iteratively maximizes the function $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ until convergence. For each iteration, we compute the posterior probabilities $P(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t))$ and $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ using the current parameter estimate $\mathbf{\Theta}(t)$ in the E-step, and update the parameter estimate by maximizing $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ with respect to $\mathbf{\Theta}$ to obtain $\mathbf{\Theta}(t{+}1)$ in the M-step. The algorithm can be summarized as follows:

initialize $\mathbf{\Theta}$, $t \leftarrow 0$
  do $t \leftarrow t + 1$
    E-step: Compute posterior probabilities $P(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t))$ and $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ using the current parameter estimate and Equation (3)
    M-step: $\mathbf{\Theta}(t{+}1) \leftarrow \arg\max_{\mathbf{\Theta}} Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ using Equations (A.2), (A.3) and (A.7)
  until some convergence condition is satisfied
return $\mathbf{\Theta}(t{+}1)$

One possible convergence condition is that the difference in log-likelihood between two time steps, i.e., $\ell(\mathbf{\Theta}(t{+}1); \mathbf{D}) - \ell(\mathbf{\Theta}(t); \mathbf{D})$, is less than some prespecified threshold. Another possibility, which is used in our implementation, is to detect if the change from $\mathbf{\Theta}(t)$ to $\mathbf{\Theta}(t{+}1)$ is significantly large. Detailed derivation of the corresponding M-step equations can be found in Appendix A.

4.3 Reducing the Dependence on Parameter Initialization

Since the standard EM algorithm can only guarantee finding a local maximum, the quality of the clustering result depends heavily on the initial parameter values. We have introduced two methods to reduce the dependence on parameter initialization.

The first method is based on the so-called stochastic EM (SEM) algorithm [33,34], which is a stochastic variant of the standard EM algorithm. It adds a stochastic step (S-step) between the E-step and M-step of standard EM. This S-step generates a partition of the time series data by assigning each time series to a cluster randomly according to the posterior probabilities estimated. The M-step then uses this partition to update the parameter estimates. The randomness associated with the S-step increases the chance that the algorithm converges to a good set of model parameters. In our experiments, we use the solution obtained by the significantly faster SEM algorithm to initialize the model for the subsequent standard EM algorithm.

Although SEM reduces the effect due to parameter initialization, it cannot solve the problem completely. In the second method, we first incorporate the noise variance with the ARMA coefficients estimated from each time series as feature vectors and feed them to the distance-based k-means clustering algorithm to get a specified number of centers of the parameter values. After that we use these centers as the initial parameter values of the component models to run the standard EM algorithm. The k-means algorithm is run several times with different starting points of the feature vectors to get different random initial parameter values. Among the results of the standard EM algorithm based on these initial parameter values, we finally pick the one with the maximum log-likelihood $\ell(\mathbf{\Theta}; \mathbf{D})$.

5 Model Selection

The learning problem discussed in the previous section assumes that the model has already been selected, i.e., the number of clusters has already been spec-

ified by the user before clustering is performed. However, in many real-world problems, the actual model size is unknown. We have to select the most appropriate model (size) for clustering problems. In this section, we address the model selection problem. Two families of model selection methods in common use are *cross validation* [35–37] and *Bayesian model selection* (e.g., [38,39]). The Bayesian approach to model selection is to compute the posterior model probabilities of all possible models in the model space, and to select the model with the highest posterior probability. There are several criterion-based approaches to model selection that have a Bayesian motivation, among which the Bayesian information criterion (BIC) [40] is a commonly used one.

The BIC, which is an approximation of the Bayes factor, is based on the maximized log-likelihood minus a penalty term to estimate the posterior model probability quickly and efficiently. The BIC takes the form

$$\mathrm{BIC} = \ln P(\mathbf{D}|\hat{\boldsymbol{\Theta}}) - \frac{1}{2} V \ln N,$$

where $\hat{\boldsymbol{\Theta}}$ is the MLE parameter of the model and V is the number of independent parameters to be estimated in the model. The first term is the maximized log-likelihood which tends to favor larger models with more parameters, while the second is the penalty term which favors smaller models with less parameters. The BIC criterion tries to strike a balance between the simplicity of a model and its fit to the data. The larger the BIC value, the better the model.

Under our framework, the best mixture model for clustering has the maximum marginal likelihood probability $P(\mathbf{D}|\boldsymbol{\Theta})$. BIC approximation of the marginal likelihood can be used as a model selection criterion to search for the correct number of component models for a mixture model.

Given a partition with M clusters, the BIC criterion is expressed as

$$\mathrm{BIC} = \sum_{i=1}^{N} \ln \left[\sum_{k=1}^{M} P(\mathbf{x}_i|\omega_k, \hat{\boldsymbol{\Phi}}_k) \hat{P}(\omega_k) \right] - \frac{1}{2} \Big(M\nu + M - 1 \Big) \ln N,$$

where ν is the number of parameters in each component model.

Given a set of time series, we first specify the maximum number of component models that we want to try. We run the EM algorithm several times with an

increasing number of component models, ranging from two to the maximum number specified. For each number of components, multiple trials are run with different random initializations. The trial that gives the highest BIC value is chosen to be the solution for that number of components. The number of clusters can then be determined by comparing the best trials for different mixture model sizes to find the one with the highest BIC value.

6 Experimental Results for Simulated Datasets

As in [30], experiments are conducted on both simulated and real datasets. Instead of handling ARIMA time series directly, a preprocessing step of differencing is first applied to convert each nonstationary ARIMA time series into the corresponding stationary ARMA time series. Moreover, as discussed in [27], ARMA models can be converted into equivalent AR models. [1] Thus, for simplicity, we in fact use mixtures of AR models in most of our experiments. Experiments with ARMA mixtures are also given at the end of this section to demonstrate that the EM algorithm presented above can be used for general ARMA mixtures.

We have implemented our method in MATLAB. In addition, we have also implemented the hybrid method proposed by Kalpakis *et al.* [30]. The AR coefficients of each time series in a dataset are estimated using the forward-backward method [41] provided by the System Identification Toolbox in MATLAB. The LPC cepstral coefficients are then computed based on the estimated AR coefficients. The Euclidean distance between the first eight LPC cepstral coefficients of two time series is used as the distance measure for the PAM partitional distance-based clustering method to partition the time series into clusters.

To evaluate and compare the clustering results obtained by Kalpakis *et al.*'s method (abbreviated in the tables below as CEP for cepstral coefficients) and

[1] Theoretically speaking, finite-order ARMA models are equivalent to infinite-order AR models. In practice, however, there usually exist finite-order AR models that are good enough for approximation.

our method (abbreviated as MAR for mixtures of AR models), we use the cluster similarity measure in [42], which can be defined as

$$Sim(G, A) = \frac{1}{M} \sum_{i=1}^{M} \max_{1 \le j \le M} Sim(G_i, A_j),$$

where $G = G_1, G_2, \ldots, G_M$ is the clustering for the ground truth, $A = A_1, A_2, \ldots, A_M$ is that obtained by a clustering method under evaluation, and

$$Sim(G_i, A_j) = \frac{2\,|G_i \cap A_j|}{|G_i| + |A_j|}.$$

Apparently, the similarity measure has values ranging from 0 to 1, with 1 corresponding to the case when G and A are identical. Note that this similarity measure is asymmetric.

6.1 Experiments with Known Number of Clusters

We first study the simpler scenario with simulated time series data generated by a known number of AR models. We consider two cases separately. The first case involves AR models with the same noise variance, and the second case involves AR models with different noise variances.

6.1.1 Time Series with Same Noise Variance

In this experiment, we use two zero constant term AR(1) models with their AR coefficients uniformly distributed in the ranges (0.30 ± 0.01) and (0.60 ± 0.01), respectively. The noise variance is 0.01 for both models. Each model generated 15 time series to form the dataset. As expected, both our MAR method and the CEP method work very well because the two groups of time series are easily separable. The cluster similarity measure is always equal to 1.

We further conduct more experiments on time series generated by two closer AR(1) models. As before, the AR coefficient of one model is uniformly distributed in the range (0.30 ± 0.01), but that for the other model is set to four different ranges in four different experiments, varying from (0.55 ± 0.01) to (0.40 ± 0.01). In each experiment, each model generated 15 time series to form

the dataset. Both methods are run 10 times on each dataset. The minimum, average, and maximum values of the cluster similarity measure over 10 trials have been recorded. Table A.1 summarizes the results obtained by the two methods. Our method is slightly better than CEP when the two AR(1) models are farther from each other, but CEP becomes slightly better when the range of AR coefficient of one model decreases to (0.40 ± 0.01), which is very close to that of the other model.

6.1.2 Time Series with Different Noise Variances

We repeat the experiments above under the same setup, except that the two zero constant term AR(1) models have the same AR coefficient distribution range of (0.30 ± 0.01) but different noise variances of 0.01 and 0.02, respectively. Table A.2 shows the results obtained by the two methods. Our method gives perfect clustering of the two groups of time series, but CEP, which makes no use of the noise variances, gives very poor results on this dataset.

This set of experiments based on simulated datasets allows us to explore the strengths and weaknesses of the two methods under different controlled settings. While our method, like other EM-based methods, generally degrades in clustering performance when the underlying clusters are very close to each other, it is better than Kalpakis *et al.*'s distance-based method under more general situations. Specifically, our method is significantly better when the models have different noise variances. It is also more flexible in determining the underlying number of clusters automatically, as we will show in the next subsection.

6.2 Experiments with Unknown Number of Clusters

We now consider the more general scenario in which the number of underlying clusters is unknown.

We have tested three different groups of time series datasets generated by three, four, and six zero constant term AR(1) component models, respectively.

The parameter distribution ranges of these component models are shown in Table A.3.

Experimental results (not shown here to save space) show that when the number of component models specified is equal to or less than the actual number of clusters, the basic EM algorithm converges fast. However, if the number of component models specified is larger than the actual number of clusters, the EM algorithm converges even within a reasonably large number of iterations. Moreover, some component models learn to become very similar to each other. The posterior probability values of some data patterns associated with these component models are also much nearer than others. These characteristics can help us to decide whether too many component models are specified and hence determine the maximum number of component models to try.

BIC values are used to select the number of clusters as well as the best local maximum among different solutions corresponding to different initializations for the same number of components. For each number of component models, multiple trials of the experiment are performed with different random initializations. The number of component models ranges from two to some maximum value chosen (seven, eight, and ten, respectively, for the three groups of datasets). Along with each round which includes one trial for each number of component models tried, we maintain a sequence of cumulative highest BIC values. To avoid having to specify the maximum number of component models to try, one could use an alternative method by stopping to try larger mixture models as soon as the sequence of BIC values decreases.

Tables A.4, A.5, and A.6 show the first five sequences of the cumulative highest BIC values corresponding to typical clustering results for different specified numbers of component models. The cluster similarity values obtained for the three datasets are 1.00, 0.98, and 1.00, respectively.

For each of the 3-cluster, 4-cluster and 6-cluster cases, we randomly generate 12 different datasets for our experiment. The number of trials required for the program to find the correct number of clusters for each dataset can be found in Table A.7. For most cases (33 out of 36 cases), our program is able to find the correct number of clusters within five trials. Moreover, the BIC value is

effective in helping to determine the correct number of clusters for all cases.

6.3 Experiments with Different AR Orders

We have tested four different groups of time series datasets generated by four different sets of AR models, each of which contains three different AR models. The parameter specification of these AR models is shown in Table A.8. As we can see, all AR generating models in the second experiment have the same noise variance, while all models in the third experiment have similar process mean. [2] Except for the fourth experiment, all experiments use AR models of the same order.

The number of time series generated by each model is the same in the balanced cases (10 time series from each generating model) but it varies in the unbalanced cases (5, 10, and 15 time series, respectively). We run the clustering algorithm on each dataset with different AR orders.

Table A.9 shows the experimental results. For comparison with the results in [29], we use the average cluster impurity rate here to evaluate the clustering performance. The average cluster impurity rate is simply the ratio between the total number of time series assigned to the wrong group, and the total number of time series in the dataset. The results using different AR orders are similar especially when the time series are long. Note that our results are either the same or even better than those of [29] for most of the datasets generated by similar AR models.

6.4 Experiments with ARMA Models

To demonstrate that our method is applicable to general ARMA mixtures, we conduct two experiments with ARMA models in this subsection. Both experiments involve three ARMA(2,1) models with zero constant terms. As shown in Table A.10, although the ARMA models from the two experiments

[2] The process mean of an AR model can be computed as $\phi_0/\left(1-\sum_{j=1}^{p}\phi_j\right)$.

have the same autoregressive and moving average coefficients, the models in the first experiment have different noise variances.

Ten time series are generated for each component model. We randomly generate 10 datasets for each experiment and run our program on each dataset with the correct number of clusters. Experimental results (see Table A.11) show that our method can correctly group all time series into clusters for each dataset of the first experiment. Even for datasets of the second experiment, where time series are generated by ARMA models with the same noise variance, our results are quite good.

7 Experimental Results for Real Datasets

For comparison, we conduct further experiments with the same four real datasets used by Kalpakis *et al.* [30] and the EEG datasets from the UCI KDD Archive. The same preprocessing steps used by them are also applied to the datasets to remove the nonstationarity in the data.

7.1 First Four Datasets

7.1.1 Personal Income Dataset

The personal income dataset consists of 25 time series representing the per capita personal income from 1929–1999 in 25 states of the USA. [3] The 25 states are divided into two groups based on their personal income growth rate. The east coast states, CA and IL form the first group with a high growth rate, while the mid-west states form the second group with a low growth rate.

Kalpakis *et al.* decided that this dataset could be modeled by ARIMA$(1, 1, 0)$ models. Thus AR(1) coefficients were extracted to compute the LPC cepstral coefficients. For our method, we use a mixture of two AR(1) models.

[3] `http://www.bea.gov/bea/regional/spi/`

The results for the preprocessed time series using both methods are shown in Table A.12. Our method performs better than the CEP method.

7.1.2 ECG Dataset

The ECG dataset was obtained from the ECG database at PhysioNet. [4] It consists of two groups of time series. The first group contains 22 time series representing the 2 second ECG recordings of people having malignant ventricular arrhythmia, and the second group contains 13 time series of the 2 second ECG recordings of healthy people.

For the CEP method, the time series was assumed to be from ARIMA$(2, 3, 0)$ models. So we use a mixture of two AR(2) models for our method. Table A.12 shows the results. The two methods give the same result for this dataset.

7.1.3 Temperature Dataset

The temperature dataset is a collection of time series of the daily temperature in the year 2000 from various places in Florida, Tennessee and Cuba. [5] The dataset we use includes temperature recordings from 9 locations in Tennessee, 4 locations in northern Florida, 8 locations in southern Florida, and 8 locations in Cuba. Based on geographical proximity and similarity in temperature, time series from Tennessee and northern Florida form the first group and those from southern Florida and Cuba form the second group. The CEP method assumed ARIMA$(2, 1, 0)$ models and our method uses a mixture of two AR(2) models. The clustering results for both methods are shown in Table A.12. Since the variances of the two groups of time series are quite different, our method yields good results.

[4] `http://www.physionet.org/physiobank/database/`
[5] `http://lwf.ncdc.noaa.gov/oa/climate/climatedata.html`

7.1.4 Population Dataset

The population dataset is a collection of 20 time series representing the population estimates from 1900–1999 in 20 states of the USA. [6] The 11 time series in the first group have an exponentially increasing trend and the remaining time series in the second group have a stabilizing trend. The CEP method assumed ARIMA$(1, 1, 0)$ models and our method uses a mixture of two AR(1) models. Table A.12 summarizes the results. Although our method gives the same result as the CEP method, both are not considered satisfactory in clustering the time series into two groups. The major reason for the unsatisfactory clustering performance is that both methods actually work on stationary time series after the nonstationarity is removed. However, for this dataset, it is the exact nonstationary trend of the time series that is the most discriminating feature.

7.2 Discussions

Compared with the CEP method, our method can give the same (for two datasets) or even better (for another two datasets) results. However, both our method and the CEP method, due to their nature of modeling stationary ARMA processes only, do not learn the differences in trends of the time series and hence cannot give very satisfactory results for the population dataset. It should be noted, however, that the trends of the two groups of population time series are actually visually distinguishable. An extension of our method to address this issue is discussed in the next section.

7.3 EEG Datasets

7.3.1 Experiments with Univariate ARMA Mixtures

To evaluate the effectiveness of our method on time series from real-world applications, we further conduct some experiments with EEG data. EEG time

[6] `http://eire.census.gov/popest/archives/state/st_stts.php`

series have been widely used in the area of human-computer interaction for the study of underlying human brain processes. We use a subset of the EEG dataset from the UCI KDD Archive [7] for our experiments. The complete EEG dataset arose from a large study to examine EEG correlates of genetic predisposition to alcoholism. There are two kinds of subjects in the data: control subjects and alcoholic subjects. Multi-channel time series were recorded for these two kinds of subjects. The whole dataset includes 122 subjects, with each subject completed 120 trials where different stimuli were shown. The small dataset provided contains data for two subjects only, with 10 trials for each of three matching paradigms, named Task 1, Task 2, and Task 3, respectively. We include in our datasets time series from two channels (F4, P8) for each trial of the two subjects under the three tasks, i.e., we form a total of six datasets on different channels and under different tasks. We perform clustering on these six datasets with the goal of separating time series of different subjects.

Following previous results by Keirn and Aunon [43], AR(6) models are used in all our experiments to represent the EEG time series. A differencing step is first applied to the time series as the preprocessing step to remove the nonstationarity in trend. We follow the same BIC selection procedure as in the previous section on the six datasets. The BIC criterion correctly selected two clusters for the datasets of both channels of Task 2 and Task 3, but it selected three clusters for those of both channels of Task 1, with most of the time series from the alcoholic subject separated into two different clusters. The cluster similarity values for the 2-cluster results of each dataset are presented in Table A.13. Our results are quite promising and even slightly better than those obtained by Zhong *et al.* [44], who performed HMM-based classification on similar datasets by utilizing class membership information of the training data.

[7] `http://kdd.ics.uci.edu/`

7.3.2 First Attempt at Multivariate ARMA Mixtures

We further extend our method to simple multivariate models. We make use of both channels of the EEG time series, but ignore the correlation between time series values from different channels, i.e., the ARMA coefficient matrices and noise covariance matrices of the simple multivariate models are all diagonal matrices. The conditional log-likelihood function for the multivariate time series becomes the sum of the log-likelihood functions for the univariate time series from different channels. As shown in Table A.14, the results for the simple multivariate models are better than those of the univariate models.

8 Conclusion and Future Work

In this paper, we have proposed a model-based method for clustering univariate and simple multivariate ARMA time series. This mixture-model method, based on mixtures of ARMA models, uses an EM algorithm to learn the mixing coefficients as well as the parameters of the component models. In addition, we use the BIC criterion to determine the number of clusters in the data. Experimental results show the effectiveness of this method.

Our method can be improved in a number of aspects. One natural extension is to allow general multivariate ARMA component models. This extension is currently being pursued. Also, computational speedup can be achieved by pruning away some models if their posterior probabilities become very close to 0, indicating that their significance is negligible. The issue to address here is to decide the appropriate time to apply this pruning operation without making premature decisions.

One problem with our method, like other EM-based methods, is that its clustering performance can degrade significantly when the underlying clusters are very close to each other. Another problem is that it is not designed for modeling the differences in trend of the time series. A possible extension is to model ARIMA time series directly without having to remove the nonstationarity in trend as a preprocessing step.

References

[1] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, NJ, USA, 1988.

[2] G.J. McLachlan and K.E. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York, NY, USA, 1988.

[3] C.S. Poulsen. Mixed Markov and latent Markov modelling applied to brand choice behaviour. *International Journal of Research in Marketing*, 7(1):5–19, 1990.

[4] G. Ridgeway. Finite discrete Markov process clustering. Technical Report MSR-TR-97-24, Microsoft Research, Redmond, WA, USA, September 1997.

[5] P. Smyth. Probabilistic model-based clustering of multivariate and sequential data. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*, pages 299–304, Fort Lauderdale, FL, USA, 4–6 January 1999.

[6] I.V. Cadez, S. Gaffney, and P. Smyth. A general probabilistic framework for clustering individuals and objects. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 140–149, Boston, MA, USA, 20–23 August 2000.

[7] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Model-based clustering and visualization of navigation patterns on a web site. Technical Report MSR-TR-00-18, Microsoft Research, Redmond, WA, USA, March 2000. Revised September 2001.

[8] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 280–284, Boston, MA, USA, 20–23 August 2000.

[9] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[10] P. Sebastiani, M. Ramoni, P. Cohen, J. Warwick, and J. Davis. Discovering dynamics using Bayesian clustering. In *Proceedings of the Third International*

Symposium on Intelligent Data Analysis, pages 199–209, Amsterdam, Netherlands, 9–11 August 1999.

[11] M. Ramoni, P. Sebastiani, and P. Cohen. Multivariate clustering by dynamics. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 633–638, Austin, TX, USA, 30 July 30 - 3 August 2000.

[12] M. Ramoni, P. Sebastiani, and P. Cohen. Bayesian clustering by dynamics. *Machine Learning*, 47(1):91–121, 2002.

[13] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[14] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[15] L.R. Rabiner, C.H. Lee, B.H. Juang, and J.G. Wilpon. HMM clustering for connected word recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 405–408, Glasgow, UK, 23–26 May 1989.

[16] T. Oates, L. Firoiu, and P.R. Cohen. Using dynamic time warping to bootstrap HMM-based clustering of time series. In R. Sun and C.L. Giles, editors, *Sequence Learning: Paradigms, Algorithms, and Applications*, pages 35–52. Springer-Verlag, Berlin, Germany, 2001.

[17] A. Krogh, M. Brown, I.S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology: applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, 1994.

[18] L.M.D. Owsley, L.E. Atlas, and G.D. Bernard. Self-organizing feature maps and hidden Markov models for machine-tool monitoring. *IEEE Transactions on Signal Processing*, 45(11):2787–2798, 1997.

[19] P. Smyth. Clustering sequences with hidden Markov models. In *Advances in Neural Information Processing Systems 9*, pages 648–654. MIT Press, 1997.

[20] C. Li and G. Biswas. A Bayesian approach to temporal data clustering using hidden Markov models. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 543–550, Stanford, CA, USA, 29 June - 2 July 2000.

[21] M.P. Perrone and S.D. Connell. K-means clustering for hidden Markov models. In *Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition*, pages 229–238, Amsterdam, Netherlands, 11–13 September 2000.

[22] M.H. Law and J.T. Kwok. Rival penalized competitive learning for model-based sequence clustering. In *Proceedings of the Fifteenth International Conference on Pattern Recognition*, volume 2, pages 195–198, Barcelona, Spain, 3–7 September 2000.

[23] W.S. DeSarbo and W.L. Cron. A maximum likelihood methodology for clusterwise linear regresion. *Journal of Classification*, 5(1):249–282, 1988.

[24] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 63–72, San Diego, CA, USA, 15–18 August 1999.

[25] S. Gaffney and P. Smyth. Curve clustering with random effects regression mixtures. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, Florida, USA, 3–6 January 2003.

[26] G.E.P. Box and G.M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden Day, San Francisco, CA, USA, 1970. Revised 1976.

[27] G.E.P. Box, G.M. Jenkins, and G.C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, Englewood Cliffs, NJ, USA, 3rd edition, 1994.

[28] H.Y. Kwok, C.M. Chen, and L. Xu. Comparison between mixture of ARMA and mixture of AR model with application to time series forecasting. In *Proceedings of the Fifth International Conference on Neural Information Processing*, pages 1049–1052, Kitakyushu, Japan, 21–23 October 1998.

[29] M.F. Ramoni, P. Sebastiani, and I.S. Kohane. Cluster analysis of gene expression dynamics. *Proceedings of the National Academy of Sciences*, 99(14):9121–9126, 2002.

[30] K. Kalpakis, D. Gada, and V. Puttagunta. Distance measures for effective clustering of ARIMA time-series. In *Proceedings of the IEEE International Conference on Data Mining*, pages 273–280, San Jose, CA, USA, 29 November - 2 December 2001.

[31] Y. Xiong and D.Y. Yeung. Mixtures of ARMA models for model-based time series clustering. In *Proceedings of the IEEE International Conference on Data Mining*, pages 717–720, Maebashi City, Japan, 9–12 December 2002.

[32] R.A. Redner and H.F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1984.

[33] G. Celeux and J. Diebolt. The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82, 1985.

[34] G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14:315–332, 1992.

[35] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B*, 36(1):111–147, 1974.

[36] M. Stone. Asymptotics for and against cross-validation. *Biometrika*, 64(1):29–35, 1977.

[37] P. Smyth. Model selection for probabilistic clustering using cross-validated likelihood. Technical Report 98-09, Department of Information and Computer Science, University of California, Irvine, Irvine, CA, USA, February 1998.

[38] J.D. Banfield and A.E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803–821, 1993.

[39] R.E. Kass and A.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90:773–795, 1995.

[40] G. Schwartz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[41] L. Ljung. *System Identification Toolbox User's Guide*. MathWorks, 5th edition, 2000.

[42] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani. Mining the stock market: Which measure is best? In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 487–496, Boston, MA, USA, 20–23 August 2000.

[43] Z. A. Keirn and J. I. Aunon. A new mode of communication between man and his surroundings. *IEEE Transactions on Biomedical Engineering*, 37(12):1209–1214, 1990.

[44] S. Zhong and J. Ghosh. HMMs and coupled HMMs for multi-channel EEG classification. In *Proceedings of the 2002 International Joint Conference on Neural Networks*, pages 1154–1159, Hawaii, USA, 12–17 May 2002.

A Derivation of M-Step Equations of EM Algorithm for ARMA Mixtures

Recall that in the M-step of the EM algorithm, we find a new parameter estimate by maximizing $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$.

We first maximize $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ with respect to each $P(\omega_k)$, subject to the constraint that $\sum_{k=1}^{M} P(\omega_k) = 1$. This can be solved using the Lagrangian multiplier method:

$$\frac{\partial}{\partial P(\omega_k)}\bigg(Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big) - \lambda\Big(\sum_{k=1}^{M} P(\omega_k) - 1\Big)\bigg) = \frac{\partial Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)}{\partial P(\omega_k)} - \lambda = 0. \quad \text{(A.1)}$$

We first rewrite $Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big)$ as

$$Q\Big(\mathbf{\Theta}|\mathbf{\Theta}(t)\Big) = R(\mathbf{\Theta}) + S(\mathbf{\Theta}),$$

where

$$R(\mathbf{\Theta}) = \sum_{i=1}^{N}\sum_{k=1}^{M} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big) \ln P(\mathbf{x}_i|\omega_k, \mathbf{\Phi}_k),$$
$$S(\mathbf{\Theta}) = \sum_{i=1}^{N}\sum_{k=1}^{M} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big) \ln P(\omega_k).$$

Since

$$\frac{\partial S(\mathbf{\Theta})}{\partial P(\omega_k)} = \frac{1}{P(\omega_k)}\sum_{i=1}^{N} P(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)),$$

we can apply the above constraint to obtain the best estimate as

$$\hat{P}(\omega_k) = \frac{1}{N}\sum_{i=1}^{N} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big). \quad \text{(A.2)}$$

To get the best parameter estimate of σ_k^2, we need to solve this equation:

$$\frac{\partial R(\mathbf{\Theta})}{\partial \sigma_k^2} = \sum_{i=1}^{N} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big)\Big[-\frac{n}{2\sigma_k^2} + \frac{1}{2\sigma_k^4}\sum_{t=1}^{n} e_{i,t}^2\Big] = 0,$$

that is,

$$\sum_{i=1}^{N} P\Big(\omega_k|\mathbf{x}_i, \mathbf{\Theta}(t)\Big)\Big[-\frac{n\sigma_k^2}{2} + \frac{1}{2}\sum_{t=1}^{n} e_{i,t}^2\Big] = 0,$$

or

$$\sum_{i=1}^{N}\Big[-P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\frac{n\sigma_k^2}{2}+\frac{1}{2}P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}e_{i,t}^2\Big]=0.$$

We get the best parameter estimate of σ_k^2 as

$$\hat{\sigma}_k^2=\frac{\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}e_{i,t}^2\Big]}{\sum_{i=1}^{N}\Big[nP\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\Big]}. \tag{A.3}$$

To get the best estimates of $\boldsymbol{\phi}_k$ and $\boldsymbol{\theta}_k$, we need to solve the following equations:

$$\frac{\partial R(\boldsymbol{\Theta})}{\partial\phi_{k,j}}=\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\Big(-\frac{1}{2\sigma_k^2}\Big)\sum_{t=1}^{n}\Big(2\,e_{i,t}\,\frac{\partial e_{i,t}}{\partial\phi_{k,j}}\Big)\Big]=0,\qquad j=0,1,2,\ldots,p.$$

Since

$$\frac{\partial e_{i,t}}{\partial\phi_{k,0}}=-1,$$

we have

$$\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}e_{i,t}\Big]=0,$$

or

$$\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}\Big(x_{i,t}-\phi_{k,0}-\sum_{l=1}^{p}\phi_{k,l}\,x_{i,t-l}-\sum_{l=1}^{q}\theta_{k,l}\,e_{i,t-l}\Big)\Big]=0,$$

or

$$\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\Big(\sum_{t=1}^{n}x_{i,t}-n\,\phi_{k,0}-\sum_{l=1}^{p}\phi_{k,l}\sum_{t=1}^{n}x_{i,t-l}-\sum_{l=1}^{q}\theta_{k,l}\sum_{t=1}^{n}e_{i,t-l}\Big)\Big]=0,$$

that is,

$$n\,\phi_{k,0}\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\Big]+\sum_{l=1}^{p}\phi_{k,l}\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}x_{i,t-l}\Big]+$$

$$\sum_{l=1}^{q}\theta_{k,l}\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}e_{i,t-l}\Big]=\sum_{i=1}^{N}\Big[P\Big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\Big)\sum_{t=1}^{n}x_{i,t}\Big]. \tag{A.4}$$

Since

$$\frac{\partial e_{i,t}}{\partial\phi_{k,j}}=-x_{i,t-j},\qquad j=1,2,\ldots,p,$$

we have

$$\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}\big(e_{i,t}\,x_{i,t-j}\big)\Big]=0,$$

or

$$\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}\big(x_{i,t}\,x_{i,t-j}-\phi_{k,0}\,x_{i,t-j}-\sum_{l=1}^{p}\phi_{k,l}\,x_{i,t-l}\,x_{i,t-j}-\sum_{l=1}^{q}\theta_{k,l}\,e_{i,t-l}\,x_{i,t-j}\big)\Big]=0,$$

or

$$\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\big(\sum_{t=1}^{n}x_{i,t}\,x_{i,t-j}-\phi_{k,0}\sum_{t=1}^{n}x_{i,t-j}-\sum_{l=1}^{p}\phi_{k,l}\sum_{t=1}^{n}x_{i,t-l}\,x_{i,t-j}-\sum_{l=1}^{q}\theta_{k,l}\sum_{t=1}^{n}e_{i,t-l}x_{i,t-j}\big)\Big]=$$

that is,

$$\phi_{k,0}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}x_{i,t-j}\Big]+\sum_{l=1}^{p}\phi_{k,l}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}x_{i,t-l}\,x_{i,t-j}\Big]+$$

$$\sum_{l=1}^{q}\theta_{k,l}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}e_{i,t-l}\,x_{i,t-j}\Big]=\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}x_{i,t}\,x_{i,t-j}\Big],\qquad j=1,2,\ldots,p. \tag{A.5}$$

For the same reason, we have

$$\frac{\partial e_{i,t}}{\partial\theta_{k,j}}=-e_{i,t-j},$$

$$\phi_{k,0}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}e_{i,t-j}\Big]+\sum_{l=1}^{p}\phi_{k,l}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}x_{i,t-l}\,e_{i,t-j}\Big]+$$

$$\sum_{l=1}^{q}\theta_{k,l}\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}e_{i,t-l}\,e_{i,t-j}\Big]=\sum_{i=1}^{N}\Big[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\sum_{t=1}^{n}x_{i,t}\,e_{i,t-j}\Big],\qquad j=1,2,\ldots,q. \tag{A.6}$$

Combining the $(1+p+q)$ equations from Equations (A.4), (A.5) and (A.6), we get

$$\widehat{W}_k\;\widehat{\boldsymbol{\delta}}_k=\widehat{U}_k,$$

or

$$\widehat{\boldsymbol{\delta}}_k=\big(\widehat{W}_k\big)^{-1}\widehat{U}_k, \tag{A.7}$$

where

$$\widehat{\boldsymbol{\delta}}_k=(\phi_{k,0}\;\phi_{k,1}\;\phi_{k,2}\;\ldots\;\phi_{k,p}\;\theta_{k,1}\;\theta_{k,2}\;\ldots\;\theta_{k,q})^T,$$

$$\widehat{W}_k=\sum_{i=1}^{N}\left[P\big(\omega_k|\mathbf{x}_i,\boldsymbol{\Theta}(t)\big)\begin{pmatrix}A & B\\ C & D\end{pmatrix}\right],$$

$$\widehat{U}_k = \sum_{i=1}^{N} \Big[P\Big(\omega_k | \mathbf{x}_i, \mathbf{\Theta}(t)\Big) \Big(a_0 \ a_1 \ a_2 \ \ldots \ a_p \ c_1 \ c_2 \ \ldots \ c_q\Big)^T \Big],$$

and

$$A = (a_{uv})_{(0\sim p)\times(0\sim p)}, \quad B = (b_{uv})_{(0\sim p)\times(1\sim q)}, \quad C = (c_{uv})_{(1\sim q)\times(0\sim p)}, \quad D = (d_{uv})_{(1\sim q)\times(1\sim q)};$$

$$\begin{aligned}
a_{00} &= n, \\
a_{0v} &= \sum_{t=1}^{n} x_{i,t-v}, \quad v = 1, 2, \ldots, p, \\
b_{0v} &= \sum_{t=1}^{n} e_{i,t-v}, \quad v = 1, 2, \ldots, q, \\
a_{u0} &= \sum_{t=1}^{n} x_{i,t-u}, \quad u = 1, 2, \ldots, p, \\
c_{u0} &= \sum_{t=1}^{n} e_{i,t-u}, \quad u = 1, 2, \ldots, q, \\
a_{uv} &= \sum_{t=1}^{n} x_{i,t-u}\, x_{i,t-v}, \quad u = 1, 2, \ldots, p, \quad v = 1, 2, \ldots, p, \\
b_{uv} &= \sum_{t=1}^{n} e_{i,t-u}\, x_{i,t-v}, \quad u = 1, 2, \ldots, p, \quad v = 1, 2, \ldots, q, \\
c_{uv} &= \sum_{t=1}^{n} x_{i,t-u}\, e_{i,t-v}, \quad u = 1, 2, \ldots, q, \quad v = 1, 2, \ldots, p, \\
d_{uv} &= \sum_{t=1}^{n} e_{i,t-u}\, e_{i,t-v}, \quad u = 1, 2, \ldots, q, \quad v = 1, 2, \ldots, q, \\
a_0 &= \sum_{t=1}^{n} x_{i,t}, \\
a_v &= \sum_{t=1}^{n} x_{i,t}\, x_{i,t-v}, \quad v = 1, 2, \ldots, p, \\
c_v &= \sum_{t=1}^{n} x_{i,t}\, e_{i,t-v}, \quad v = 1, 2, \ldots, q.
\end{aligned}$$

Table A.1
Clustering results for time series generated by two AR(1) models with the same noise variance but different AR coefficient distribution ranges

Range of AR coefficient	Cluster similarity (min/avg/max)	
	MAR	CEP
(0.55 ± 0.01)	(0.93/0.99/1.00)	(0.93/0.98/1.00)
(0.50 ± 0.01)	(0.83/0.93/0.97)	(0.80/0.93/0.97)
(0.45 ± 0.01)	(0.80/0.88/0.93)	(0.71/0.86/0.93)
(0.40 ± 0.01)	(0.63/0.77/0.90)	(0.63/0.79/0.93)

Table A.2
Clustering results for time series generated by two AR(1) models with the same AR coefficient distribution range but different noise variances

Cluster similarity (min/avg/max)	
MAR	CEP
(1.00/1.00/1.00)	(0.51/0.59/0.67)

Table A.3
Parameter distribution ranges of AR(1) models for generating time series datasets

3-cluster datasets		
Component	AR coefficient	Noise variance
1	(0.20 ± 0.01)	(0.01 ± 0.001)
2	(0.50 ± 0.01)	(0.01 ± 0.001)
3	(0.80 ± 0.01)	(0.01 ± 0.001)
4-cluster datasets		
Component	AR coefficient	Noise variance
1	(0.20 ± 0.01)	(0.01 ± 0.001)
2	(0.50 ± 0.01)	(0.01 ± 0.001)
3	(0.20 ± 0.01)	(0.02 ± 0.001)
4	(0.50 ± 0.01)	(0.02 ± 0.001)
6-cluster datasets		
Component	AR coefficient	Noise variance
1	(0.20 ± 0.01)	(0.01 ± 0.001)
2	(0.50 ± 0.01)	(0.01 ± 0.001)
3	(0.80 ± 0.01)	(0.01 ± 0.001)
4	(0.20 ± 0.01)	(0.02 ± 0.001)
5	(0.50 ± 0.01)	(0.02 ± 0.001)
6	(0.80 ± 0.01)	(0.02 ± 0.001)

Table A.4
Cumulative highest BIC values for a typical 3-cluster time series dataset

# of clusters	2	3	4	5
After round 1	10051	**10116**	10110	10104
After round 2	10051	**10116**	10110	10105
After round 3	10051	**10116**	10110	10105
After round 4	10051	**10116**	10110	10105
After round 5	10051	**10116**	10110	10105

# of clusters	6	7
After round 1	10099	10094
After round 2	10099	10094
After round 3	10099	10094
After round 4	10100	10094
After round 5	10100	10094

Table A.5
Cumulative highest BIC values for a typical 4-cluster time series dataset

# of clusters	2	3	4	5
After round 1	10669	10747	10741	**10806**
After round 2	10669	10747	10741	**10806**
After round 3	10669	10747	**10812**	10806
After round 4	10669	10747	**10812**	10806
After round 5	10669	10747	**10812**	10806

# of clusters	6	7	8
After round 1	10799	10793	10787
After round 2	10799	10793	10787
After round 3	10799	10793	10787
After round 4	10799	10793	10787
After round 5	10799	10793	10787

Table A.6
Cumulative highest BIC values for a typical 6-cluster time series dataset

# of clusters	2	3	4	5	6
After round 1	15199	15872	15941	15934	16142
After round 2	15450	15872	16083	16076	16142
After round 3	15450	15872	16083	16076	16142
After round 4	15450	15872	16083	16149	**16208**
After round 5	15450	15872	16083	16149	**16208**

# of clusters	7	8	9	10
After round 1	**16201**	16194	16188	16180
After round 2	**16201**	16194	16188	16180
After round 3	**16201**	16194	16188	16180
After round 4	16201	16194	16188	16180
After round 5	16201	16195	16188	16180

Table A.7
Number of trials required to find the correct number of clusters for each dataset

Dataset	# of trials for each dataset											
3-cluster	1	1	1	1	3	1	1	1	1	3	1	1
4-cluster	1	2	5	3	9	4	5	1	1	4	1	2
6-cluster	1	3	1	6	4	1	1	2	1	3	8	4

Table A.8
Parameter specification of the AR(p) models used in the four experiments

Model	Experiment 1				
Parameters	ϕ_0	ϕ_1	ϕ_2	ϕ_3	σ^2
$AR(3)_1$	1.13	-0.05	0.52	-0.21	0.26
$AR(3)_2$	0.33	0.36	0.10	0.16	0.07
$AR(3)_3$	0.62	0.34	0.27	0.06	0.34
Model	Experiment 2				
Parameters	ϕ_0	ϕ_1	ϕ_2	ϕ_3	σ^2
$AR(3)_1$	1.13	-0.05	0.52	-0.21	0.27
$AR(3)_2$	0.33	0.36	0.10	0.16	0.27
$AR(3)_3$	0.62	0.34	0.27	0.06	0.27
Model	Experiment 3				
Parameters	ϕ_0	ϕ_1	ϕ_2	ϕ_3	σ^2
$AR(3)_1$	1.13	-0.05	0.52	-0.21	0.26
$AR(3)_2$	0.57	0.36	0.10	0.16	0.07
$AR(3)_3$	0.50	0.34	0.27	0.06	0.34
Model	Experiment 4				
Parameters	ϕ_0	ϕ_1	ϕ_2	ϕ_3	σ^2
AR(1)	1.13	-0.05	-	-	0.27
AR(2)	0.33	0.36	0.10	-	0.07
AR(3)	0.62	0.34	0.27	0.06	0.34

Table A.9
Average impurity rates of the 24 experiments

	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
Balanced	25	50	100	25	50	100	25	50	100	25	50	100
AR(1)	0.03	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AR(2)	0.03	0.03	0.00	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AR(3)	0.03	0.03	0.00	0.03	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00
AR(4)	0.00	0.03	0.00	0.03	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00
AR(5)	0.03	0.03	0.00	0.03	0.00	0.00	0.07	0.00	0.00	0.00	0.00	0.00
	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
Unbalanced	25	50	100	25	50	100	25	50	100	25	50	100
AR(1)	0.03	0.00	0.00	0.07	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00
AR(2)	0.03	0.00	0.00	0.07	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00
AR(3)	0.03	0.00	0.00	0.17	0.00	0.00	0.17	0.00	0.00	0.00	0.00	0.00
AR(4)	0.03	0.00	0.00	0.23	0.00	0.00	0.17	0.00	0.00	0.00	0.00	0.00
AR(5)	0.03	0.00	0.00	0.30	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00

Table A.10
Parameter specification of the ARMA(2,1) models

Model	Experiment 1			
Parameters	ϕ_1	ϕ_2	ϕ_3	σ^2
$AR(2,1)_1$	-0.05	0.52	0.44	0.26
$AR(2,1)_2$	0.36	0.10	0.06	0.07
$AR(2,1)_3$	0.34	0.27	-0.25	0.34
Model	Experiment 2			
Parameters	ϕ_1	ϕ_2	ϕ_3	σ^2
$AR(2,1)_1$	-0.05	0.52	0.44	0.27
$AR(2,1)_2$	0.36	0.10	0.06	0.27
$AR(2,1)_3$	0.34	0.27	-0.25	0.27

Table A.11
Clustering results for time series generated by the ARMA(2,1) models

	Cluster similarity (min/avg/max)
Experiment 1	(1.00/1.00/1.00)
Experiment 2	(0.93/0.98/1.00)

Table A.12
Cluster similarity values for first four datasets

	MAR	CEP
Personal income dataset	0.90	0.84
ECG dataset	0.94	0.94
Temperature dataset	1.00	0.65
Population dataset	0.64	0.64

Table A.13
Cluster similarity values for EEG datasets using univariate models

	Task 1	Task 2	Task 3
Channel F4	0.90	1.00	0.95
Channel P8	0.90	1.00	0.90

Table A.14
Cluster similarity values for EEG datasets using simple multivariate models

	Task 1	Task 2	Task 3
Channel F4 & P8	0.95	1.00	0.95