

Wael Abd-Elmageed · Aly I. El-Osery
Christopher E. Smith

Estimating time-varying densities using a stochastic learning automaton

© Springer-Verlag 2005

Abstract The popular Expectation Maximization technique suffers a major drawback when used to approximate a density function using a mixture of Gaussian components; that is the number of components has to be *a priori* specified. Also, Expectation Maximization by itself cannot estimate time-varying density functions. In this paper, a novel stochastic technique is introduced to overcome these two limitations. Kernel density estimation is used to obtain a discrete estimate of the true density of the given data. A Stochastic Learning Automaton is then used to select the number of mixture components that minimizes the distance between the density function estimated using the Expectation Maximization and discrete estimate of the density. The validity of the proposed approach is verified using synthetic and real univariate and bivariate observation data.

1 Introduction

The Expectation Maximization algorithm (EM) [9] perhaps is the most frequently used technique for estimating class-conditional probability density functions (PDF) in both univariate and multivariate cases. It has been widely applied in computer vision [3], speech processing [19] and pattern rec-

ognition [4] applications. In all of these areas, EM is used to model the underlying PDF of a data set using a given parametric model. Usually a mixture of Gaussians with a finite number of components is used to approximate the density function.

The advantage of EM is providing a closed-form analytical representation of the PDF. However, for mixtures of Gaussians, it suffers a number of drawbacks. For example, the number of components has to be *a priori* specified. This number has to be accurately specified in order to balance between the accuracy of the estimated PDF on one hand, and its complexity, which affects the speed of both training and testing, on the other. Also, the resulting PDF is very sensitive to the initialization of the mixture parameters (particularly the mean vectors.) Therefore, it is not guaranteed to converge to the global minimum. Finally, EM by itself cannot be used to estimate time-varying densities.

On the other hand, kernel density estimators (also known as Parzen Windows and nonparametric density estimators) such as [20], nonparametrically provide an accurate estimate of the true density function but has a complexity of $\mathcal{O}(N \times M)$, where N is the number of observations and M is the number of points for which the density need to be estimated. Elgammal et al. [10] reduced this complexity to $\mathcal{O}(N + M)$, for up to 3D data, using the Fast Gauss Transform (FGT) [12]. Yang et al. [23] introduced the Improved FGT for computing the density with $\mathcal{O}(N + M)$ complexity in higher dimensions. Therefore, using EM is advantageous since it computes the density with constant complexity compared to the linear (at best) complexity of the nonparametric techniques.

Several attempts have been made to overcome the drawbacks of the EM algorithm. Figueirde and Jain [11] broadly classify these methods into two categories: deterministic approaches and stochastic approaches. Deterministic methods, such as [6–8], are based on selecting the number of components according to some model selection criterion, which usually contains an increasing function that penalizes higher number of components. In [5], [18] and [21] stochastic approaches based on Markov Chain Monte Carlo methods are used.

W. Abd-Elmageed (✉)
Computer Vision Laboratory, Institute for Advanced Computer Studies,
University of Maryland,
College Park, MD 20742, USA
E-mail: wamageed@umiacs.umd.edu

Aly I. El-Osery
Department of Electrical Engineering,
New Mexico Tech.,
Socorro NM 87801, USA
E-mail: elosery@ee.nmt.edu

C.E. Smith
Department of Electrical and Computer Engineering,
University of New Mexico,
Albuquerque NM 87131, USA
E-mail: chsmith@ece.unm.edu

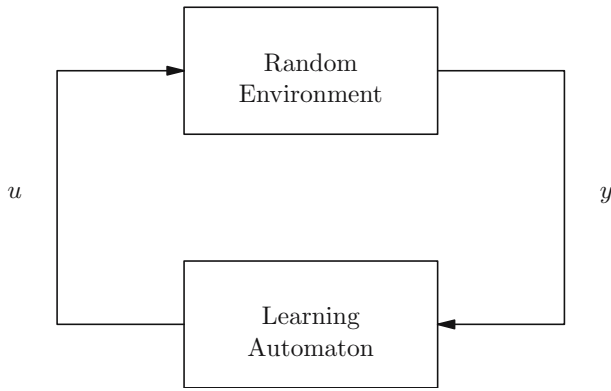


Fig. 1 Automaton operating in a random environment where u represents the optimal action as selected by the automaton and y indicates the satisfaction of the environment with the selected action

This paper introduces a new stochastic approach to overcome the drawbacks of the EM algorithm. The proposed approach is based on computing a distance measure between a density function estimated using the EM and the discrete density estimated using kernel methods. A Stochastic Learning Automaton (SLA) [15] is then used to find the number of mixture components that minimizes the distance.

Studying of Learning Automata started in the early 1960s [14, 22]. Learning automata theory provides a framework for the design of automata which interact with a random environment and dynamically learn the behavior that minimizes the probability of a penalty. Since the 1960s, this field has seen vast improvements and developments [1, 17]. The main advantage of SLA is that no knowledge about random environment, in which the automaton operates, or the function to be optimized is required.

This paper is organized as follows. Section 2 introduces the proposed approach for selecting the number of mixture components. Experimental results verifying the validity of the proposed approach are presented in Sec. 3. Section 4 concludes the paper and introduces directions for future research.

2 The approach

2.1 Density estimation

Let $X = \{x_1, x_2, \dots, x_N\}$ be a set of N multivariate vectors for which a parametric probability density needs to be estimated. The density function can be modeled as a mixtures of Gaussian components as shown in Eq. 1

$$p_{EM}(\mathbf{x}) = \sum_{j=1}^k \pi_j p(\mathbf{x}|\Theta_j), \quad (1)$$

where x is the vector, k is the number of mixture components, π_j s are the mixing weights such that $\sum_{j=1}^k \pi_j = 1$ and Θ_j is the parameter set of component j . The conditional probability density of components j is given by

$$p(\mathbf{x}|\Theta_j) = \frac{1}{(2\pi)^{d/2} |\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_j)^T \Sigma_j^{-1} (\mathbf{x} - \mu_j)\right), \quad (2)$$

where μ_j and Σ_j are the mean and covariance matrix of component j respectively and d is the dimensionality. EM can be used to estimate the parameters of the mixture (i.e. π_j s and Θ_j s) by iteratively applying the update Eqs. 3, 4 and 5, as follows:

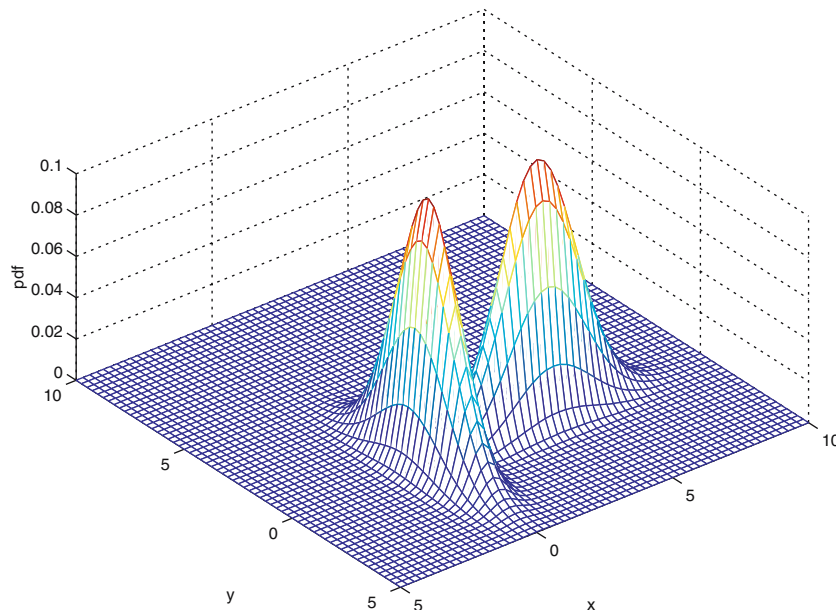


Fig. 2 Mixture used in first example. Two well-separated bivariate Gaussians

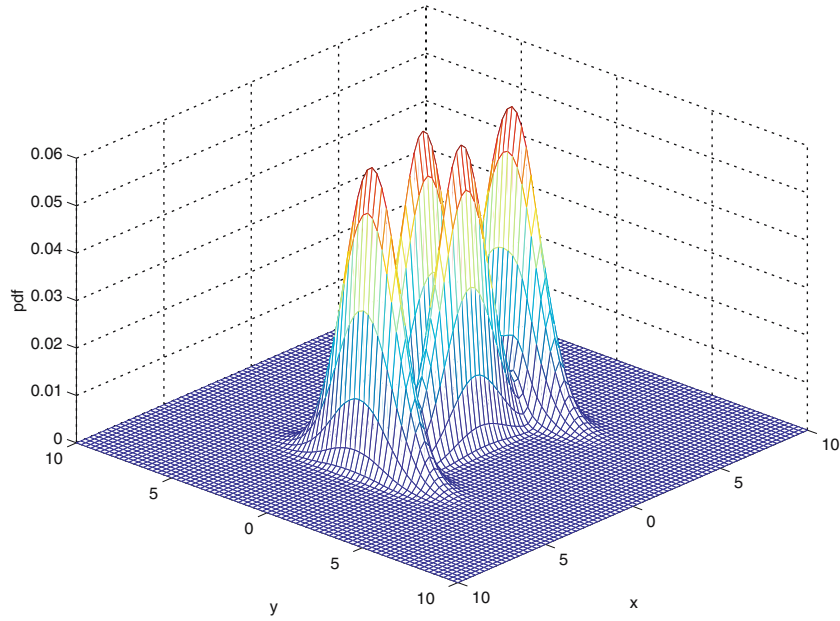


Fig. 3 Mixture used in the second example. Four Gaussians that form a rectangular density function

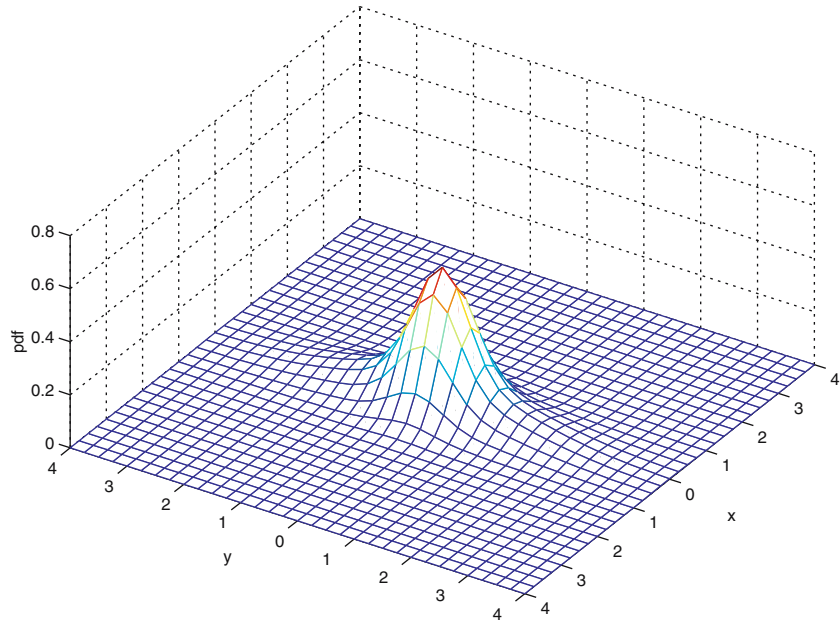


Fig. 4 Mixture used in the third experiment. A narrow-variance Gaussian embedded into a wide-variance Gaussian

$$\mu_j^i = \frac{\sum_{\forall \mathbf{x} \in \mathbf{X}} \mathbf{x} P(j|\mathbf{x})}{\Gamma_j^{i-1}},$$

$$\pi_j^i = \frac{\Gamma_j^{i-1}}{N} \quad \text{and,}$$

and

$$\Sigma_j^i = \frac{\sum_{\forall \mathbf{x} \in \mathbf{X}} P^{i-1}(j|\mathbf{x})(\mathbf{x} - \mu_j^i)(\mathbf{x} - \mu_j^i)^T}{\Gamma_j^{i-1}},$$

where

$$\Gamma_j^{i-1} = \sum_{\forall \mathbf{x} \in \mathbf{X}} P^{i-1}(j|\mathbf{x}), \quad (6)$$

$$P(j|\mathbf{x}) = \frac{\pi_j P(\mathbf{x}|j)}{p(\mathbf{x})}, \quad (7)$$

(5) and i is the iteration number. The initial values, π_j^0 , μ_j^0 and Σ_j^0 , are system parameters and they strongly affect the convergence of the EM.

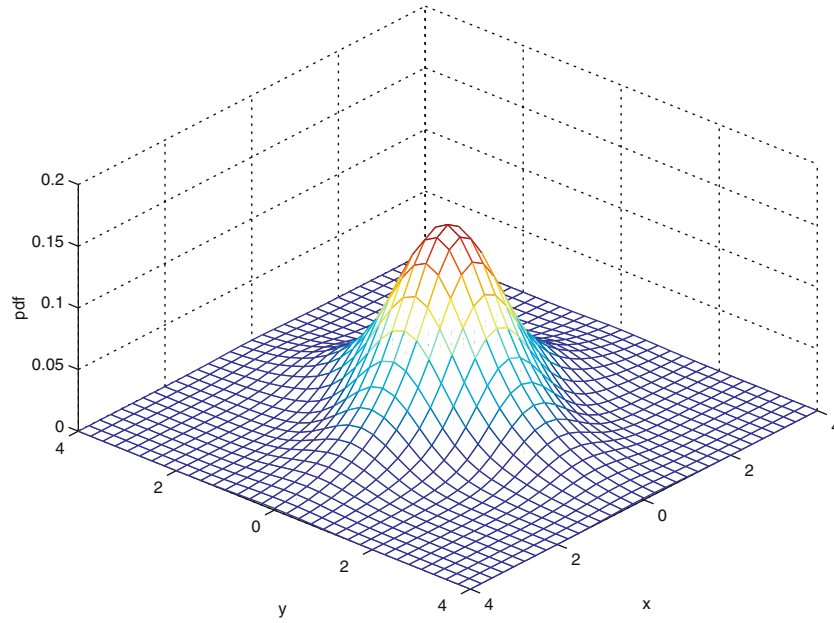


Fig. 5 Mixture used in the fourth example. Two Gaussians that form a cross

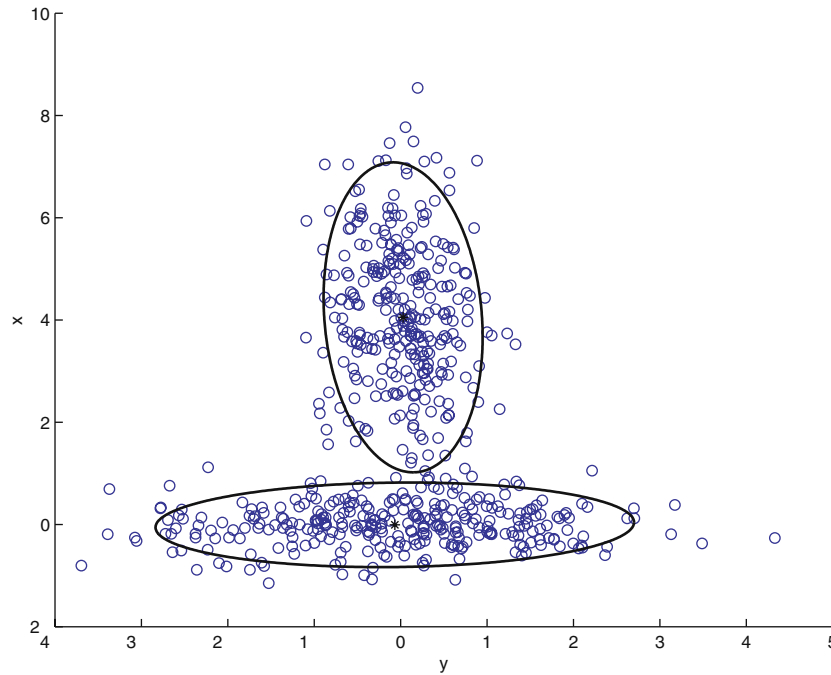


Fig. 6 Clustering result for a data set generated using the PDF of Fig. 2

On the other hand, kernel density estimation [20] can be used to estimate a nonparametric version of the density function of X , as follows

$$p_{KDE}(x) = \frac{1}{N} \sum_{j=1}^N \frac{1}{|H_j|} \mathcal{K} \left(\frac{x - x_j}{H_j} \right) \quad (8)$$

where H_j is the adaptive bandwidth matrix of vector j and $\mathcal{K}(\cdot)$ is the kernel function. Here, the standard multivariate

Gaussian is used as the kernel function as shown in Eq. 9.

$$\mathcal{K}(x) = \frac{1}{(2\pi)^{d/2}} \exp \left(-\frac{1}{2} x^T x \right) \quad (9)$$

The direct evaluation of p_{KDE} at M points requires $\mathcal{O}(M \times N)$ evaluations of the kernel function \mathcal{K} . Elgammal et al. [10] reduced this complexity to $\mathcal{O}(M + N)$ using FGT [12] without significant loss of accuracy. The FGT method works for up to 3D data. In [23] Yang et al. introduced the Improved

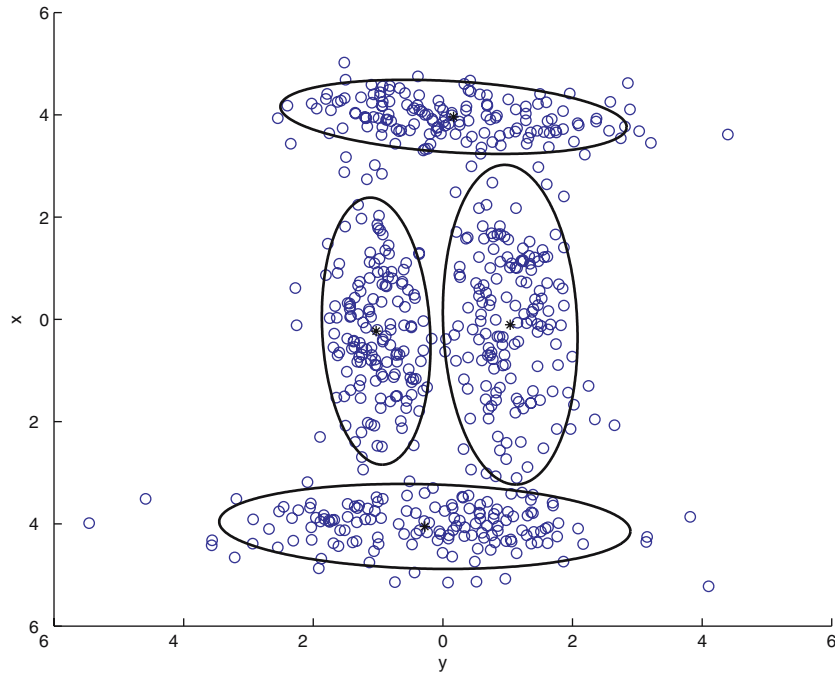


Fig. 7 Clustering result for a data set generated using the PDF of Fig. 3

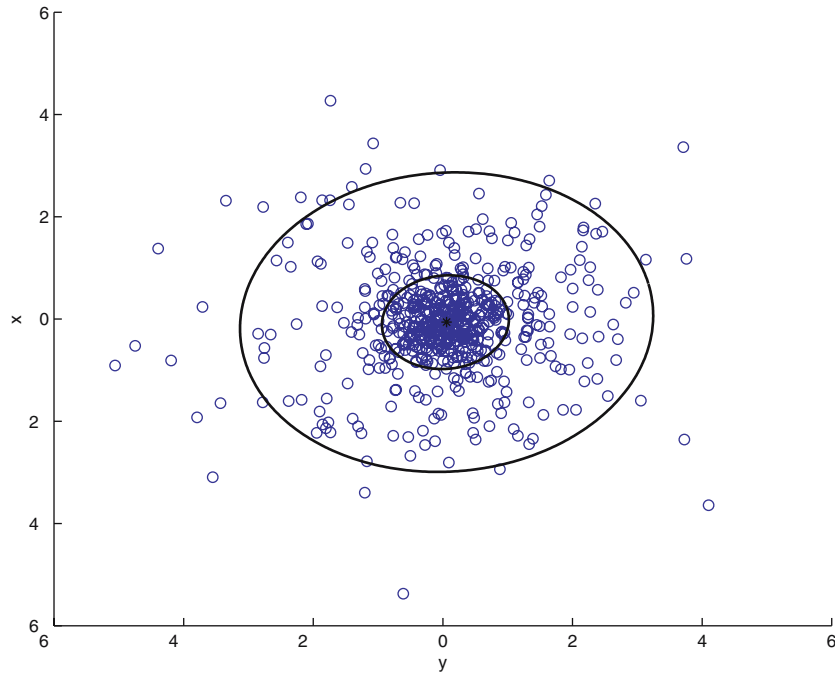


Fig. 8 Clustering result for a data set generated using the PDF of Fig. 4

FGT in order to compute the kernel-based estimate in $\mathcal{O}(M + N)$ for higher than three dimensions.

To quantify the similarity of the PDF estimated by the EM, and the PDF approximated by kernel density estimation, a distance measure is needed. In [16], Kullback developed a measure of the similarity between two density functions known as Kullbak-Leibler (KL) distance (divergence).

The distance between p_{EM} and p_{KDE} can be approximated using the KL distance as follows:

$$\delta(p_{KDE}(\mathbf{x}), p_{EM}(\mathbf{x})) = \int_{\mathbf{S}} p_{KDE}(\mathbf{x}) \log \frac{p_{KDE}(\mathbf{x})}{p_{EM}(\mathbf{x})} d\mathbf{x}, (10)$$

where δ is the distance between the two density functions. The main shortcoming of this distance measure is that it

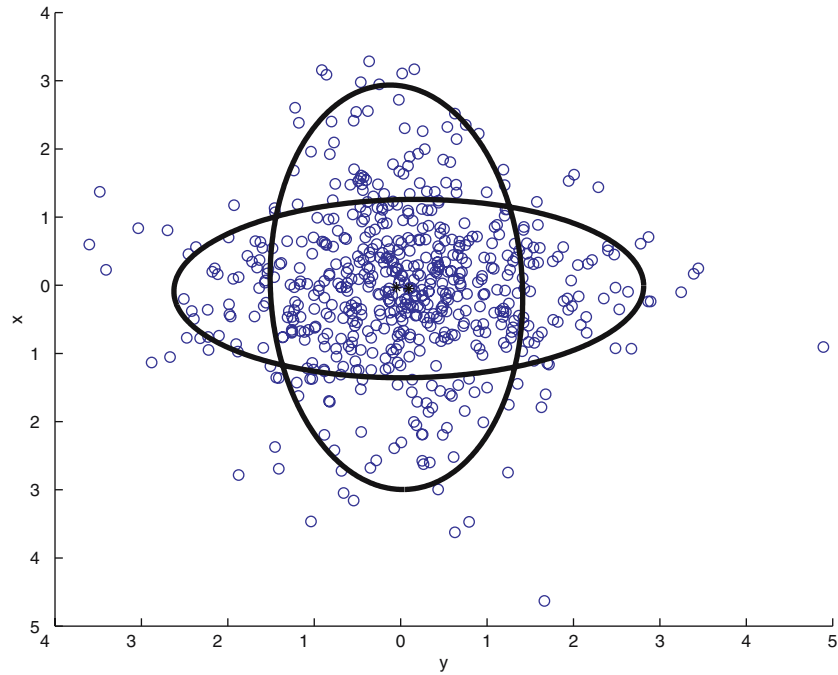


Fig. 9 Clustering result for a data set generated using the PDF of Fig. 5

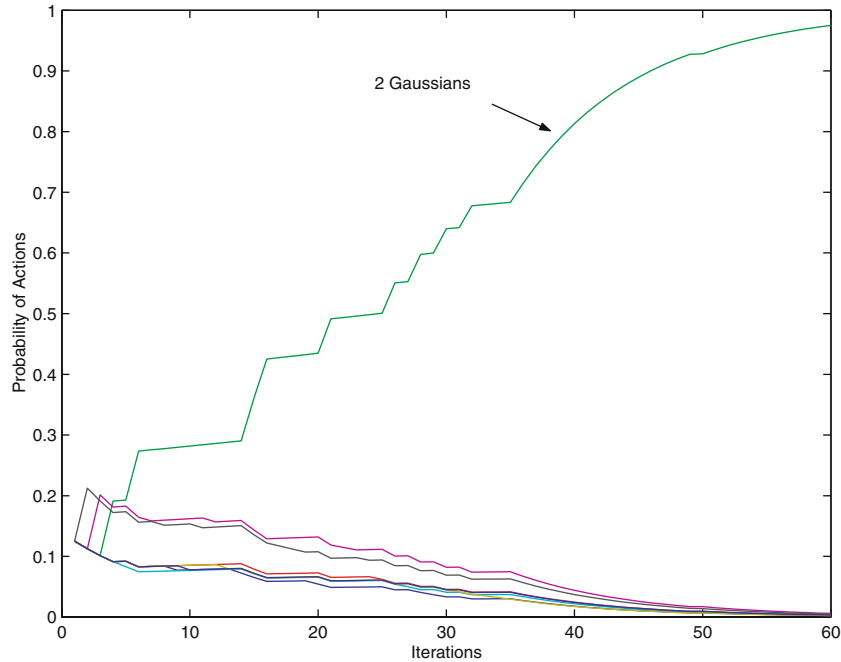


Fig. 10 Probability distribution for the SLA actions applied to the data set generated using the PDF of Fig. 2

is nonsymmetric. In other words, $\delta(p_{KDE}(\mathbf{x}), p_{EM}(\mathbf{x})) \neq \delta(p_{EM}(\mathbf{x}), p_{KDE}(\mathbf{x}))$. Many approaches have been proposed to symmetrize the KL distance, such as D. Johnson and S. Sinanovic (submitted). In the proposed approach, the KL distance is always computed between $p_{KDE}(\mathbf{x})$ and a variable $p_{EM}(\mathbf{x})$. Therefore, the symmetry of the distance is not significant.

The optimal number of mixture components in the interval $[k_{min}, k_{max}]$ is the one that minimizes the corresponding KL distance. In other words,

$$\hat{k} = \arg_k \min \delta(p_{KDE}(\mathbf{x}), p_{EM}(\mathbf{x})), \quad k = k_{min}, \dots, k_{max} \quad (11)$$

Equation 11 gives a method for estimating the optimal, in terms of the KL divergence, number of components if density

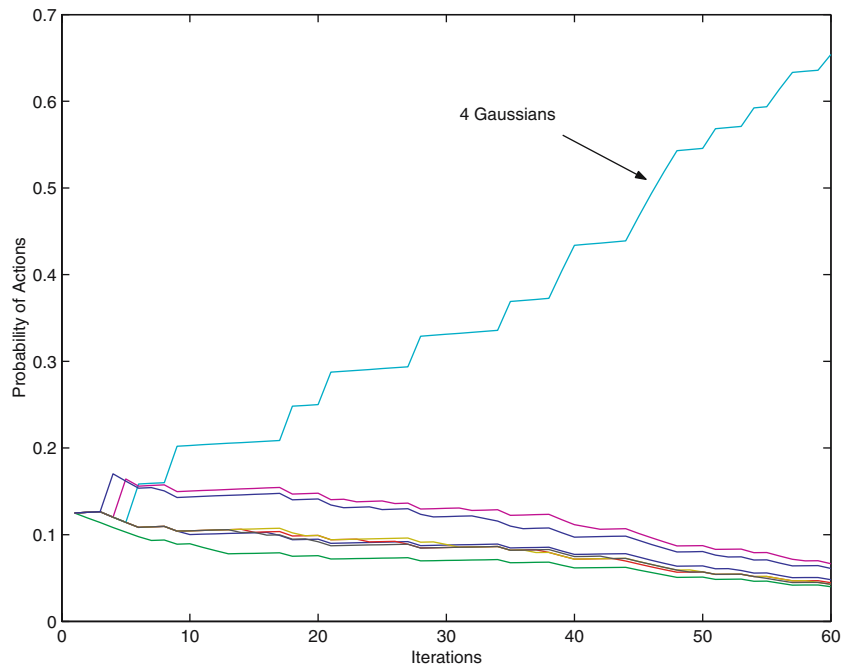


Fig. 11 Probability distribution for the SLA actions applied to the data set generated using the PDF of Fig. 3

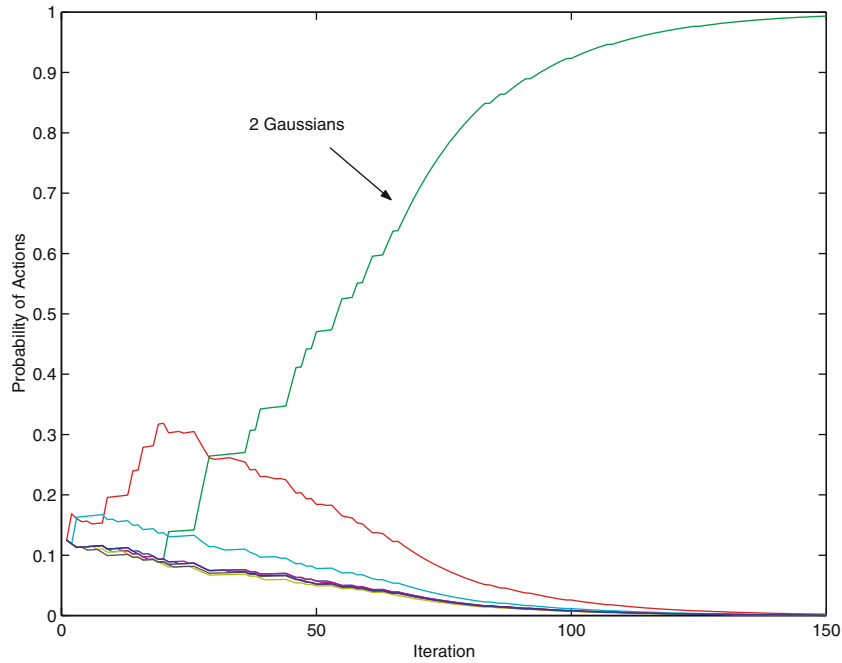


Fig. 12 Probability distribution for the SLA actions applied to the data set generated using the PDF of Fig. 4

of the data does not change with time. One can simply compute KL divergence for all possible values of k . The optimal k is the one that corresponds to the minimum KL. However, if the density of the data is time-varying, a higher-level layer is needed to interact with this change and modify the number of mixture components accordingly.

2.2 Stochastic learning automata

One main advantage of learning automaton is that it requires no knowledge of the environment in which it operates or any analytical knowledge of the function to be optimized. The learning automaton is a sequential machine characterized by

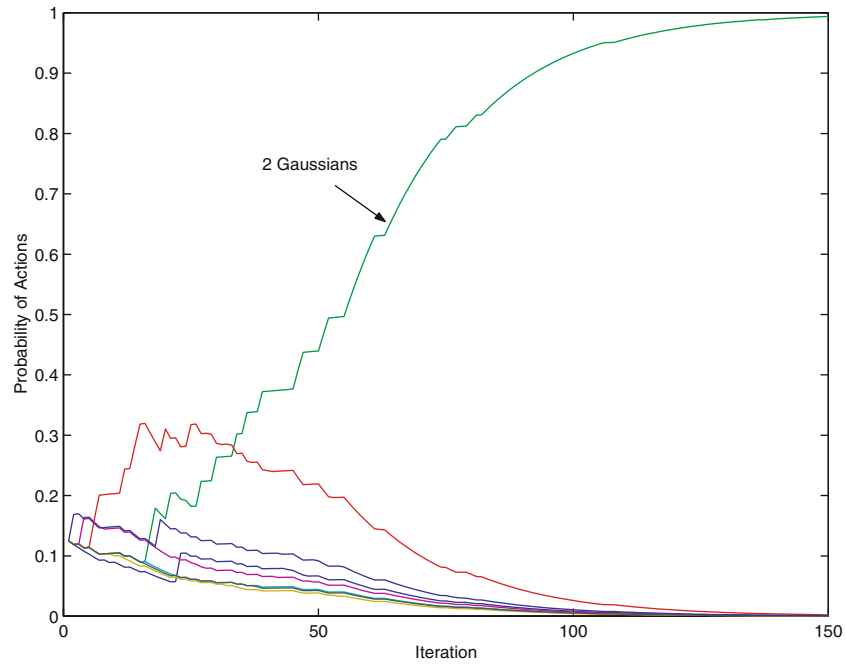


Fig. 13 Probability distribution for the SLA actions applied to the data set generated using the PDF of Fig. 5

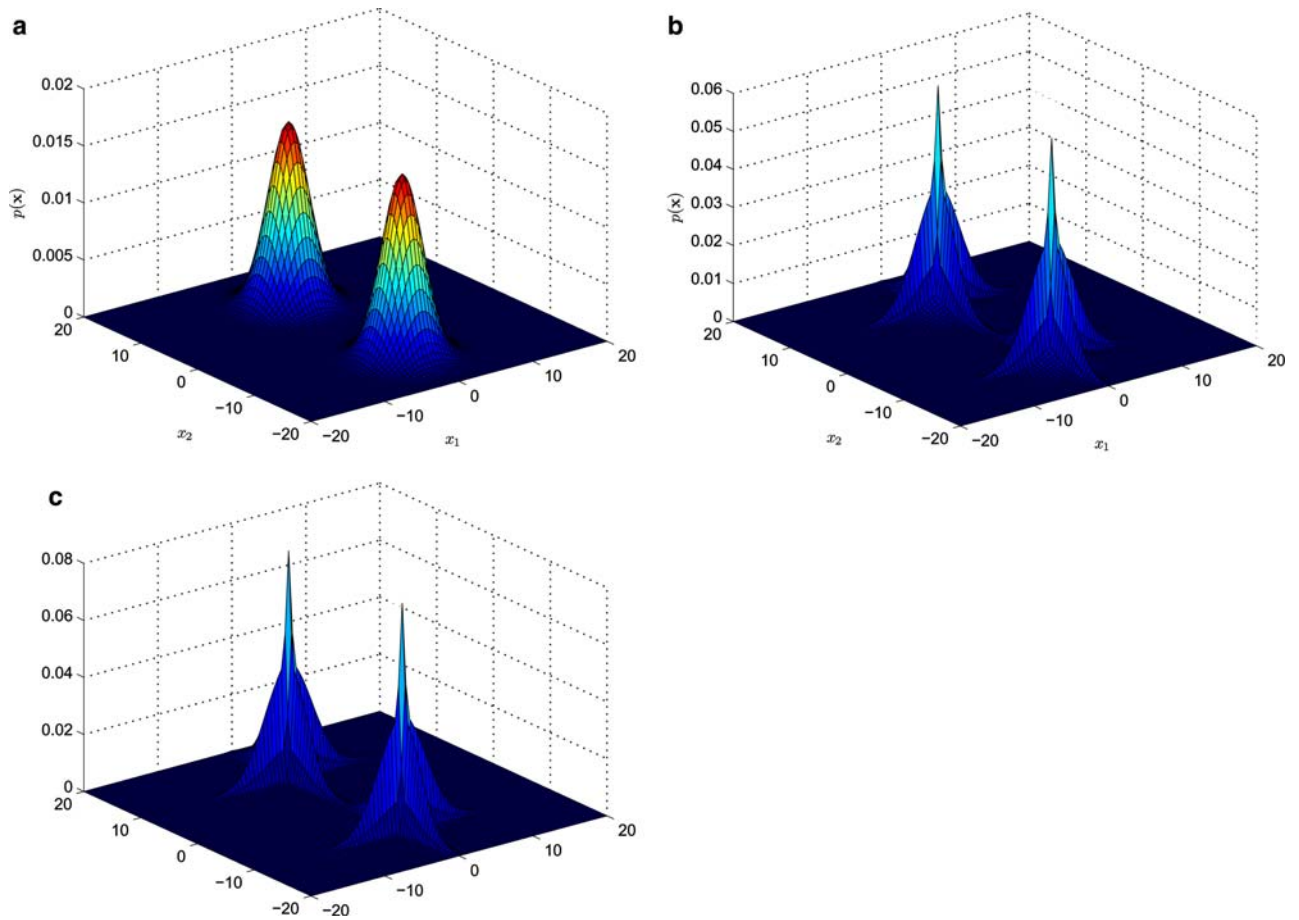


Fig. 14 The underlying PDFs. **a** The underlying PDF from iteration 1 to iteration 100. **b** The underlying PDF from iteration 100 to iteration 200. **c** The underlying PDF from iteration 200 to iteration 300

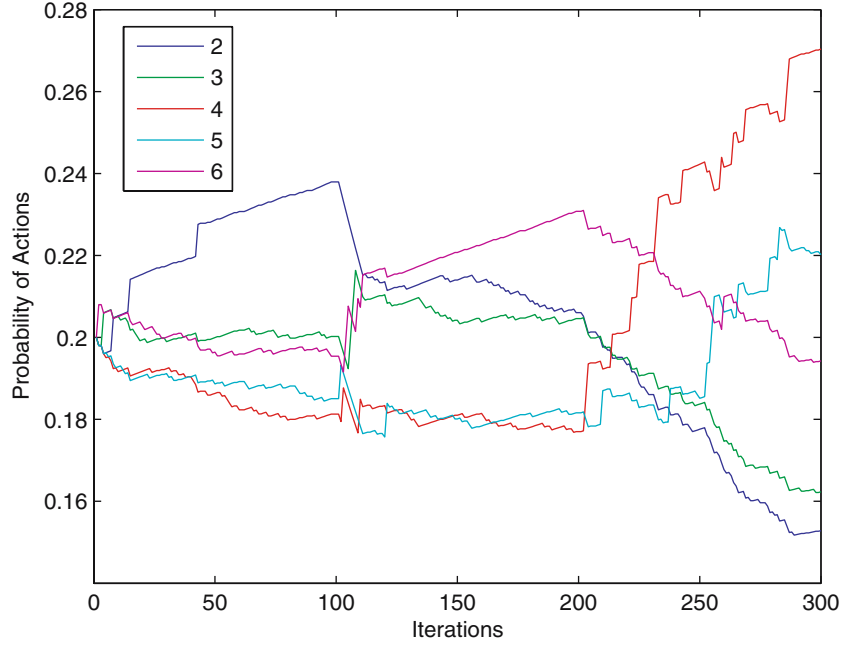


Fig. 15 The performance of the SLA for the time-varying densities shown in Fig. 14

a set of internal states, input actions, state probability distributions, a reinforcement scheme, and an output function and is connected to the environment in feedback loop as shown in Fig. 1.

The probability distribution of the actions is adjusted using reinforcement scheme to achieve the desired objective. At each step, the performance of the SLA through the environment is evaluated and the SLA is either penalized due to an unsatisfactory performance or rewarded because of a satisfactory performance. A stochastic automaton is a quintuple $\{Y, Q, U, F, G\}$ where:

- Y represents the reward/penalty mechanism. Here we use the so-called P -model automaton where $Y = \{0, 1\}$.
- Q is a finite set of states, $Q = \{q_1, \dots, q_s\}$,
- U is a finite set of outputs, $U = \{u_1, \dots, u_m\}$,
- F is the next state function that is given by

$$q(n+1) = F[y(n), q(n)], \quad (12)$$

- And G is a function mapping the next state to the output of the automaton such that

$$u(n) = G[q(n)]. \quad (13)$$

In general, the function F is stochastic and the function G may be deterministic or stochastic. Because of the stochastic nature of the state transitions, stochastic automata are considered suitable for modeling learning systems. If the output of the automaton is u_j , $j = 1, 2, \dots, m$, the random environment generates a penalty with probability τ_j or a reward with probability $(1 - \tau_j)$ based on the environment's dissatisfaction or satisfaction, respectively.

The reinforcement scheme used to update the probability distribution of the actions is as follows [2]. Assume that

the action selected by the automaton at iteration n is u_i , i.e. $u(n) = u_i$.

If $y(n) = 0$ (i.e. the automaton is to be rewarded)

$$P_{u_i}(n+1) = (1 - \alpha)P_{u_i}(n) + \alpha, \quad \text{and} \quad (14)$$

$$P_{u_j}(n+1) = (1 - \alpha)P_{u_j}(n) \quad (j \neq i). \quad (15)$$

If $y(n) = 1$ (i.e. the automaton is to be penalized)

$$P_{u_i}(n+1) = P_{u_i}(n) - v\alpha(1 - P_{u_i}(n)) \left(\frac{H}{1 - H} \right), \quad (16)$$

$$P_{u_j}(n+1) = P_{u_j}(n) + v\alpha P_{u_j}(n) \left(\frac{H}{1 - H} \right), \quad (j \neq i) \quad (17)$$

where P_{u_i} is the probability distribution of action u at time i ,

$$H = \min[P_{u_1}(n), \dots, P_{u_m}(n)], \quad (18)$$

$$0 < \alpha < 1, \quad (19)$$

$$0 < v\alpha < 1, \quad (20)$$

and

$$P_{u_1}(0) = \dots = P_{u_m}(0) = \frac{1}{m}. \quad (21)$$

2.3 The nonparametric expectation maximization

The SLA operates as follows. A range of number of mixture components $\{k_{\min}, \dots, k_{\max}\}$ is specified by the user. This set represents the set of actions, $\{u_1, \dots, u_m\}$, that the SLA will learn based on the environment's responses. At time t

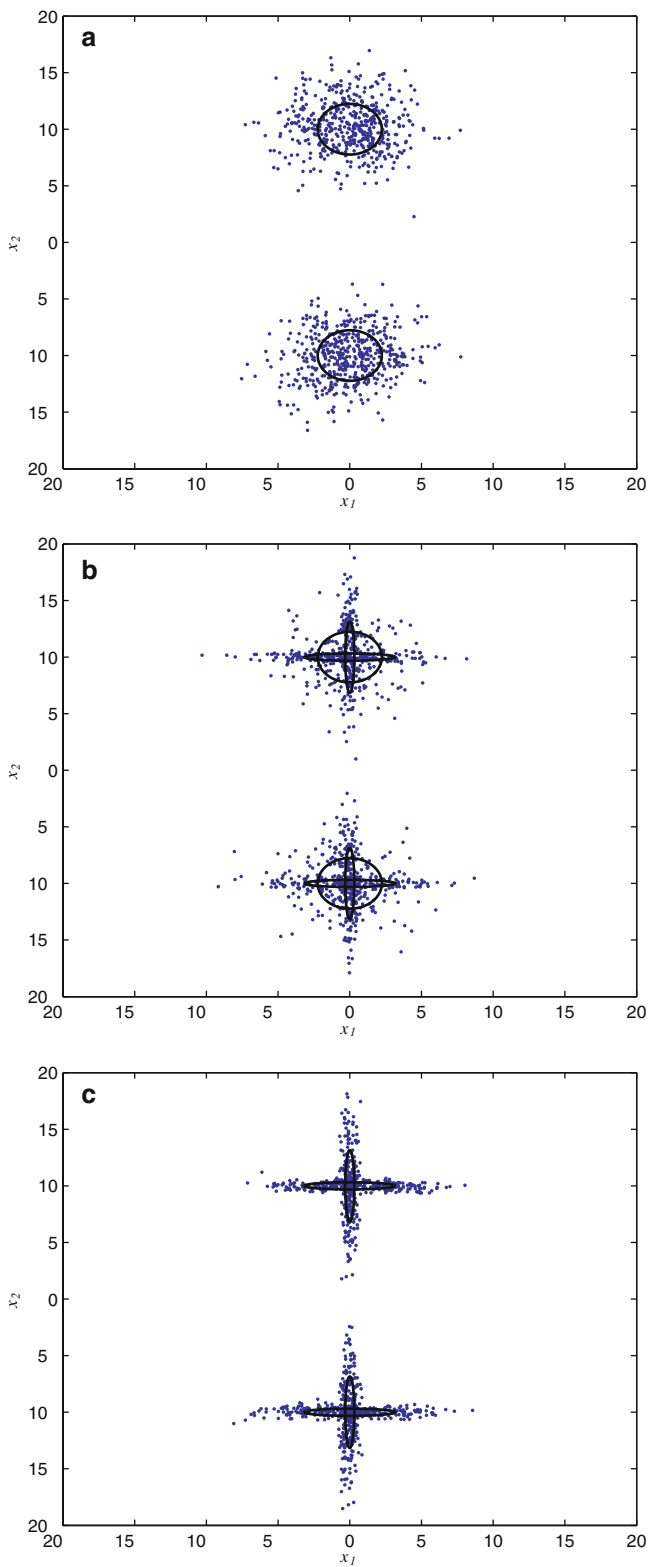
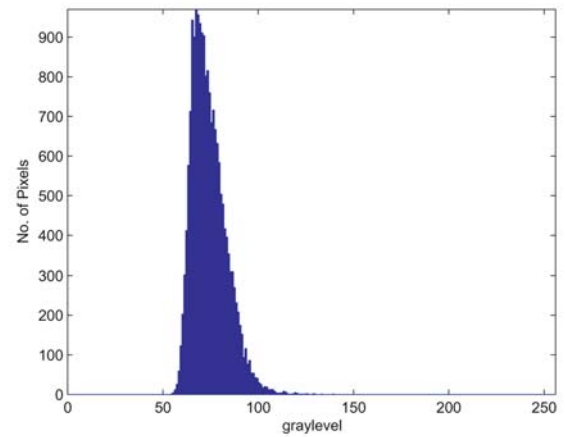


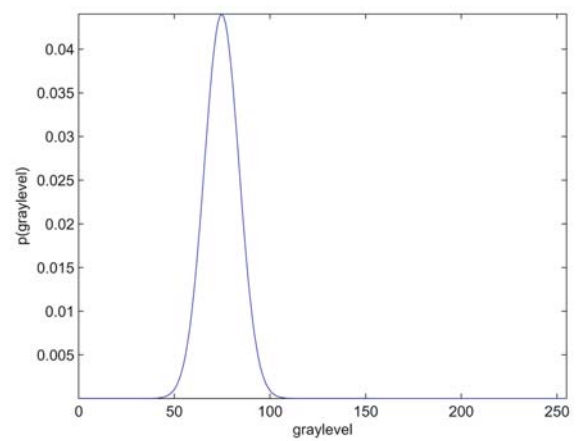
Fig. 16 The underlying PDFs



a Frame 30.

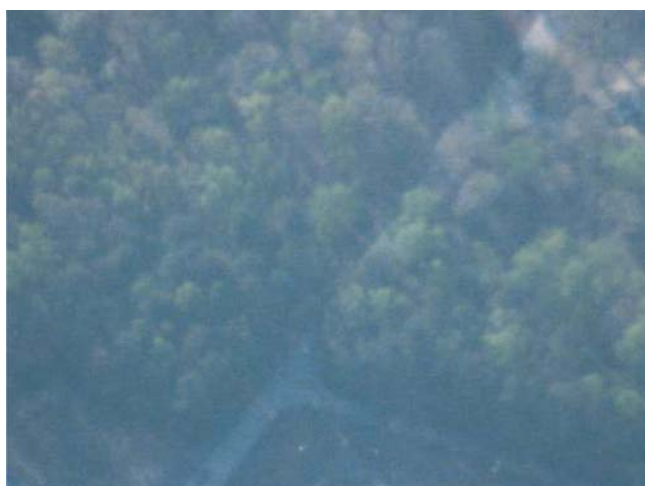


b Histogram of frame 30.

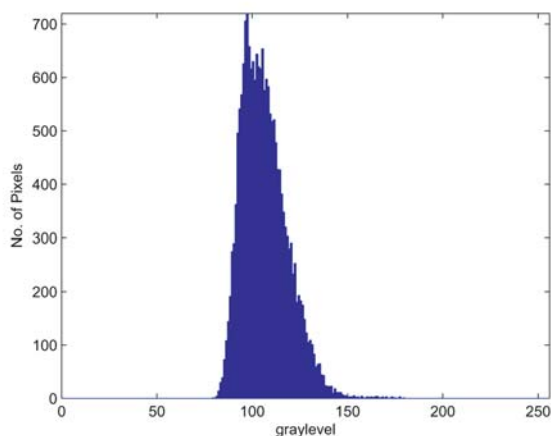


c PDF as estimated using the SLA.

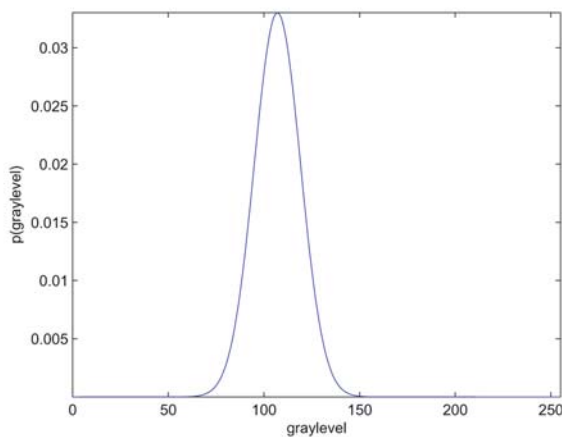
Fig. 17 Result of applying the proposed method on frame 30. **a** Frame 30. **b** Histogram of frame 30. **c** PDF as estimated using the SLA



a Frame 200.



b Histogram of frame 200

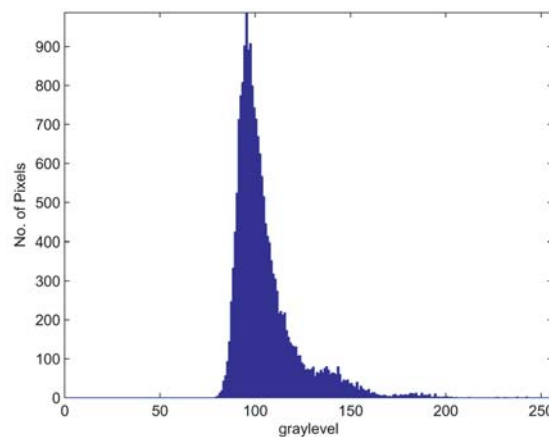


c PDF as estimated using the SLA

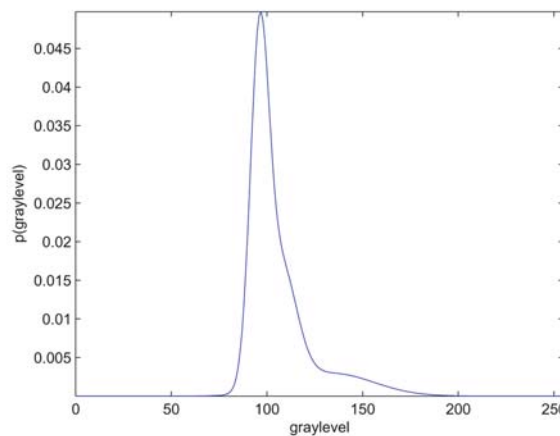
Fig. 18 Result of applying the proposed method on frame 200. **a** Frame 200. **b** Histogram of frame 200. **c** PDF as estimated using the SLA



a Frame 270.



b Histogram of frame 270.

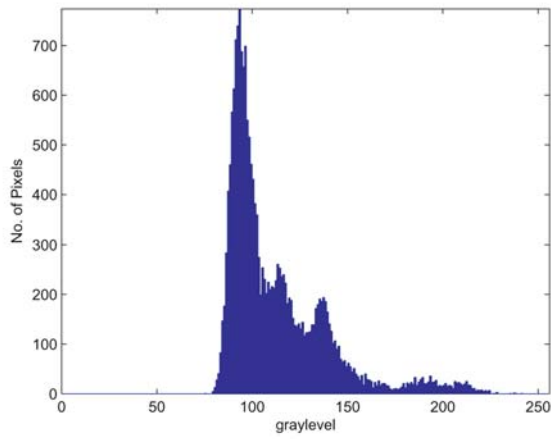


c PDF as estimated using the SLA

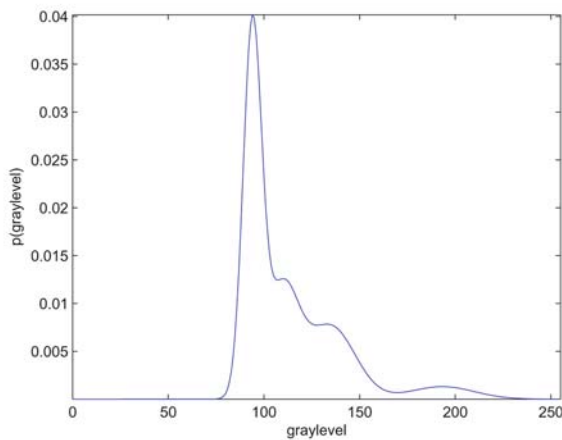
Fig. 19 Result of applying the proposed method on frame 270. **a** Frame 270. **b** Histogram of frame 270. **c** PDF as estimated using the SLA



a Frame 320.



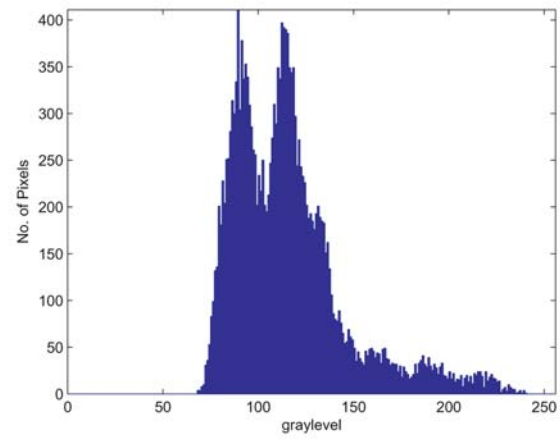
b Histogram of frame 320



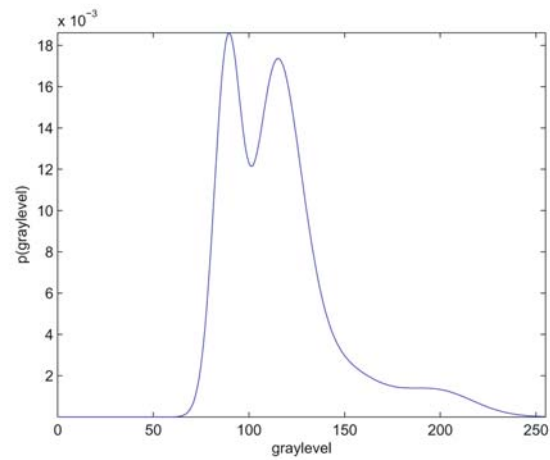
c PDF as estimated using the SLA



a Frame 410.



b Histogram of frame 410.



c PDF as estimated using the SLA.

Fig. 20 Result of applying the proposed method on frame 320. **a** Frame 320. **b** Histogram of frame 320. **c** PDF as estimated using the SLA

Fig. 21 Result of applying the proposed method on frame 410. **a** Frame 410. **b** Histogram of frame 410. **c** PDF as estimated using the SLA

the density p_{KDE} is estimated according to Eq. 8. The SLA selects an action u_i , i.e. a number of mixture components, at random. The density p_{EM} , where the number of components is given by u_i , is computed using the classic EM algorithm. The higher-level application, which represents the environment, assesses the decision of the automaton by computing the distance $\delta(p_{KDE}, p_{EM})$ and comparing it to the minimum distance δ_{\min} . If $\delta(p_{KDE}, p_{EM}) < \delta_{\min}$, then the automaton is rewarded and the probability of selection action is increased and the probabilities of the other actions are decreased based on Eqs. 14 and 15, respectively. Finally, the δ_{\min} is set to $\delta(p_{KDE}, p_{EM})$. Notice that the initial value of δ_{\min} is set to infinity. On the other hand, if $\delta(p_{KDE}, p_{EM}) > \delta_{\min}$, then the action is penalized by decreasing its probability and increasing the probabilities of the other actions, according to Eqs. 16 and 17, respectively. In other words, the environment and the automaton cooperate to find \hat{k} such that

$$\hat{k} = \underset{k}{\operatorname{argmin}} \delta(p_{\text{Parzen}}(\mathbf{x}), p_{EM}(\mathbf{x})) \quad (22)$$

in a stochastic fashion rather than a linear search fashion. This is particularly useful when the underlying density of the data is dynamically changing.

3 Experimental results

In this section, the results of the algorithm introduced in Sect. 2 are demonstrated. The algorithm has been tested with several bivariate data sets with different number of Gaussian components. Density functions of bivariate data are used here because they are easy to visualize. However, the approach can be directly applied without modifications to higher-dimensional feature spaces. Figs. 2, 3, 4 and 5 show four mixtures used to verify the validity of the proposed algorithm. While the data change over time, the underlying PDFs do not change. The mixtures in Figs. 2 and 3 contain two and four Gaussian components, respectively, that are relatively well separated. The mixture in Fig. 4 represents a very narrow Gaussian contained inside a wide Gaussian. Finally, Fig. 5 shows a mixture of two Gaussians that form a cross.

The size of each of the data sets is 600 bivariate observations. The SLA parameters used for all three simulations are $\alpha = 0.05$ and $v = 0.9$. Figures 6, 7, 8 and 9 illustrate the clustering results using mixture parameters estimated by EM and selected by the SLA. The mixtures chosen have different degrees of complexity as is clear from the figures.

The probability distribution of all the actions of each simulation are demonstrated in Figs. 10, 11, 12 and 13. The figures clearly show that the probability distributions of the actions converged to the correct number of Gaussian components in each simulation. In Figs. 10 and 11, where the mixture is fairly simple, the probability of the correct number of components started to dominate after five iterations. As the mixture starts getting more complex in Figs. 12 and 13, the probability of the correct number of components started to dominate after about 30 iterations.

In the following experiment we allowed the underlying PDF to change every 100 iterations. Figure 14 shows the three PDFs used during the simulation. The performance of the SLA is shown in Fig. 15. From the figure we can see that the SLA adapts correctly to the dynamics of the environment. The SLA took between 10 and 20 iterations until finding the optimal mixture. In Fig. 16 we show the mixtures selected by the SLA overlaid on the corresponding data.

Finally, we applied the proposed method to estimating the underlying density of the graylevels of a video sequence. Figures 17a, 18a, 19a, 20a and 21a show five different frames of an airborne video sequence. It is clear from the video sequence that the underlying density varies over time because of the nature of the scene. The graylevel histograms of these frames as well as the PDFs as estimated using the SLA are shown in the Figs. 17b,c, 18b,c, 19b,c, 20b,c and 21b,c.

4 Conclusions and future work

In this paper a novel approach to overcome the drawbacks of Expectation Maximization technique has been introduced. The algorithm automatically estimates the number of Gaussian components that best approximate the given data set.

The proposed approach is based on minimizing a similarity measure between the probability density function estimated using the standard EM, and the discrete density function estimated using kernel density estimation. The distance, obtained by Kullback-Liebler, is minimized using an SLA. The stochastic nature of the distance measure makes the SLA approach a natural choice.

The experimental results show the effectiveness of the approach. The approach was applied to mixtures of various degrees of complexity and in all simulations the mixture parameters were correctly approximated.

A well-known limitation of kernel estimators is that they behave poorly when applied to estimating density function in higher-dimensional spaces. However, recently, Yang et al. [23] proposed an approach for estimating the density function using kernel estimators in high-dimensional feature spaces using a modified FGT [12].

Future work will include testing different reinforcement schemes for updating the action probability distributions. This may help improve the convergence time. Also, since this approach is not specific to a certain distance measure, different measures of distance need to be explored.

References

1. Poznyak AS, Najim K, Gomez-Ramirez E (1994) Learning automata: theory and applications. Self-learning control of finite Marker chains. Pergamon, New York
2. Poznyak AS, Najim K (1997) Learning automata and stochastic optimization. Lecture Notes in Control and Information Sciences 225. Springer
3. Abd-Almageed W, Smith CE (2002) Mixture models for dynamix statistical pressure snakes. In: Proceedings of IEEE internation conference on pattern recognition, pp 721-724

4. Abu-Naser A, Galatsanos NP, Wernick MN, Schonfeld D (1998) Object recognition based on impulse restoration with use of the expectation-maximization algorithm. *J Opt Soc Am A (Optics Image Science and Vision)* 15(9):2327–2340
5. Bensmail H, Celeux G, Rafetry A, Robert C (1997) Inference in model-based cluster analysis. *Stat Comput* 7:1–10
6. Biernacki C, Celeux G, Govaert G (2000) Assessing a Mixture Model for Clustering with the Integrated Classification Likelihood. *IEEE Trans Pattern Anal Mach Intell* 22(7):719–725
7. Campbell J, Fraley C, Murtagh F, Raftery A (1997) Linear Flaw Detection in Woven Textiles Using Model-Based clustering. *Pattern Recognit Lett* 18:1539–1548
8. Dasgupta A, Rafetry A (1998) Detecting Features in Spatial Point Patterns with Clutter Via Model-Based Clustering. *J Am Stat Assoc* 93:294–302
9. Dempster AP, Laird NM, Rubin DB (1977) Maximum Likelihood from Incomplete Data via the EM Algorithm. *J R Stat Soc B-39*:1–38
10. Elgammal A, Duraiswami R, Davis L (2003) Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. *IEEE Trans Pattern Anal Mach Intell* 25(11):1499–1504
11. Figueirido M, Jain A (2002) Unsupervised learning of finite mixture models. *IEEE Trans Pattern Anal Mach Intell* 24(3):381–396
12. Greengard L, Strain J (1991) The Fast Gauss Transform. *SIAM J Sci Comput* 12(1):79–94
13. Johnson D, Sinanovic S (2001) Symmetrizing the Kullback-Leibler Distance. *IEEE Trans Inf Theory* (Submitted)
14. Fu KS, McMurtry GJ (1966) A study of stochastic automata as a model for learning and adaptive controllers. *IEEE Trans Autom Control* 11:379–387
15. Narendra KS, Thathachar MAL (1974) Learning automata: a survey. *IEEE Trans Syst Man Cybern* 4:323–334
16. Kullback S (1968) *Information theory and statistics*. Dover, New York
17. Thathachar MAL, Sastry PS (2002) Varieties of learning automata: an overview. *IEEE Trans Syst Man Cybern* 32(6):711–722
18. Neal R (1992) Bayesian mixture modeling. In: *Proceedings of 11th International workshop on maximum entropy and bayesian methods of statistical analysis*, pp 197–211
19. Nefian AV, Liang L, Pi X, Xiaoxiang L, Mao C, Murphy K (2002) A coupled HMM for audio-visual speech recognition. In: *International conference on acoustics speech and signal processing (CASSP'02)* Orlando FL, pp 13–17
20. Parzen E (1962) On the Estimation of a Probability Density Function and the Mode. *Ann Math Stat* 33:1065–1076
21. Richardson S, Green P (1997) On Bayesian analysis of mixtures with unknown number of components. *J R Stat Soc* 59:731–792
22. Tsetlin ML (1962) On The Behavior of Finite Automata in Random Media. *Autom Remote Control* 22:1210–1219
23. Yang C, Duraiswami R, Gumerov N, Davis L (2003) Improved fast gauss transform and efficient kernel density estimation. In: *Proc IEEE international conference computer vision*