BYY Harmony Learning on Weibull Mixture with Automated Model Selection

Zhijie Ren and Jinwen Ma

Department of Information Science
School of Mathematical Sciences and LMAM
Peking University, Beijing, 100871, China
jwma@math.pku.edu.cn

Abstract. Bayesian Ying-Yang (BYY) harmony learning has provided a new learning mechanism to implement automated model selection on finite mixture during parameter learning with a set of sample data. In this paper, two kinds of BYY harmony learning algorithms, called the batch-way gradient learning algorithm and the simulated annealing learning algorithm, respectively, are proposed for the Weibull mixture modeling based on the maximization of the harmony function on the two different architectures of the BYY learning system related to Weibull mixture such that model selection can be made automatically during the parameter learning on Weibull mixture. The two proposed algorithms are both demonstrated well by the simulation experiments on some typical sample data sets with certain degree of overlap.

Keywords: Bayesian Ying-Yang (BYY) harmony learning, Weibull mixture, Automated model selection, Parameter learning, Simulated annealing.

1 Introduction

Weibull mixture is a leading model in the field of reliability. In fact, there have been several statistical methods to solve the problem of parameter learning or estimation on the Weibull mixture model, such as maximum likelihood estimation, graphics estimation and the EM algorithm. However, these methods usually assume that the number k of components in the mixture is pre-known. If this number is unknown, it can be selected according to the Akaike's information criterion [1] or its extensions [2,3]. However, this conventional approach involves a large computational cost since the entire process of parameter estimation has to be repeated for a number of different choices of k. Since k is just a scale of Weibull mixture model, its selection is essentially a model selection for the Weibull mixture modelling.

The Bayesian Ying-Yang(BYY) harmony learning system and theory, proposed in 1995 in [4] and developed subsequently in [5,6,7], has provided a new efficient tool to solve the compound problem of model selection and parameter learning on the finite mixture model. In fact, by maximizing a harmony function on a certain BYY learning system related to finite mixture, model selection can be made automatically during parameter learning for Gaussian mixture either on a BI-architecture via some gradient-type and fixed-point learning algorithms [8,9,10,11] or on a B-architecture via the BYY annealing learning algorithm [12]. Recently, this BYY harmony learning approach has been also applied to the Poisson mixture modeling [13].

In this paper, we extend the BYY harmony learning mechanism of parameter learning with automated model selection to Weibull mixture. Actually, we consider the two-parameter Weibull model which is by far the most widely used probability distribution for life data analysis. Its probability density function (pdf) takes the following explicit expression (refer to [14]):

$$f(x) = \frac{ax^{a-1}}{b^a} \exp[-($$

2.1 BI-Architecture of BYY Learning System

The BYY system is called to have a BI-architecture if $p(y|x)$ and $q(x|y)$ are both parametric. That is, $p(y|x)$ and $q(x|y)$ are both from a family of probability densities with a parameter θ. We use the following BI-architecture of the BYY system for the Weibull mixture. The inner representation y is discrete, i.e., $y \in \{1, 2, \ldots, k\} \subset R$ and $q(y = j) = \alpha_j \geq 0$ with $\sum_{j=1}^{k} \alpha_j = 1$. $p(x)$ is specified by the empirical density $p_0(x) = \frac{1}{N} \sum_{t=1}^{N} G(x - x_t)$, where $x \in R$, $G(\cdot)$ is a kind of kernel function, and the Yang path is given by the following form:

$$p(y = j|x) = \frac{\alpha_j q(x|\theta_j)}{q(x|\Theta_k)}, \quad q(x|\Theta_k) = \sum_{j=1}^{k} \alpha_j q(x|\theta_j), \tag{4}$$

where $q(x|\theta_j) = q(x|y = j)$, and $\Theta_k = \{\alpha_j, \theta_j\}_{j=1}^{k}$ denote the set of parameters. Putting all these component densities into Eq.(3) and letting the kernel function approach the delta function $\delta(x)$, the harmony functional $H(p\|q)$ is transformed into the following harmony function:

$$J(\Theta_k) = \frac{1}{N} \sum_{t=1}^{N} \sum_{j=1}^{k} \frac{\alpha_j q(x_t|\theta_j)}{\sum_{i=1}^{k} \alpha_i q(x_t|\theta_i)} \ln[\alpha_j q(x_t|\theta_j)], \tag{5}$$

where $q(x_t|\theta_j)$ is the two-parameter Weibull pdf, and $\theta_j = \{a_j, b_j\}$.

2.2 B-Architecture of BYY Learning System

If $q(x|y)$ is parametric and $p(y|x)$ is free to be determined by learning, the BYY system is called to have a B-architecture. For the Weibull mixture, we use the following B-architecture of BYY system. The inner representation y, $q(y = j)$, $p(x)$ are defined as the BI-architecture. And the regularization term z_p is ignored too. Moreover, $p(y|x)$ is a probability distribution that is free to be determined under the general constraints: $p(j|x) \geq 0$, $\sum_{j=1}^{k} p(j|x) = 1$. In the same way, we can get the following harmony function:

$$J(\Theta_k) = \frac{1}{N} \sum_{t=1}^{N} \sum_{j=1}^{k} p(j|x_t) \ln[\alpha_j q(x_t|a_j, b_j)], \tag{6}$$

where $\Theta_k = \{\Theta_1, \Theta_2\}$, $\Theta_1 = \{p(j|x_t)\}_{j=1,t=1}^{k,N}$ and $\Theta_2 = \{\alpha_j, a_j, b_j\}_{j=1}^{k}$.

2.3 Batch-Way Gradient BYY Learning Algorithm

To get rid of the constraints on α_j, we utilize the transformation for each j: $\alpha_j = \exp(\beta_j)/\sum_{i=1}^{k} \exp(\beta_i)$, where $-\infty < \beta_1, \ldots, \beta_k < +\infty$. After such a transformation, the parameters of the harmony function $J(\Theta_k)$ given by Eq.(5) are essentially $\{\beta_j, \theta_j\}_{j=1}^{k}$, $\theta_j = \{a_j, b_j\}$.

By computing the derivatives of $J(\Theta_k)$ with respect to β_j and a_j, b_j, we can obtain the batch-way gradient learning algorithm for the Weibull mixture modeling. Actually, its update rule can be given as follows:

$$\Delta a_j = \frac{\eta}{N}\sum_{t=1}^{N} p(j|x_t)\lambda_j(t)(\frac{1}{a_j} + \ln\frac{x_t}{b_j}(1-(\frac{x_t}{b_j})^{a_j})), \tag{7}$$

$$\Delta b_j = \frac{\eta}{N}\sum_{t=1}^{N} p(j|x_t)\lambda_j(t)(-\frac{a_j}{b_j}(1-(\frac{x_t}{b_j})^{a_j})), \tag{8}$$

$$\Delta \beta_j = \frac{\eta}{N}\sum_{t=1}^{N}\frac{1}{q(x_t|\Theta_k)}\sum_{i=1}^{k}\lambda_i(t)(\delta_{ij}-\alpha_j)U_i(x_t). \tag{9}$$

where $\eta > 0$ is the learning rate which can be selected by experience, $U_j(x) = \alpha_j q(x|\theta_j)$, $\lambda_j(t) = 1 - \sum_{l=1}^{k}(p(l|x_t) - \delta_{jl})\ln U_l(x_t)$, $j = 1, 2, \ldots, k$ and δ_{ij} is the Kronecker function.

2.4 Simulated Annealing Learning Algorithm

Because the maximization of Eq.(6) is a discrete optimization, so it is very easy

On the other hand, we fix Θ_1 and solve the maximum of Θ_2. Also, by the method of Lagrange multipliers, we obtain a series of equations and a unique solution for α_j as follows, for $j = 1, \ldots, k$:

$$\frac{1}{N}\sum_{t=1}^{N} p(j|x_t)[\frac{1}{a_j} + \ln\frac{x_t}{b_j} - (\frac{x_t}{b_j})^{a_j}\ln(\frac{x_t}{b_j})] = 0, \tag{13}$$

$$\frac{1}{N}\sum_{t=1}^{N} p(j|x_t)(-\frac{a_j}{b_j} + \frac{a_j x_t^{a_j}}{b_j^{a_j+1}}) = 0, \tag{14}$$

$$\hat{\alpha}_j = \frac{1}{N}\sum_{t=1}^{N} p(j|x_t). \tag{15}$$

From Eq.(13) and (14), we can obtain an approximative solution of $\hat{a}_j$, $\hat{b}_j$ with the help of some mathematical tools.

From the above derivation, we have already constructed an alternative optimization algorithm for maximizing $L_\lambda(\Theta_k)$. Furthermore, if λ attenuates appropriately a long time, this alternative maximization algorithm anneals to search for the global maximum of $J(\Theta_k)$ and thus the automated model selection with parameter estimation is able to be implemented.

3 Experimental Results

In this section, several simulated experiments are conducted to demonstrate the performance of the batch-way gradient learning algorithm and the simulated annealing learning algorithm for both model selection and parameter estimation on some sample data sets from typical Weibull mixtures. Moreover, we compare the learning efficiency of these two proposed algorithms. For feasibility of the implementation, we only consider the situation of $a > 1$ in our experiments.

3.1 Sample Data Sets and Initialization of the Parameters

We begin with a description of the four sets of sample data used in our experiments. Actually, we conducted 4 Monte Carlo experiments in which samples are drawn from a mixture of four or three variate Weibull distributions, being respectively showed in Fig.(1-4).

In order to clearly observe the samples from different Weibull components in the figures, we represent the samples of each Weibull component with different symbols defined on the upper-right hand corner. That is, the samples of different components are displayed with different symbols on the plane. The x-coordinate of a point is the numerical value of a sample, but the y-coordinates of the points of each component keep the same value, which is given artificially, but changes with the component just for the observation.

The true (or actual) values of the parameters in the Weibull mixture to generate the four sample data sets are given in Table 1, where a_j, b_j, α_j and N_j

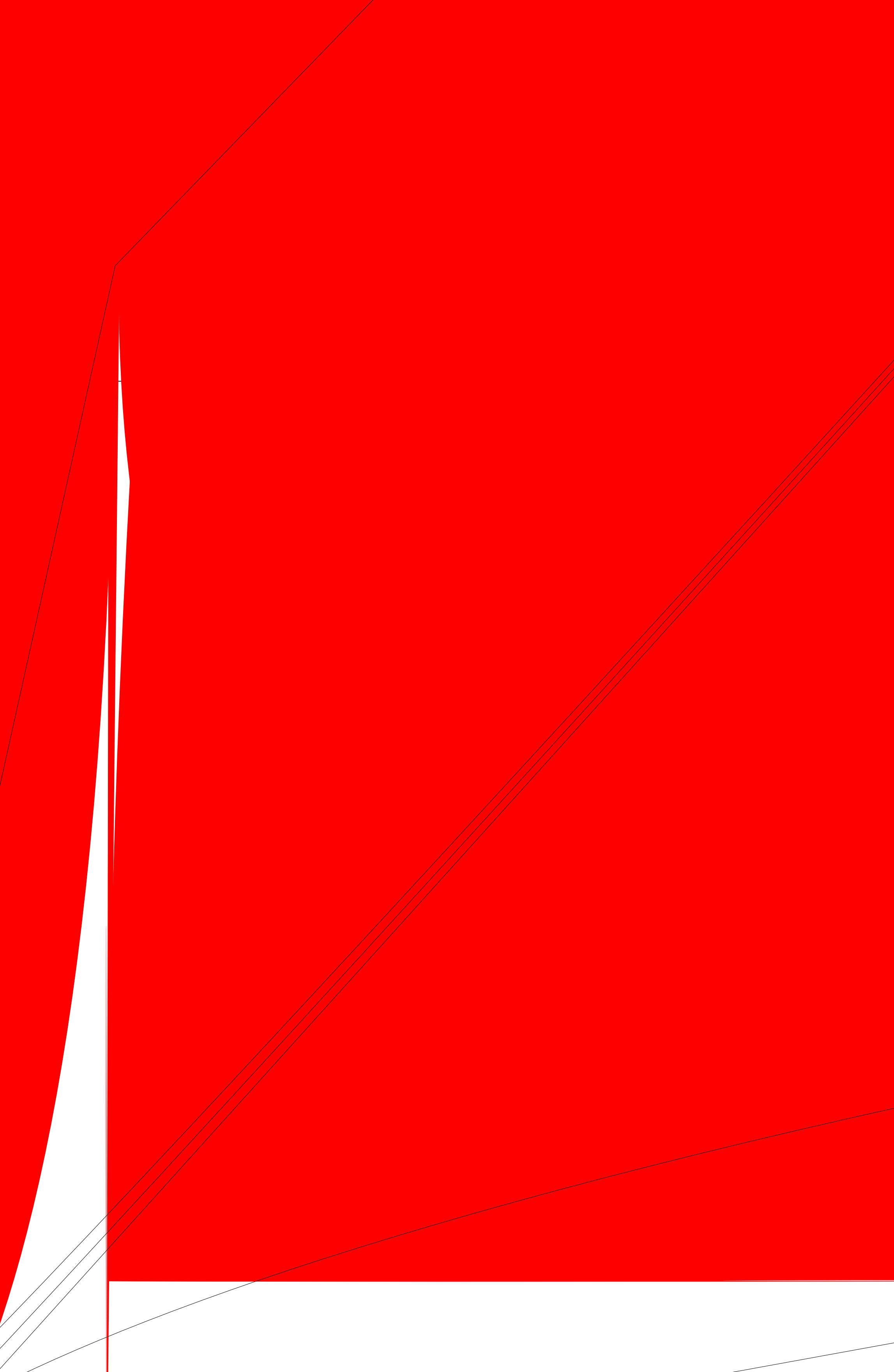

Table 1. The Parameters of the Original Weibull Mixtures to Generate the Four Sample Data sets

The sample set	Weibulls	a_j	b_j	α_j	N_j
$\mathcal{S}_1$	Weibull1	2	2	0.25	300
$(N = 1200)$	Weibull2	4	20	0.35	420
	Weibull3	10	40	0.40	480
$\mathcal{S}_2$	Weibull1	2	2	0.175	35
$(N = 200)$	Weibull2	4	15	0.35	70
	Weibull3	12	35	0.225	45
	Weibull4	15	65	0.25	50
$\mathcal{S}_3$	Weibull1	2	2	0.25	300
$(N = 1200)$	Weibull2	6	20	0.25	300
	Weibull3	10	50	0.25	300
	Weibull4	20	80	0.25	300
$\mathcal{S}_4$	Weibull1	2	2	0.25	300
$(N = 1200)$	Weibull2	4	10	0.25	300
	Weibull3	6	20	0.25	300
	Weibull4	8	35	0.25	300

Table 2. The Degrees of Overlap between any Two Components in Each of the Four Sample Data Sets

The sample set	Overlapping degree of adjacent clusters		
$\mathcal{S}_1(k^* = 3)$	0.0021	0.0214	
$\mathcal{S}_2(k^* = 4)$	0.0038	0.0088	0.0008
$\mathcal{S}_3(k^* = 4)$	0.0001	0.0014	0.0034
$\mathcal{S}_4(k^* = 4)$	0.0168	0.0420	0.0484

annealing algorithm, and the batch-way gradient learning algorithm converges more efficiently when the initial values of these β_j are equal or close. In our simulation experiments, a_j and b_j are initialized in virtue of the Weibull transformation which is deduced in [14]. For the BYY annealing learning algorithm, $\{p(y = j|x_t), j = 1, \ldots, k, t = 1, \ldots, N\}$ can be initialized randomly.

3.2 Simulation Results for Model Selection and Parameter Estimation

Firstly, we implemented the batch-way gradient algorithm on each of the four sample data sets $\mathcal{S}_1$-$\mathcal{S}_4$. The stoping criterion of the algorithm is $|J_{new} - J_{old}| < 10^{-7}$, and all the experiment results are given in Table 3, which are all successful on both model selection and parameter estimation. However, the automated model selection on the sample set $\mathcal{S}_4$ fell into a failure. As the stoping criterion was satisfied, there were five active components in the resulted Weibull mixture, which does not agree with the original Weibull mixture. The reason of this failure

Table 3. The Experimental Results of the Batch-way Gradient Learning Algorithm

The sample set	Weibulls	$\hat{a}_j$	$\hat{b}_j$	$\hat{\alpha}_j$
$\mathcal{S}_1$	Weibull1	1.9482	2.0529	0.2526
$(N = 1200)$	Weibull2	4.7094	20.1738	0.3533
	Weibull3	10.9082	40.2776	0.3941
$\mathcal{S}_2$	Weibull1	2.9695	2.1548	0.1774
$(N = 200)$	Weibull2	4.1712	16.2098	0.3637
	Weibull3	12.3271	34.8763	0.2094
	Weibull4	16.8998	65.1038	0.2495
$\mathcal{S}_3$	Weibull1	1.9780	2.0325	0.2501
$(N = 1200)$	Weibull2	6.5847	20.0977	0.2510
	Weibull3	10.2482	50.1388	0.2494
	Weibull4	20.6729	80.1908	0.2496

Table 4. The Experimental Results of the Simulated Annealing Learning Algorithm

The sample set	Weibulls	$\hat{a}_j$	$\hat{b}_j$	$\hat{\alpha}_j$
$\mathcal{S}_1$	Weibull1	1.9637	2.0358	0.2508
$(N = 1200)$	Weibull2	4.4712	20.0428	0.3478
	Weibull3	10.2418	40.0965	0.4014
$\mathcal{S}_2$	Weibull1	2.9678	2.1428	0.1750
$(N = 200)$	Weibull2	3.9992	16.1519	0.3650
	Weibull3	12.0365	34.8157	0.2100
	Weibull4	16.6213	65.0674	0.2500
$\mathcal{S}_3$	Weibull1	1.9790	2.0312	0.2500
$(N = 1200)$	Weibull2	6.5456	20.0758	0.2500
	Weibull3	10.0101	50.0399	0.2492
	Weibull4	20.3964	80.1490	0.2508
$\mathcal{S}_4$	Weibull1	1.8056	1.9616	0.2633
$(N = 1200)$	Weibull2	5.1810	9.8643	0.2442
	Weibull3	7.3671	20.1510	0.2464
	Weibull4	8.6626	35.5023	0.2461

might be that the degrees of overlap between some adjacent components in $\mathcal{S}_4$ are quite high.

We further implemented the simulated annealing learning algorithm on the four sample data sets. The stoping criterion is $|L_\lambda(\Theta_k^{new}) - L_\lambda(\Theta_k^{old})| < 10^{-7}$. And λ is given by the expression: $\lambda(t) = 1/(a(1 - \exp(-b(t-1))) + c)$, where t denotes the iteration time. In this case, $a = 500$, $b = \ln 10/10000$, $c = 0.5$. The experiment results of the simulated annealing algorithm on the four sample data sets are given in Table 4, which are all successful on both model selection and parameter estimation.

Finally, we compare the performance of the batch-way gradient and simulated annealing learning algorithms through the following specific analysis. We begin to compare the performance of the two algorithms on parameter estimation.

Table 5. Δ_x of the Two Algorithms

The sample data set	learning algorithm	Δ_α	Δ_a	Δ_b
$\mathcal{S}_1$	BWG	0.00041	0.0404	0.00082
$(N = 1200)$	SA	0.00006	0.0148	0.00033
$\mathcal{S}_2$	BWG	0.0065	0.2536	0.0125
$(N = 200)$	SA	0.0063	0.2459	0.0110
$\mathcal{S}_3$	BWG	0.00002	0.0114	0.0003
$(N = 1200)$	SA	0.00002	0.0088	0.00026

Table 6. The Runtime Complexities of the Two Algorithms

The sample set	BWG	SA
$\mathcal{S}_1$	98.9210	40.1560
$\mathcal{S}_2$	56.9680	5.6720
$\mathcal{S}_3$	149.5940	13.5160

According to the experimental results on a sample data set, for each parameter x we can compute $\bar{x}$, the radio of the estimated parameters to the actual parameters and then define $\Delta_x = \|\bar{x} - 1\|^2$ to equivalently describe the mean-square error between the estimated parameter and the actual parameter. Thus, Δ_x can be used as a criterion for evaluating the performance of a learning algorithm on the parameter estimation. The results of Δ_x of the batch-way gradient and simulated annealing algorithms on the first three sample data sets are given in Table 5, where x represents a single parameter in the Weibull mixture, BWG represents the batch-way gradient learning algorithm, and SA represents the simulated annealing learning algorithm.

It can be observed from Table 5 that these two algorithms both perform well on parameter estimation as the number of samples is relatively large. But if the number of samples is small, the mean-square error becomes high for the both algorithms. Moreover, the degree of overlap between the components in a sample data set also plays an important role in the parameter learning. It can be found from Table 5 that the mean-square error is much lower if the degree of overlap is small enough. As showed in Table 5, on the same sample sets, the mean-square errors estimated by the simulated annealing learning algorithm is lower than the ones estimated by the batch-way gradient learning algorithm, which can be also demonstrated by the further experiments.

Secondly, we consider the range of the degree of overlap among the components in a sample data set such that these two proposed learning algorithms can be successful with the sample data set. It was found from the simulation experiments that the simulated annealing learning algorithm generally owns a larger range than the batch-way gradient algorithm does.

Thirdly, we compare the ranges from which the initial k can be selected for these two algorithms. From the simulated experiments, it was found that the selected range of k for the simulated annealing learning algorithm is $[k^*, 2k^*+1]$, which is wider than the range $[k^*, 2k^* - 1]$ for the batch-way gradient learning algorithm.

Fourthly, we compare the runtime costs of the two algorithms. Actually, the runtime complexities which are costed by these two algorithms on the sample data sets $\mathcal{S}_1$-$\mathcal{S}_3$ have been listed in Table 6.

It can be observed from Table 6 that the runtime of the batch-way gradient learning algorithm is always longer than that of the simulated annealing learning algorithm on these sample data sets.

As a result from the above comparisons on the four aspects, the simulated annealing learning algorithm is much better than the batch-way gradient learning algorithm not only on the automated model selection but also on the parameter estimation and the runtime. Therefore, the simulated annealing learning algorithm is more efficient for the Weibull mixture modeling.

4 Conclusions

After introducing the BYY learning system, BI and B-architectures, and the harmony function, we have established two BYY learning algorithms: a batch-way gradient learning algorithm on the BI-architecture and a simulated annealing learning algorithm on the B-architecture, for Weibull mixture with automated model selection. The two algorithms are demonstrated well on the sample sets from Weibull mixtures with certain degrees of overlap. Moreover, we have compared the two algorithms from four aspects and found out that the simulated annealing learning algorithm is more efficient for the Weibull mixture modeling than the batch-way gradient learning algorithm.

Acknowledgements. This work was supported by the Natural Science Foundation of China for grants 60771061 and 60471054.

References

1. Akaike, H.: A New Look at the Statistical Model Identification. IEEE Trans. Automatic Control, AC- 19, 716–723 (1974)
2. Bozdogan, H.: Model Selection and Akaike's Information Criterion: the General Theory and its Analytical Extensions. Psychometrika 52, 345–370 (1978)
3. Scharz, G.: Estimating the Dimension of a Model. The Annals of Statistics 6, 461–464 (1978)
4. Xu, L.: Ying-Yang Machine: a Bayesian-Kullback Scheme for Unified Learnings and New Results on Vector Quantization. In: Proceedings of the 1995 International Conference on Neural Information Processing (ICONIP 1995), vol. 2, pp. 977–988 (1995)
5. Xu, L.: Best Harmony, Unified RPCL and Automated Model Selection for Unsupervised and Supervised Learning on Gaussian Mixtures, Three-layer Nets and ME-RBF-SVM Models. International Journal of Neural Systems 11, 43–69 (2001)
6. Xu, L.: Ying-Yang Learning. In: Arbib, M.A. (ed.) The Handbook of Brain Theory and Neural Networks, 2nd edn., pp. 1231–1237. The MIT Press, Cambridge (2002)

7. Xu, L.: BYY Harmony Learning, Structural RPCL, and Topological Self-organizing on Mixture Models. Neural Networks 15, 1231–1237 (2002)
8. Ma, J., Wang, T., Xu, L.: A Gradient BYY Harmony Learning Rule on Gaussian Mixture with Automated Model Selection. Neurocomputing 56, 481–487 (2004)
9. Ma, J., Gao, B., Wang, Y., et al.: Conjugate and Natural Gradient Rules for BYY Harmony Learning on Gaussian Mixture with Automated Model Selection. International Journal of Pattern Recognition and Artificial Intellegence 19, 701–713 (2005)
10. Ma, J., Wang, L.: BYY Harmony Learning on Finite Mixture: Adaptive Gradient Implementation and a Floating RPCL Mechanism. Neural Processing Lett. 24(1), 19–40 (2006)
11. Ma, J., He, X.: A Fast Fixed-point BYY Harmony Learning Algorithm on Gaussian Mixture with Automated Model Selection. Pattern Recognition Letters 29(6), 701–711 (2008)
12. Ma, J., Liu, J.: The BYY Annealing Learning Algorithm for Gaussian Mixture with Automated Model Selection. Pattern Recognition 40, 2029–2037 (2007)
13. Liu, J., Ma, J.: An Adaptive Gradient BYY Learning Rule for Poisson Mixture with Automated Model Selection. In: Huang, D.-S., Heutte, L., Loog, M. (eds.) ICIC 2007. LNCS, vol. 4681, pp. 1059–1069. Springer, Heidelberg (2007)
14. Robert, B.A.: The New Weibull Handbook, 4th edn. North Palm Beach, Fla. (2000)