ELSEVIER

Neurocomputing 19 (1998) 223–257

NEUROCOMPUTING

RBF nets, mixture experts, and Bayesian Ying–Yang learning[1]

Lei Xu*

Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, People's Republic of China

Received 14 January 1997; accepted 30 November 1997

Abstract

The connections of the *alternative model for mixture of experts* (ME) to the *normalized radial basis function* (NRBF) nets and *extended normalized RBF* (ENRBF) nets are established, and the well-known expectation-maximization (EM) algorithm for maximum likelihood learning is suggested to the two types of RBF nets. This new learning technique determines the parameters of the input layer (including the covariance matrices or so-called receptive fields) and the parameters of the output layer in the RBF nets globally, instead of separately training the input layer by the K-means algorithm and the output layer by the least-squares learning as done in most of the existing RBF learning methods. In addition, coordinated competitive learning (CCL) and adaptive algorithms are proposed to approximate the EM algorithm for considerably speeding up the learning of the original and alternative ME models as well as the NRBF and ENRBF nets. Furthermore, the two ME models are linked to the recent proposed *Bayesian Ying–Yang* (*BYY*) learning system and theory such that not only the architecture of ME and RBF nets is shown to be more preferred than multilayer architecture, but also a new model selection criterion has been obtained to determine the number of experts and basis functions. A number of experiments are made on the prediction of foreign exchange rate and trading investment as well as piecewise nonlinear regression and piecewise line fitting. As shown in these experiments, the EM algorithm for NRBF nets and ENRBF nets obviously outperforms the conventional RBF learning technique, CCL speeds up the learning considerably with only a slight sacrifice on performance accuracy, the adaptive algorithm gives significant improvements on financial predication and trading investment, as well as that the proposed criterion can select the number of basis functions successfully. In addition, the ENRBF net and the alternative ME model are also shown to be able to implement curve fitting and detection.

[1] This work was supported by the HK RGC Earmarked Grants CUHK250/94E and CUHK484/95E and by Ho Sin-Hang Education Endowment Fund for Project HSH 95/02.

* E-mail: lxu@cse.cuhk.edu.hk

0925-2312/98/$19.00
PII S0925-2312(97)00091-X

Keywords: Radial basis function network; Mixture of experts; BayesianYing–Yang learning; Model selection; Coordinated competitive learning; Fast computation; Adaptive algorithm; Financial time series; Curve fitting

1. Introduction

Radial basis function (RBF) network is one of the most popular models that have been studied intensively with a lot of applications and many theoretical results [1–3,8,14–21,36], as well as various new learning techniques developed (e.g., the robust learning proposed by Ref. [22]. The list here is far from completed, and many others can be found in the reference list of the papers [37,34].

For the existing learning techniques for RBF nets, there are still some major problems. *One* is how to select the number of basis functions, which will affect the performance considerably. In the paper [28], via setting up the connections between RBF nets and kernel regression estimators, a number of interesting theoretical results have been obtained for the upper bounds of convergence rate of the approximation error with respect to the number of basis functions, and the upper bounds for the pointwise convergence rate and L_2 convergence rate of the best consistent estimator with respect to both the samples and the number of basis functions. However, these theoretical results are not directly usable in practice. Rival penalized competitive learning (RPCL) is able to automatically select the number of clusters and thus suggested for RBF nets [35,36]. However, although it experimentally works well, RPCL is a heuristic approach and still in lack of theoretical justification. The *second* problem is that the parameters of the input layer are usually trained first by some clustering algorithm and then the parameters of the output layer are usually trained by the least-squares learning [16]. That is, the learning in the two layers is separated with suboptimal results. *Another* problem is that the covariance matrices or so-called receptive field parameters in the input layer are either pre-specified (e.g., ball-shaped scalar variance) or computed after clustering, which are again not well tuned. This also affects the performance [15,37].

This paper aims to solve these problems, through systematically summarizing and developing the basic ideas and several preliminary results obtained by the present author in the conference papers [24,25,28]. In Section 2, we refine the connection of the *alternative model for mixture of experts* (ME) to the *normalized radial basis function* (NRBF) nets and *extended normalized RBF nets* (ENRBF), established first in 1996 [24]. In Section 3, following the basic idea and the preliminary algorithm proposed in [24], we describe that detailed steps on how the EM algorithm developed in [32,33] is used for training the two types of RBF nets such that the parameters of the input layer (including the covariance matrices) and the parameters of the output layer are learned globally [24]. Moreover, coordinated competitive learning and adaptive algorithms [24,25] are also adopted and further improved to approximate the EM algorithm for considerably speeding up the learning in the original and alternative ME models as well as the NRBF and ENRBF nets. In Section 4, a systematic elaboration and some

further extensions on the previous results in [24,25,28] have been made such that the two ME models are further linked to the recent proposed *Bayesian Ying–Yang* learning system and theory such that not only the architecture of ME and RBF net are shown to be more preferred than multilayer architecture, but also new model selection criteria have been obtained to determine the number of experts and basis functions. Furthermore, in Section 5 a number of experiments are made on prediction of foreign exchange rate, trading investment, piecewise nonlinear regressions, piecewise curve fitting and detection, which demonstrate that the proposed algorithms and criterion work well. Finally, we conclude in Section 6.

2. Mixtures experts, the EM algorithm and RBF net learning

2.1. Alternative ME model and the EM algorithm

The original *mixtures of experts* model is based on the following conditional mixture [9,11]:

$$\begin{aligned} &p(z|x,\Theta)=\sum_{j=1}^{k} p(j|x,v)p(z|x,\theta_j), \qquad p(j|x,v)=\mathrm{e}^{\beta_j(x,v)}\Big/\sum_{j=1}^{k}\mathrm{e}^{\beta_j(x,v)},\\ &p(z|x,\theta_j)=(2\pi)^{-d_z/2}|\Gamma_j|^{-1/2}\exp\{-\tfrac{1}{2}[z-f_j(x,w_j)]^{\mathrm{T}}\Gamma_j^{-1}[z-f_j(x,w_j)]\}, \end{aligned} \tag{1}$$

where d_z is the dimension of z, Γ_j is the $d_z\times d_z$ covariance matrix, and $f_j(x,w_j)$ is a vector of the dimension d_z which is the output of the jth expert net. Each expert is represented by $\theta_j=\{w_j,\Gamma_j\}$. Moreover, $p(j|x,v), j=1,\ldots,k$, are the output of the so-called softmax gating net. Finally, we denote $\Theta=\{v,\{\theta_j\}_{j=1}^{k}\}$.

The parameter Θ is estimated by maximum likelihood (ML) learning $L=\sum_{i=1}^{N}\ln p(z_i|x_i,\Theta)$, which can be made by the EM algorithm [11,12]. It is an iterative procedure as follows:

The EM Algorithm – Original

E-step: Fix Θ^{old} and compute $h(j|x_i)=p(j|x_i,z_i)$ by

$$h(j|x_i)=p(j|x_i,z_i)=\frac{p(j|x_i,v^{\mathrm{old}})p(z_i|x_i,\theta_j^{\mathrm{old}})}{\sum_{j=1}^{k}p(j|x_i,v^{\mathrm{old}})p(z_i|x_i,\theta_j^{\mathrm{old}})}. \tag{2}$$

M-step: Find a new estimate $\Theta^{\mathrm{new}}=\{\{\theta_j^{\mathrm{new}}\}_{j=1}^{k},v^{\mathrm{new}}\}$ with

$$\begin{aligned} &\theta_j^{\mathrm{new}}=\arg\max_{\theta_j}Q_j^{e}(\theta_j), \quad Q_j^{e}(\theta_j)=\sum_{i=1}^{N}h(j|x_i)\ln p(z_i|x_i,\theta_j),\ j=1,\ldots,k;\\ &v^{\mathrm{new}}=\operatorname{argmax}_{v}Q(v), \quad Q(v)=\sum_{i=1}^{N}\sum_{j=1}^{k}h(j|x_i)\ln p(j|x,v^{\mathrm{old}}). \end{aligned} \tag{3}$$

The favorable properties of the EM algorithm have also been shown by theoretical analyses [12,31]. One inconvenience of the original *mixtures of experts* architecture is

the nonlinearity of the softmax gating network Eq. (1), which makes the maximization with respect to v of the gating network become nonlinear and unsolvable analytically even for the simplest generalized linear case. An algorithm called *iteratively reweighted least squares* (IRLS) is proposed for the nonlinear optimization [11,12]. However, IRLS is a Newton-type iterative algorithm and thus needs some safeguard measure for convergence.

To overcome this disadvantage of the softmax-based gating net, the following modified gating network is proposed in [32,33]:

$$\begin{aligned} &p(j|x,v) = \alpha_j p(x|v_j)/\sum_{j=1}^{k} \alpha_j p(x|v_j), \\ &p(x|v_j) = a_j(v_j)^{-1} b_j(x)\exp\{c_j(v_j)^{\mathrm{T}} t_j(x)\}, \sum_{j=1}^{k} \alpha_j = 1, \quad \alpha_j \geq 0, \end{aligned} \tag{4}$$

where $v = \{\alpha_j, v_j, j = 1, \ldots, k\}$ and $p(x|v_j)$'s are density functions from the exponential family. The most common example is the Gaussian:

$$p(x|v_j) = G(x, m_j, \Sigma_j) = (2\pi)^{-d_x/2}|\Sigma_j|^{-1/2}\exp\{-\tfrac{1}{2}(x - m_j)^{\mathrm{T}}\Sigma_j^{-1}(x - m_j)\}, \tag{5}$$

where d_x is the dimension of x, and Σ_j is the $d_x \times d_x$ covariance matrix.

In Eq. (4), $p(j|x, v)$ is actually the posterior probability $p(j|x)$ that x is assigned to the partition corresponding to the jth expert net, obtained from Bayes' rule

$$p(j|x, v) = p(j|x) = \alpha_j p(x|v_j)/p(x|v), \qquad p(x|v) = \sum_{j=1}^{k} \alpha_j p(x|v_j). \tag{6}$$

Inserting this $p(j|x,v)$ into the model Eq. (1), we get

$$p(z|x, \Theta) = \sum_{j=1}^{k} \frac{\alpha_j p(x \mid v_j)}{p(x \mid v)} p(z|x, \theta_j), \tag{7}$$

which can be further rewritten as

$$p(x, z) = p(z|x,\Theta)p(x|v) = \sum_{j=1}^{k} \alpha_j p(x|v_j)p(z|x, \theta_j). \tag{8}$$

This suggests an asymmetrical representation for the joint density. We accordingly perform ML estimate based on $L' = \sum_{i=1}^{N} \ln p(z_i, x_i)$ to determine the parameters $\Theta = \{\alpha_j, v_j, \theta_j\}_{j=1}^{k}$ of the gating net and expert nets. This can be made by the following EM algorithm [32,33]:

The EM Algorithm – Alternative

E-step: Fix Θ^{old} and for each pair $\{x_i, z_i\}$ we compute

$$h(j|x_i) = p(j|x_i, z_i) = \frac{\alpha_j^{\mathrm{old}} p(x_i \mid v_j^{\mathrm{old}}) p(z_i|x_i, \theta_j^{\mathrm{old}})}{\sum_{j=1}^{k} \alpha_j^{\mathrm{old}} p(x_i \mid v_j^{\mathrm{old}}) p(z_i \mid x_i, \theta_j^{\mathrm{old}})}. \tag{9}$$

M-step: Find a new estimate $\Theta^{\text{new}} = \{\{\alpha_j^{\text{new}}, v_j^{\text{new}}, \theta_j^{\text{new}}\}_{j=1}^k\}$ with θ_j^{new} given in the same way as in Eq. (3) and with $\alpha_j^{\text{new}}, v_j^{\text{new}}$ given by

$$v_j^{\text{new}} = \frac{\sum_{i=1}^N h(j|x_i) t_j(x_i)}{\sum_{i=1}^N h(j|x_i)}, \qquad \alpha_j^{\text{new}} = \frac{1}{N} \sum_{i=1}^N h(j|x_i), \tag{10}$$

because $Q(v)$ can be further decomposed into

$$Q_j(v_j) = \sum_{i=1}^N h(j|x_i) \ln p(x_i|v_j), \quad j = 1, \ldots, k, \qquad Q^\alpha = \sum_{i=1}^N \sum_{j=1}^k h(j|x_i) \ln \alpha_j, \tag{11}$$

and their maximizations with respect to v_j, α_j become analytically solvable as long as $p(x|v_j)$ is from the exponential family. Particularly, when $p(x|v_j)$ is a Gaussian density, the updating becomes

$$\begin{aligned} m_j^{\text{new}} &= \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i) x_i, \\ \Sigma_j^{\text{new}} &= \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)[x_i - m_j^{\text{new}}][x_i - m_j^{\text{new}}]^{\text{T}}. \end{aligned} \tag{12}$$

2.2. Normalized RBF nets and the EM algorithm

The normalized RBF (NRBF) nets can be summarized by the following general form [16,17]:

$$f_k(x) = \frac{\sum_{j=1}^k w_j \phi([x - m_j]^{\text{T}} \Sigma_j^{-1} [x - m_j])}{\sum_{j=1}^k \phi([x - m_j]^{\text{T}} \Sigma_j^{-1} [x - m_j])}, \tag{13}$$

where $\phi(r^2)$ is a prespecified basis function satisfying certain weak conditions. The most common choice is the Gaussian function $\phi(r^2) = \mathrm{e}^{-r^2}$, but a number of alternatives can also be used (e.g., several choices are listed in [19]). m_j is called the center vector and w_j is a weight vector. Σ_j is a $d \times d$ positive matrix (called a covariance matrix or the receptive field of the basis function), which defines a range for which an input $x \in R^d$ can cause a sufficiently large output.

For the existing approaches [16,2,3], the learning on the parameters in Eq. (13) is separated into two steps:

(1) *Determining the parameters in the input layer*. The centers m_j, $j = 1, \ldots, k$, are determined only based on the input samples $\mathscr{D}_x = \{x_i\}_{i=1}^N$ via some clustering technique, for example, the K-means algorithm [16]. That is, the cluster centers are usually used as m_j, $j = 1, \ldots, k$, with the k heuristically pre-fixed, while the covariance matrix Σ_j is either externally prespecified at some value or computed from the resulted cluster centered at m_j.

(2) *Determining the parameters in the output layer*. After the parameters are settled, for each samples x_i, the following value can be directly computed:

$$\phi_{i,k} = \frac{\phi([x_i - m_j]^{\text{T}} \Sigma_j^{-1} [x_i - m_j])}{\sum_{j=1}^k \phi([x_i - m_j]^{\text{T}} \Sigma_j^{-1} [x_i - m_j])}.$$

As a result, the parameter vector w_j can be determined by the least-squares method based on the paired data set $\mathcal{D}_{x,z} = \{x_i, z_i\}_{i=1}^N$. The procedure can be implemented either in the batch way or by the adaptive least-squares algorithm.

The extended NRBF (ENRBF) net is a modification of the NRBF net Eq. (13) by replacing w_j with a linear vector function $W_j^T x_i + c_j$. That is, we have [10]

$$f_k(x) = \frac{\sum_{j=1}^k (W_j^T x + c_j)\phi([x - m_j]^T \Sigma_j^{-1}[x - m_j])}{\sum_{j=1}^k \phi([x - m_j]^T \Sigma_j^{-1}[x - m_j])}, \tag{14}$$

where W_j is a parameter matrix.

Its learning is basically the same as the above one for the normalized RBF nets, except that the least squares learning is a little bit more complicated for the output layer.

Next, we introduce and further refine a connection, established by the present author first in 1996 [24]. We consider that each expert network in Eq. (1) is simply a Gaussian

$$p(z|x, \theta_j) = G(z, c_j, \Gamma_j), \tag{15}$$

and the gating net is given by Eq. (4) with

$$\alpha_j = \frac{\sqrt{|\Sigma_j|}}{\sum_{j=1}^k \sqrt{|\Sigma_j|}}, \qquad p(j|x,v) = p(j|x) = \frac{e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}{\sum_{j=1}^k e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}. \tag{16}$$

By taking the conditional expectation $E(z|x)$ based on both sides of Eq. (1), we have

$$\text{out}(x) = E(z|x, \Theta) = \sum_{j=1}^k c_j \frac{e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}{\sum_{j=1}^k e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}. \tag{17}$$

We can easily find that it is just a NRBF net with Gaussian basis functions as $\phi(.)$.

If Eq. (15) is further extended into a linear regression form:

$$p(z|x, \theta_j) = G(z, W_j^T x + c_j, \Gamma_j), \tag{18}$$

we have that Eq. (17) will be modified into

$$\text{out}(x) = E(z|x, \Theta) = \sum_{j=1}^k (W_j^T x + c_j) \frac{e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}{\sum_{j=1}^k e^{-0.5(x-m_j)^T \Sigma_j^{-1}(x-m_j)}}, \tag{19}$$

which is just an ENRBF net with Gaussian basis functions as $\phi(.)$.

By defining $p(z|x, \theta_j)$ to be another density function, similarly we obtain another basis function.

Therefore, the *EM Algorithm – Alternative* can be used for training the above NRBF and ENRBF nets, resulting in:

The EM Algorithm – RBF

E-Step: Fix Θ^{old}, get

$$h(j|x_i) = \frac{e^{-0.5(x_i - m_j^{\text{old}})^T (\Sigma_j^{\text{old}})^{-1}(x_i - m_j^{\text{old}})} G(z_i, r_j^{\text{old}}, \Gamma_j^{\text{old}})}{\sum_{j=1}^k e^{-0.5(x_i - m_j^{\text{old}})^T (\Sigma_j^{\text{old}})^{-1}(x_i - m_j^{\text{old}})} G(z_i, r_j^{\text{old}}, \Gamma_j^{\text{old}})}, \tag{20}$$

where $r_j^{\text{old}} = c_j^{\text{old}}$ for an NRBF net and $(W_j^{\text{old}})^T x_i + c_j^{\text{old}}$ for an ENRBF net.

M-Step: First, update

$$m_j^{\text{new}} = \frac{\sum_{i=1}^N h(j|x_i)x_i}{\sum_{i=1}^N h(j|x_i)},$$
$$\Sigma_j^{\text{new}} = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)[x_i - m_j^{\text{new}}][x_i - m_j^{\text{new}}]^{\text{T}}, \tag{21}$$

and then for an NRBF net, update

$$c_j^{\text{new}} = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)z_i,$$
$$\Gamma_j^{\text{new}} = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)(z_i - c_j^{\text{new}})(z_i - c_j^{\text{new}})^{\text{T}}, \tag{22}$$

and for an ENRBF net, update

$$Ez_j = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)z_i,$$
$$R_{xz} = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)[x_i - m_j^{\text{new}}][z_i - Ez_j]^{\text{T}},$$
$$W_j^{\text{new}} = [\Sigma_j^{\text{new}}]^{-1} R_{xz}, \qquad c_j^{\text{new}} = Ez_j - (W_j^{\text{new}})^{\text{T}} m_j^{\text{new}},$$
$$\Gamma_j^{\text{new}} = \frac{1}{\sum_{i=1}^N h(j|x_i)} \sum_{i=1}^N h(j|x_i)[z_i - (W_j^{\text{new}})^{\text{T}}x_i - c_j^{\text{new}}][z_i - (W_j^{\text{new}})^{\text{T}}x_i - c_j^{\text{new}}]^{\text{T}}. \tag{23}$$

Particularly, when $\Gamma_j = \gamma_j^2 I$, which is widely assumed in the literature, the updating on Γ_j can be simply

$$\gamma_j^2 = \frac{1}{N} \sum_{j=1}^k \sum_{i=1}^N h(j\,|\,x_i)\|z_i - r_j^{\text{new}}\|^2,$$
$$r_j^{\text{old}} = \begin{cases} c_j^{\text{old}} & \text{for an NRBF net,} \\ (W_j^{\text{old}})^{\text{T}}x_i + c_j^{\text{old}} & \text{for an ENRBF net.} \end{cases} \tag{24}$$

The above introduced algorithm can be regarded as a detailed extension of the preliminary form of EM algorithm for RBF nets, proposed first in [24].

3. Coordinated competitive learning and adaptive algorithms

3.1. Coordinated competitive learning

For the EM algorithms given in the previous section, we need to compute a posterior probability $h(j|x_i)$ which indicates the probability of assigning the mapping task of the pair $x_i \to z_i$ to the jth expert.

Alternatively, by adopting the basic ideas suggested in [25], this soft assignment can be approximated by a winner-take-all (WTA) competition according to Bayesian decision via replacing $h(j|x_i)$ in the M-step of *all the algorithms* with the following hard-cut indicator:

$$I(j|x_i) = \begin{cases} 1 & \text{if } j = \text{argmin}_r - \log h(r|x_i), \\ 0 & \text{otherwsie.} \end{cases} \tag{25}$$

Although some accuracy may be lost in performance (as will be shown later by experiments, this loss can usually be ignored), it may bring two advantages. *First*, it can speed up considerably, not only because the multiplication with $h(j|x_i)$ is not needed and the summation of the terms with $I(j|x_i) = 0$ can be saved, but also because the computing cost for $h(j|x_i)$ can be considerably reduced since the computing for $I(j|x_i)$ is significantly simplified, as can be further observed more clearly later. *Second*, it can speed up the convergence. *After running for a certain period, the EM learning usually slows down considerably, sometimes it will become very slow. In this case, we can switch* $h(j|x_i)$ *into* $I(j|x_i)$, *which can obviously speed up the convergence*, without affecting the performance too much.

To get some insights, let us first to consider the *EM Algorithm – Original* with $h(j|x_i)$ given by Eq. (2) and $p(j|x, v)$ given by Eq. (1). In this case, Eq. (25) becomes

$$I(j|x_i) = \begin{cases} 1 & \text{if } j = \arg\min_r\{ -2\beta_r(x,v) + \log|\Gamma_r| + e_r^2(x_i)\}, \\ & \quad e_j^2(x_i) = [z_i - f(x_i, W_j)]^{\mathrm{T}} \Gamma_j^{-1} [z_i - f(x_i, W_j)], \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

This competition consists of two coordinated parts. One is $\log|\Gamma_r| + e_r^2(x_i)$, which is contributed by the expert nets, and the other is $-2\beta_r(x,v)$, contributed by the gating net. So, we call it *coordinated competition* and the corresponding learning the *coordinated competitive learning* (*CCL*).

As shown in [25], there are several variants for further simplifying Eq. (26). For example, the term $\log|\Gamma_r|$ can be omitted in the case $\Gamma_j = \Gamma$. Also, we can even ignore the term $-2\beta_r(x,v)$ and have only

$$j = \arg\min_r e_r^2(x_i) \quad \text{or even} \quad j = \arg\min_r \|z_i - f(x_i, w_r)\|^2, \text{ for } \Gamma = \gamma^2 I, \tag{27}$$

where the competition is only among the experts and the gating net simply follows the result of this competition.

Similarly, considering the *EM Algorithm – Alternative* with $h(j|x_i)$ given by Eq. (9), we have

$$I(j|x_i) = \begin{cases} 1 & \text{if } j = \text{argmin}_r\{ -2\log\alpha_r + \log|\Sigma_r| + (x_i - m_j)^{\mathrm{T}} \Sigma_j^{-1} (x_i - m_j) \\ & \quad + \log|\Gamma_r| + e_r^2(x_i)\}, \\ 0 & \text{otherwise,} \end{cases} \tag{28}$$

which can also be simplified into several variants, e.g.,

$$\begin{aligned} & j = \arg\min_r\{(x_i - m_j)^{\mathrm{T}} \Sigma_j^{-1} (x_i - m_j) + e_r^2(x_i)\}, \quad \text{or even} \\ & j = \arg\min_r\{\|x_i - m_r\|^2 + \eta\|z_i - f(x_i, w_r)\|^2\}, \end{aligned} \tag{29}$$

where $\eta = \sigma_j^2/\gamma_j^2$. That is, the competition is made only among whether x_i is close to m_j and whether the pair x_i,z_i is close to the regression model of the jth expert. Here, we can clearly see that how the mismatch between the data and model versus the variances of two models affects the competition. Furthermore, we can ignore the gating part and get Eq. (27) again.

For the *EM Algorithm–RBF*, we get the corresponding $I(j|x_i)$ by simply discarding the term $-2\log\alpha_r$ in Eq. (28). To save the computing on Γ_j in the *EM Algorithm-RBF*, we can also ignore the competition among the expert nets, resulting in

$$\begin{aligned} &j = \arg\min_r (x_i - m_j)^{\mathrm{T}} \Sigma_j^{-1} (x_i - m_j) \quad \text{or even} \\ &j = \arg\min_r \|x_i - m_j\|^2, \quad \text{for } \Sigma_j = \sigma^2 I. \end{aligned} \tag{30}$$

3.2. Adaptive algorithms

For the practical problems that samples come one by one, such as prediction of foreign exchange rate, we need to adaptively track the temporal relationship in data. For those learnings of parameter optimization of type $(1/N)\sum_{i=1}^{N} e_i(\theta)$ with $e_i(\theta)$ related to only the current sample, we usually use stochastic approximation to modify the batch learning into the following adaptive one:

$$\theta^{\text{new}} = \theta^{\text{old}} - \eta \left.\frac{\partial e_i(\theta)}{\partial \theta}\right|_{\theta=\theta^{\text{old}}}. \tag{31}$$

The convergence of this stochastic approximation algorithm depends on the form of $e_i(\theta)$ and the suitably selected learning rate $\eta > 0$. There are some existing theories for guiding the convergence analyses. In this paper, we will not go deep into theoretical aspects. Instead, we only discuss how to get adaptive learning algorithms for training the previous mixture of experts, NRBF and ENRBF nets.

The key consists of two points. One is to use the following adaptive learning rate:

$$\eta_{j,i} = \eta_0 h(j|x_i)/\alpha_j \quad \text{or} \quad \eta_{j,i} = \eta_0 I(j|x_i)/\alpha_j, \tag{32}$$

with η_0 being a prefixed learning rate and $h(j|x_i)$ or $I(j|x_i)$ modifying this rate adaptively. The other is to simply omit whatever computation that corresponds to $I(j|x_i) = 0$. The specific details are summarized into the following two major points:

1. For learning the network parameters, Eq. (31) is directly used. For example, to learn parameters W_j in $p(z|x,\theta_j) = G(z,f_j(x,W_j),\Gamma_j)$, the updating is made by

$$w_{j^*}^{\text{new}} = w_{j^*}^{\text{old}} - \eta_{j,i} \frac{\partial e_{j^*}^2(x_i)}{\partial w_{j^*}}, \qquad e_j^2(x_i) = [z_i - f(x_i,W_j)]^{\mathrm{T}} \Gamma_j^{-1} [z_i - f(x_i,W_j)]. \tag{33}$$

2. For learning the density parameters m_j, Σ_j and α_j, the updating is made in the form of

$$\begin{aligned} m_j^{\text{new}} &= m_j^{\text{old}} + \eta_{j,i}(x_i - m_j^{\text{old}}), \\ \Sigma_j^{\text{new}} &= (1-\eta_{j,i})\Sigma_j^{\text{old}} + \eta_{j,i}(x_i - m_j^{\text{old}})(x_i - m_j^{\text{old}})^{\text{T}}, \\ n_j^{\text{new}} &= (1-\eta_{j,i})n_j^{\text{old}} + \eta_{j,i}, \qquad \alpha_j^{\text{new}} = n_j^{\text{new}} / \sum_{j=1}^{k} n_j^{\text{new}}. \end{aligned} \tag{34}$$

As an example, we can turn the *EM Algorithm–RBF* into its adaptive variant as follows:

The Adaptive EM Algorithm – RBF

E-Step: Fix Θ^{old}, to get $h(j|x_i)$ in the same way as in the *EM Algorithm–RBF* or to get $I(j|x_i)$ as discussed in the previous subsection, e.g., by Eq. (29). Then let $\eta_{j,i}$ be given by Eq. (32).

M-Step: First, update

$$\begin{aligned} m_j^{\text{new}} &= m_j^{\text{old}} + \eta_{j,i}(x_i - m_j^{\text{old}}), \\ \Sigma_j^{\text{new}} &= (1-\eta_{j,i})\Sigma_j^{\text{old}} + \eta_{j,i}(x_i - m_j^{\text{old}})(x_i - m_j^{\text{old}})^{\text{T}}. \end{aligned} \tag{35}$$

Then, for a NRBF net, update

$$\begin{aligned} c_j^{\text{new}} &= c_j^{\text{old}} + \eta_{j,i}(z_i - c_j^{\text{old}}), \\ \Gamma_j^{\text{new}} &= (1-\eta_{j,i})\Gamma_j^{\text{old}} + \eta_{j,i}(z_i - c_j^{\text{new}})(z_i - c_j^{\text{new}})^{\text{T}}; \end{aligned} \tag{36}$$

for an ENRBF net, update

$$\begin{aligned} Ez_j^{\text{new}} &= Ez_j^{\text{old}} + \eta_{j,i}(z_i - Ez_j^{\text{old}}), \qquad c_j^{\text{new}} = Ez_j^{\text{new}} - w_j^{\text{old T}} m_j^{\text{old}}; \\ \Gamma_j^{\text{new}} &= (1-\eta_{j,i})\Gamma_j^{\text{old}} + \eta_{j,i}(z_i - w_j^{\text{old T}} x_i - c_j^{\text{new}})(z_i - w_j^{\text{old T}} x_i - c_j^{\text{new}})^{\text{T}}, \\ w_j^{\text{new}} &= w_j^{\text{old}} + \eta_{j,i}(z_i - w_j^{\text{new T}} x_i - c_j^{\text{new}})x_i^{\text{T}}. \end{aligned} \tag{37}$$

Usually, the initialization of parameters will affect the performance of adaptive algorithms. For the practical problems like financial data prediction, we usually have quite limited number of sample points, thus using an adaptive algorithm alone cannot bring any real advantage. *In this case, we suggest to first use one of the previous batch algorithms on a training set to get a solution as an initialization, and then to use the above adaptive algorithm to keep tracing the changes of data on the testing data via adaptation once a new data point is available.* As will be shown later by the

experimental results, this use of adaptive algorithm really works well for the problems of strongly nonstationary signals.

4. BYY learning, mixture experts, and basis function number

A unified statistical learning approach called *Bayesian Ying–Yang* (*BYY*) *system and theory* has been developed by the present author in recent years [23–29]. It provides new theories for *unsupervised pattern recognition and cluster analysis*, *factorial encoding*, *data dimension reduction*, *and independent component analysis*, as well as for *supervised classification and regression*, such that not only several existing popular supervised and unsupervised learning approaches, (e.g., finite mixture with the EM algorithm, K-means clustering algorithm, one hidden layer Helmholtz machine, principal component analysis plus various extensions, Informax and minimum mutual information approaches, etc.), are unified as special cases with new insights and results, but also a quite number of new variants and models are obtained. Especially, it provides a new general theory for model selection and regularization, which is easy to implement and suitable for the cases of finite number of training samples, with solution for several hard open problems (e.g., the number of clusters in the K-means clustering algorithms, the dimension of subspace, etc. [29,26]). Furthermore, as a whole, the BYY learning system and theory functions as a general framework and theory for supervised learning, semi-unsupervised learning, and unsupervised learning on *parameter learning*, *regularization*, *structural scale or complexity selection*, *architecture comparison and data smoothing*.

In this section, after briefly introducing the BYY learning system and theory, we use the theory to interpret why the architecture of mixture experts and RBF nets is more favorable than that of multilayer nets and to obtain criteria for selecting the appropriate number of experts and basis functions.

4.1. BYY learning system and theory

4.1.1. Basic BYY learning system and theory

As shown in Fig. 1, the BYY system consists of seven components. The first four components form the core, which itself functions as a general framework for unsupervised learning. The surrounding other three components are added for the purposes of supervised learning.

First, we need to understand the basic idea of the core. As shown in [26], unsupervised perception tasks can be summarized into the problem of estimating the joint distribution $p(x,y)$ of the observable pattern x in the visible space X and its representation pattern y in an invisible space Y. In the Bayesian framework, we have two complementary representations $p(x,y) = p(y|x)p(x)$ and $p(x,y) = p(x|y)p(y)$. We use two sets of models $M_1 = \{M_{y|x}, M_x\}$ and $M_2 = \{M_{x|y}, M_y\}$ to implement each of the two representations:

$$p_{M_1}(x,y) = p_{M_{y|x}}(y|x)p_{M_x}(x), \qquad p_{M_2}(x,y) = p_{M_{x|y}}(x|y)p_{M_y}(y). \tag{38}$$

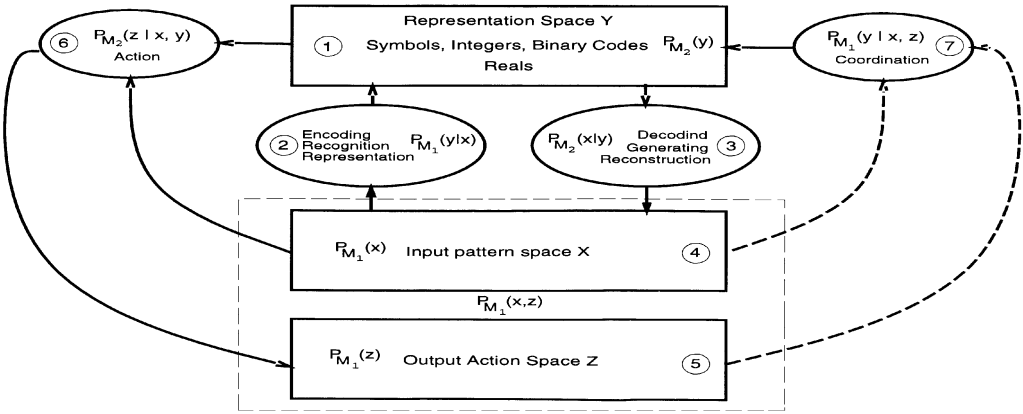


Fig. 1. The Bayesian YING–YANG System.

We call M_x a Yang/(visible) model, which describes $p(x)$ in the visible domain X, and M_y a Ying[2]/(invisible) model which describes $p(y)$ in the invisible domain Y. Also, we call the passage $M_{y|x}$ for the flow $x \to y$ a *Yang*/(male) passage since it performs the task of transferring a pattern/(a real body) into a code/(a seed). We call a passage $M_{x|y}$ for the flow $y \to x$ a *Ying*/(female) passage since it performs the task of generating a pattern/(a real body) from a code/(a seed). Together, we have a *YANG* machine M_1 to implement $p_{M_1}(x,y)$ and a *YING* machine M_2 to implement $p_{M_2}(x,y)$. A pair of YING–YANG machines is called a YING–YANG pair or a Bayesian YING–YANG system. Such a formalization compliments to a famous Chinese ancient philosophy that *every entity in the universe involves the interaction between YING and YANG.*

The task of specifying a Ying–Yang system is called *learning* in a broad sense, which consists of the following four levels of specifications:

1. Based on the nature of a specific task, the *representation domain Y and its complexity k* are designed. For example, in this paper it is a binary vector $y = [y^{(1)}, \ldots, y^{(k)}]^{\mathrm{T}}$, $y^{(j)} \in \{0,1\}$.
2. Based on the given set of training samples, some previous knowledge, assumption and heuristics, *architecture design* is made on specifying the architectures of components. First, with a given set $D_x = \{x_i\}_{i=1}^N$ from an original density $p^o(x)$,$p_{M_x}(x)$ is fixed on some parametric (e.g., finite mixture or gaussian mixture in particular) or nonparametric empirical density estimation of $p^o(x)$, e.g., $p_{M_x}(x) = p_{h_x}(x)$ given by a kernel estimate [6]

$$p_{h_x}(x) = \frac{1}{N}\sum_{i=1}^{N} K_{h_x}(x - x_i), \qquad K_{h_x}(r) = \frac{1}{h^{d_x}} K\left(\frac{r}{h}\right), \tag{39}$$

[2] It should be "Yin" in the Mainland Chinese spelling system. However, I prefer to use "Ying" for the beauty of symmetry.

with a prefixed kernel function $K(.)$ and a prefixed smoothing parameter h_x. Next, each $p_{M_a}(a), a \in \{x|y, y|x, y\}$ can be designed in two ways. One is called *Free*. It implies a totally unspecified density or probability function in the form $p(a)$ without any constraint. Thus, it is free to change such that it can be specified through other components indirectly. The other is called *parameterized architecture*. It means that $p_{M_a}(a)$, $a \in \{x|y, y|x, y\}$ is either a simple parametric density, e.g., a Gaussian $p_{M_{x|y}}(x|y) = G(x, m_{x|y}, \Sigma_{x|y})$, or a compounded parametric density with some of its parameters defined by a complicated function with a given parametric architecture consisting of a number of elementary units that are organized in a given structure [26,27]. Moreover, when $p_{M_{e|v}}(x|y)$ is parametric, the rest two $p_{M_a}(a)$, $a \in \{y|x, y\}$ can be either free or parametric; however when $p_{M_{x|y}}(x|y)$ is free, the rest two must be parametric to avoid the useless under-determined cases.

3. We also need to select the above complexity k, as well as other scale or complexity parameters for a complicated architecture [26,27]. The task is called *structural scale selection* or *model selection*.
4. After the above three levels of specifications, the unspecified part for each component $p_{M_a}(a), a \in \{x|y, y|x, y\}$ is a set θ_a of parameters in certain domains. Putting them together, we get the parameter set $\Theta = \{\theta_{x|y}, \theta_{y|x}, \theta_y\}$, which we called *parameter learning*.

Our basic theory is that the specification of a Ying–Yang pair in the above four levels best enhances the so-called *Ying–Yang harmony or marry*, through minimizing a harmony measure called *separation functional*:

$$\begin{aligned} &F_s(M_1, M_2) = F_s(p_{M_{y|x}}(y|x)p_{M_x}(x), p_{M_{x|y}}(x|y)p_{M_y}(y)) \geq 0, \\ &F_s(M_1, M_2) = 0 \text{ if and only if } p_{M_{y|x}}(y|x)p_{M_x}(x) = p_{M_{x|y}}(x|y)p_{M_y}(y), \end{aligned} \tag{40}$$

which describes the harmonic degree of the Ying–Yang pair. Such a learning theory is called *Bayesian Ying–Yang* (*BYY*) *learning theory*. Three categories of separation functionals have been suggested in [28,26]. In this paper, we only consider a special case – the well-known Kullback divergence as follows:[3]

$$KL(M_1, M_2) = \int_x \sum_y p_{M_{y|x}}(y|x)p_{M_x}(x) \ln \frac{p_{M_{y|x}}(y|x)p_{M_x}(x)}{p_{M_{x|y}}(x|y)p_{M_y}(y)} \mathrm{d}x. \tag{41}$$

In this special case, the BYY learning is called Bayesian–Kullback YING–YANG (BKYY) learning. Being different from the large number law based modern statistical learning principle, this BYY learning emphasizes the best relative harmony of the finite sample based $p_{M_x}(x)$ and the structurally constrained learning system of $p_{M_{y|x}}(y|x)$, $p_{M_{x|y}}(x|y)$ and $p_{M_y}(y)$, instead of considering the derivation from an absolute truth or standard. As the number of samples becomes large, Eq. (39) will obey the large number law and the BYY learning will be consistent to the modern statistical learning principle.

[3] In general, $\sum_y$ should be $\int_y$, here $\sum_y$ is used because y is a binary vector in this paper.

4.1.2. BYY learning system and theory

Next, we consider the complete system in Fig. 1 with the last three components joined in. One is the *output action space* $z \in Z$ with its distribution $p_{M_z}(z)$. The other is the *action terminal* (AT) described by a distribution $p_{M_{z|x,y}}(z|x,y)$ for the mapping $x \to z$ that is modulated by the internal representation y. The another one is the *coordination terminal* (CT) described by a distribution $p_{M_{y|x,z}}(y|x,z)$, which lets the invisible representation Y be in coordination with the two visible spaces X, Z. Here, with respect to the invisible space Y, the model $p_{M^L_1}(x,z) = p_{M_{z|x}}(z|x)p_{M_x}(x)$ for the joint X, Z forms a large *Yang model*; while with respect to the visible output action space Z, the model $p_{M^L_2}(x,y) = p_{M_2}(x,y) = p_{M_{x|y}}(x|y)p_{M_y}(y)$ for the joint X, Y forms a large *Ying model*. The action terminal $p_{M_{z|x,y}}(z|x,y)$ is the passage from the large Ying space to the Yang space Z. The coordination terminal $p_{M_{y|x,z}}(y|x,z)$ is the passage from the large Yang space to the Ying space Y. Thus, we have another YING–YANG pair

$$\begin{aligned} p_{M^L_1}(x,y,z) &= p_{M_{y|x,z}}(y|z,x)p_{M^L_1}(z,x) = p_{M_{y|x,z}}(y|x,z)p_{M_{z|x}}(z|x)p_{M_x}(x), \\ p_{M^L_2}(x,y,z) &= p_{M_{z|x,y}}(z|x,y)p_{M^L_2}(x,y) = p_{M_{z|x,y}}(z|x,y)p_{M_{x|y}}(x|y)p_{M_y}(y), \end{aligned} \tag{42}$$

together with the old Ying–Yang pair in Eq. (38) to form an enlarged Ying–Yang system.

Therefore, in addition to specify the old Ying–Yang pair, we need also to specify the new Ying–Yang pair. First, $p_{M^L_1}(z,x)$ is specified via $D_{x,z} = \{x_i, z_i\}_{i=1}^N$ such that $p_{M_x}(x) = p_{h_x}(x)$ is still the same as before and $p_{M_{z|x}}(z|x) = p_{h_z}(z|x)$ for a pair $(x,z') \in D_{x,z}$:

$$p_{h_z}(z|x) = \begin{cases} \delta_d(z - z') & z \text{ is discrete}, \\ \dfrac{1}{h_z^{d_z}} K\left(\dfrac{z - z'}{h_z}\right) & z \text{ is real}, \end{cases} \qquad \delta_d(z) = \begin{cases} 1 & \text{for } z = 0, \\ 0 & \text{for } z \neq 0. \end{cases} \tag{43}$$

So, only two new components $p_{M_{y|x,z}}(y|x,z)$ and $p_{M_{z|x,y}}(z|x,y)$ need to join the previously discussed core for being specified through *architecture design*, *structural scale selection*, and *parameter learning*.

Again, our basic theory is that all the specifications should best enhance the *Ying–Yang harmony* for both the Ying–Yang pairs, through minimizing

$$\begin{aligned} F_{\text{two}}(M_1, M_2) &= F^L_{\text{s}}(M_1, M_2) + F_{\text{s}}(M_1, M_2), \\ F_{\text{s}}(M_1, M_2) &= F_{\text{s}}(p_{M_{y|x}}(y|x)p_{M_x}(x), p_{M_{x|y}}(x|y)\, p_{M_y}(y)), \\ F^L_{\text{s}}(M_1, M_2) &= F_{\text{s}}(p_{M_{y|x,z}}(y|x,z)p_{M^L_1}(z,x), p_{M_{z|x,y}}(z|x,y)p_{M^L_2}(x,y)). \end{aligned} \tag{44}$$

This $\min_{M_1, M_2} F_{\text{two}}(M_1, M_2)$ can be implemented by an *alternative minimization* iterative procedure:

$$\begin{aligned} &\text{Step 1:} \quad Fix\ M_2 = M_2^{\text{old}},\ get\ M_1^{\text{new}} = \arg\min_{M_1} F_{\text{two}}(M_1, M_2), \\ &\text{Step 2:} \quad Fix\ M_1 = M_1^{\text{old}},\ get M_2^{\text{new}} = \arg\min_{M_2} F_{\text{two}}(M_1, M_2), \end{aligned} \tag{45}$$

which guarantees to reduce $F_{\text{two}}(M_1, M_2)$ until it converges to a local minimum.

In the case that the Kullback divergence is used, it becomes

$$KL_{\text{two}}(M_1, M_2) = KL^L(M_1, M_2) + KL(M_1, M_2),$$
$$KL^L(M_1, M_2) = \int_{x,z} \sum_y p_{M_{y|x,z}}(y|x,z) p_{M_1^L}(z,x) \ln \frac{p_{M_{y|x,z}}(y|x,z) p_{M_1^L}(z,x)}{p_{M_{z|x,y}}(z|x,y) p_{M_2^L}(x,y)} \, dx \, dz, \tag{46}$$

with $KL(M_1, M_2)$ still given by Eq. (41). It can be noticed that Eq. (46) will degenerate into Eq. (41) when $z = x$ which makes $KL^L(M_1, M_2) = KL(M_1, M_2)$ and $KL_{\text{two}}(M_1, M_2) = 2KL(M_1, M_2)$.

As shown in [26,27], $\min_{M_1, M_2} F_{\text{two}}(M_1, M_2)$ actually can provide theoretical guides on all the aspects of specifying a Ying–Yang system, including *parameter learning*, *regularization*, *structural scale or complexity selection*, *architecture comparison and data smoothing*. In this paper, given a training set $D_{x,z} = \{x_i, z_i\}_{i=1}^N$, we only consider the cases that the architecture has been already pre-designed and that the remaining unspecified parts are the model scale k (i.e., the bits of y) and a set Θ_k of all the parameters in the Ying–Yang system with the scale k. In this case, we denote $F_{\text{two}}(M_1, M_2)$ simply by $F_{\text{two}}(\Theta_k, k)$.

With k fixed, we determine

$$\Theta_k^* = \arg\min_{\Theta_k} F_{\text{two}}(\Theta_k, k), \tag{47}$$

which is called *parameter learning*. Then, we make *model scale selection* by determining

$$k^* = \min_{k \in \mathscr{K}} k, \quad \mathscr{K} = \{ j: J_1(j) = \min_k J_1(k)\}, \quad J_1(k) = F_{\text{two}}(\Theta_k^*, k). \tag{48}$$

That is, to pick the smallest one among those values of k that makes $J_1(k)$ reach its smallest value. In other words, we select the simplest structural scale when we have multiple choices. For some problems, $\mathscr{K}$ contains only one element only. Sometimes, $\mathscr{K}$ contains many elements. In these cases, once $J_1(k)$ reaches its smallest value at k^*, it usually keep this smallest value for all $k > k^*$.

We also have an alternative way for selecting k^*

$$k^* = \arg\min_k J_2(k),$$
$$J_2(k) = (1-\gamma) J_1(k) - \gamma \int_{x,z} \sum_y p_{M_1^L}(x,y,z)|_{\Theta_k^*} \ln p_{M_2^L}(x,y,z)|_{\Theta_k^*} \, dx \, dz, \quad 0 \le \gamma \le 1, \tag{49}$$

where $p_{M_i}(x,y,z)|_{\Theta_k^*}$, $i = 1, 2$, denote the learned joint densities given in Eq. (38) with the parameter Θ_k^* given by Eq. (47), where $\gamma = 1, J_2(k)$ is the expected complexity of the Ying machine measured by the Yang machine, is expected to be the smallest for the least complicated system. Usually, $J_2(k)$ reaches its smallest only at one value of k. In most cases, the results of Eqs. (48) and (49) are the same. However, each way has a different feature. For Eq. (48), $J_2(k)$ is more suitable for a limited number of samples, and $J_2(k)$ reduces back to $J_1(k)$ when $\gamma = 1$.

In the rest of this paper, we will only focus on the BKYY learning $\min_{M_1, M_2} KL_{\text{two}}(M_1, M_2)$.

4.2. Feedforward network vs localized architecture

As shown in Fig. 1, except that $p_{M_1^L}(z,x), p_{M_1}(x)$ and $p_M(z)$ are pre-specified by Eqs. (39) and (43), we have five components to be specified. Since each component may have several choices, we will have a family that consists of quite a number of variants, as discussed in [27]. However, in this paper we are only interested in a few of them that relate to mixture of experts and RBF nets.

First, we only consider the cases that $p_{M_{y|x,z}}(y|x,z) = p(y|x,z)$ is free and the following equality,

$$p_{M_1^L}(x,y) = p_{M_{y|x}}(y|x)p_{M_x}(x) = p_{M_{x|y}}(x|y)p_{M_y}(y) = p_{M_2^L}(x,y), \tag{50}$$

is held under some mild condition as discussed in [27]. In this case, we have that $KL(M_1, M_2) = 0$ and $\min_{M_1, M_2} KL_{\text{two}}(M_1, M_2)$ becomes equivalent to

$$p(y|x,z) = \frac{p_{M_{z|x,y}}(z|x,y)p_{M_2^L}(x,y)}{p_{M_2}(x,z)},$$

$$p_{M_2}(x,z) = \sum_y p_{M_{z|x,y}}(z|x,y)p_{M_2^L}(x,y)$$

$$= \begin{cases} \sum_y p_{M_{z|x,y}}(z|x,y)p_{M_{x|y}}(x|y)p_{M_y}(y) & \text{for a } \textit{Ying}\text{-based system,} \\ \sum_y p_{M_{z|x,y}}(z|x,y)p_{M_{y|x}}(y|x)p_{M_x}(x) & \text{for a } \textit{Yang}\text{-based system;} \end{cases}$$

For a Ying-based system:

$$\min_{M_{z|x,y}\, M_{x|y},\, M_y} KL^L(M_1, M_2) \quad \textit{or} \quad \max_{M_{z|x,y},\, M_{x|y},\, M_y} L(M_{z|x,y}, M_{x|y}, M_y),$$

$$L(M_{z|x,y}, M_{x|y}, M_y) = \int_{x,z} p_{M_1^L}(z,x) \ln \sum_y p_{M_{z|x,y}(z|x,y)p_{M_{x|y}}(x|y)p_{M_y}(y)}\, \mathrm{d}x\, \mathrm{d}z.$$

For a Yang-based system:

$$\min_{M_{z|x,y},\, M_{y|x}} KL^L(M_1, M_2) \quad \textit{or} \quad \max_{M_{z|x,y},\, M_{y|x}} L(M_{z|x,y}, M_{y|x}),$$

$$L(M_{z|x,y}, M_{y|x}) = \int_{x,z} p_{M_1^L}(z,x) \ln \sum_y p_{M_{z|x,y}}(z|x,y)p_{M_{y|x}}(y|x)p_{M_x}(x)\, \mathrm{d}x\, \mathrm{d}z. \tag{51}$$

Such a system is called *fully coordinated and fully matched BKYY learning* system [27]. More specifically, since a parametric component is actually modeled by a

physical device, and a free component is indirectly defined through the physical devices for other components, we call a BYY system a *Yang*-based system when $p_{M_{x|y}}(x|y)$ is free and $p_{M_{y|x}}(y|x)$ is parametric, a *Ying*-based system when $p_{M_{x|y}}(x|y)$ is parametric and $p_{M_{y|x}}(y|x)$ is free.

We further consider two types of features in $p_{M_{z|x,y}}(z|x,y)$. One is the special case that $p_{M_{z|x,y}}(z|x,y) = p_{M_{z|y}}(z|y)$, with $p_{M_2}(x,z)$ given in Eq. (51) becoming

$$p_{M_2}(x,z) = \sum_y p_{M_{z|y}}(z|y)p_{M_2^t}(x,y)$$
$$= \begin{cases} \sum_y p_{M_{z|y}}(z|y)p_{M_{x|y}}(x|y)p_{M_y}(y) & \text{for a } Ying\text{-based system,} \\ \sum_y p_{M_{z|y}}(z|y)p_{M_{y|x}}(y|x)p_{M_x}(x) & \text{for a } Yang\text{-based system,} \end{cases} \tag{52}$$

which links x, z via a *cascade architecture* $x \to y \to z$ or $x \leftarrow y \to z$ and thus x is independent of z when y is known. The architecture $x \to y \to z$ is usually called *three-layer feedforward net* or *three-layer perceptron*. The other is that $p_{M_{z|x,y}}(z|x,y) \neq p_{M_{z|y}}(z|y)$, and $p_{M_2}(x,z)$ is given by Eq. (51). In this case, each $p_{M_{z|x,y}}(z|x,y)$ builds a direct link $x \to z$ itself with the link gated via the internal variable y such that a weighted mixture $p_{M_2}(x,z)$ is formed by Eq. (51) in a *parallel architecture*. This architecture is usually called *localized architecture*, as the one in mixture of experts and RBF nets.

In the following, we discuss in depth the major features of the BKYY learning on *three-layer perceptron* and *localized architecture*:

4.2.1. Three-layer perceptron

One main disadvantage is that $p_{M_2}(x|z)$ obtained from $p_{M_2}(x,z)$ given by Eq. (52) cannot avoid the summation over all the value of y. For example, we have $p_{M_2}(x|z) = \sum_y p_{M_{z|y}}(z|y)p_{M_{y|x}}(y|x)$ for a *Yang*-based system. Thus, in the case that the bits k of y is large, the computing cost is very large and impractical. The similar summation will also occur during the learning given by Eq. (51). To make the computation feasible, we propose a specific design called *smoothed Yang-based learning system* as follows:

$$p_{M_{z|y}}(z|y) = p(z|E(y|x,\theta_{y|x}),\theta_{z|y}),\ p(z|y,\theta_{z|y})$$
$$= \begin{cases} \prod_{j=1}^{d_z} \pi_j(y,W_{z|y})^{z_j}(1-\pi_j(y,W_{z|y}))^{1-z_j} & \text{for a binary } z, \\ \dfrac{\sum_{j=1}^{d_z} z_j\, \pi_j(y,W_{z|y})}{\sum_{j=1}^{d_z} \pi_j(y,W_{z|y})} & \text{for a binary } z \text{ and } \sum_{j=1}^{d_z} z_j = 1, \\ G(z,f(y,W_{z|y}),\Sigma_{z|y}) & \text{for a real } z, \end{cases}$$

$$p_{M_{y|x}}(y|x) = p(y|x,\theta_{y|x})$$
$$= \begin{cases} \prod_{j=1}^{k} \mu_j(x,W_{y|x})^{y_j}(1-\mu_j(x,W_{y|x}))^{1-y_j} & \text{for a binary } y, \\ G(y,\beta(x,W_{y|x}),I) & \text{for a real } y, \end{cases} \tag{53}$$

$$\mu_j(x,W_{y|x}) = s(\beta_j(x,W_{y|x})), \quad \pi_j(y,W_{z|y}) = s(f_j(y,W_{z|y})), \quad pM_b(y) = p(y) \text{ free},$$

where

$$\beta(x,W_{y|x}) = [\beta_1(y,W_{y|x}), \ldots, \beta_k(y,W_{y|x})]$$

and

$$f(y,W_{z|y}) = [f_1(y,W_{z|y}), \ldots, f_m(z,W_{z|y})]$$

are nonlinear functions. $s(r)$ is usually a sigmoid function, e.g., $s(r) = 1/(1 + e^{-r})$ or others with its range on $[0, 1]$. Moreover, $s(r)$ can even be a monotonic increasing function (e.g., $s(r) = e^r$) for the cases of a binary z with $\sum_{j=1}^{d_z} z_j = 1$.

With this design, we have $p_{M_2}(z|x) = p(z|E(y|x,\theta_{y|x}))$ and its regression

$$E(z|x) = \begin{cases} [\pi_1(E(y|x,\theta_{y|x}),W_{z|y}), \ldots, \pi_k(E(y|x,\theta_{y|x}),W_{z|y})]^{\mathrm{T}} & \text{for a binary } z, \\ f(E(y|x,\theta_{y|x}),W_{z|y}) & \text{for a real } z, \end{cases} \tag{54}$$

where $E(y|x,\theta_{y|x}) = [\mu_1(x\, W_{y|x}), \ldots, \mu_k(x\, W_{y|x})]^{\mathrm{T}}$. In the special case that $\beta(x, W_{y|x}) = W_{y|x}x$ and $f(y\, W_{z|y}) = W_{z|y}y$. This $E(z|x)$ is just the output of the conventional three-layer perceptron and there is no implementing difficulty anymore.

For the learning under this design, from Eq. (51) we have that $\max_{M_{z|x,\,y},\, M_{y|x}} L(M_{z|x,y}, M_{y|x})$ becomes

$$\max_{\theta_{z|y},\theta_{y|x}} L(\theta_{z|y},\theta_{y|x}), \quad L(\theta_{z|y},\theta_{y|x}) = \sum_{(x,z)\in D_{x,z}} \ln p(z|E(y|x,\theta_{y|x}),\theta_{z|y}), \tag{55}$$

which is exactly the conventional maximum likelihood learning for three-layer perceptron that includes the least-squares learning by the well-known back-propagation technique.

Moreover, we can also get one interesting new result. From Eq. (49), we can get a new criterion for selection k (i.e., the number of the hidden units) as follows:

$$k^* = \arg\min_k J_2(k), \quad J_2(k) = -L(\theta^*_{z|y},\theta^*_{y|x}) - \sum_{j=1}^{k} [\mu_j(x,W^*_{y|x}) \ln \mu_j(x,W^*_{y|x}) + (1 - \mu_j(x,W^*_{y|x})) \ln(1 - \mu_j(x,W^*_{y|x}))], \tag{56}$$

where $\theta^*_{z|y},\theta^*_{y|x},W^*_{y|x}$ are the results of the learning by Eq. (55) at the current fixed k.

4.2.2. Localized architecture

As discussed above, the architecture of three-layer perceptron cannot avoid the dilemma due to the summation over all the possible values y, which causes an impractical computing cost unless y takes only a few values. However, if y just takes a few values, the representation ability of the network is very limited since y functions as a bottle-neck that the entire information flow must go through. As a result, some kind of approximation as discussed above must to be used.

However, this dilemma can be avoided by the localized architectures. We consider a particular case that $y = [y^{(1)}, \ldots, y^{(k)}]^{\mathrm{T}}$ with $y^{(j)} \in \{0,1\}$, $\sum_{j=1}^{k} y^{(j)} = 1$. That is, y only

takes k values and $y^{(j)}=1$ is equivalent to y taking the jth value. We make the following design:

$$
\begin{aligned}
&p_{M_{z|x,y}}(z|x,y)=\sum_{j=1}^{k} y^{(j)}p(z|x,\theta_j), \quad p(z|x,\theta_j) \text{ is parametric,}\\
&p_{M_{y|x}}(y|x)=\sum_{j=1}^{k} y^{(j)}p(j|x,v), \quad \sum_{j=1}^{k} p(j|x,v)=1, \quad p(j|x,v) \text{ is parametric,}\\
&p_{M_{x|y}}(x|y)=\sum_{j=1}^{k} y^{(j)}p(x|v_j), \quad p_{M_y}(y)=\sum_{j=1}^{k} y^{(j)}\alpha_j, \quad \sum_{j=1}^{k}\alpha_j=1,\ \alpha_j\geq 0,
\end{aligned}
\tag{57}
$$

and also that $p_{M_x}(x)=p_{h_x}(x)$ given by Eq. (39) and $p_{M_{z|x}}(z|x)=p_{h_z}(z|x)$ by Eq. (43) with $h_x\to 0$, $h_z\to 0$ and $N\to\infty$.

With this design, from Eq. (51) we have that

$$
p_{M_2}(z|x)=p(z|x,\Theta)
=\begin{cases}
\sum_{j=1}^{k} p(z|x,\theta_j)\dfrac{\alpha_j\ p(x|v_j)}{p(x|v)}, & \\
\quad p(x|v)=\sum_{j=1}^{k}\alpha_j p(x|v_j) & \text{for a } \textit{Ying}\text{-based system,}\\
\sum_{j=1}^{k} p(z|x,\theta_j)p(j|x,v), & \text{for a } \textit{Yang}\text{-based system.}
\end{cases}
$$

For a Ying-based system,

$$
\begin{aligned}
&\min_{\{\theta_j,v_j,\alpha_j\}_{j=1}^{k}} KL^{L}(\{\theta_j,v_j,\alpha_j\}_{j=1}^{k}) \quad \textit{or} \quad \max_{\theta_j,v_j,\alpha_j} L(\{\theta_j,v_j,\alpha_j\}_{j=1}^{k}),\\
&L(\{\theta_j,v_j,\alpha_j\}_{j=1}^{k})=\sum_{i=1}^{N}\ln\sum_{j=1}^{k} p(z|x,\theta_j)\alpha_j p(x|v_j),
\end{aligned}
\tag{58}
$$

For a Yang-based system,

$$
\begin{aligned}
&\min_{\theta_j,v} KL^{L}(\{\theta_j\}_{j=1}^{k},v) \quad \textit{or} \quad \max\nolimits_{\{\theta_j\}_{j=1}^{k},v} L(\{\theta_j\}_{j=1}^{k},v),\\
&L(\{\theta_j\}_{j=1}^{k},v)=\sum_{i=1}^{N}\ln\sum_{j=1}^{k} p(z|x,\theta_j)p(j|x,v).
\end{aligned}
$$

That is, the BKYY learning on the specific Yang-based system and Ying-based system is exactly the maximum likelihood learning on the original ME model equation (Eq. (1)) and the alternative ME model equations (Eqs. (4) and (8)), respectively! Actually, the learning by $\min_{\theta_j,v} KL^{L}(\{\theta_j\}_{j=1}^{k},v)$ and $\min_{\{\theta_j,v_j,\alpha_j\}_{j=1}^{k}} KL^{L}(\{\theta_j,v_j,\alpha_j\}_{j=1}^{k})$ can be implemented by the *alternative minimization* iterative procedure Eq. (45), which turns out to be exactly the *EM Algorithm – Original* and the *EM Algorithm – Alternative* given in Section 2, respectively.

Although we still encounter the summation over all the values of y, here y only takes k different values. Because each expert $p(z|x,\theta_j)$ has a direct link $x \to z$ itself and y only functions as a gate that weights information flows from different experts, we can trade-off the representation complexity of y and the structural complexity of each expert such that number of values that y should take are significantly fewer than it should take on three layer perceptron. This is a favorable advantage of the localized architectures of ME models and RBF nets that the forward network lacks. Moreover, as shown in Sections 2 and 3, ME models and RBF nets can be trained with the EM algorithm or its variants with good convergence properties [12,31]. In addition, as well known in the literature, the good generalization can be obtained by a localized architecture since the effect of dividing a complex tasks into a number of more regular sub-tasks is equivalent to making some regularization on the network performance.

4.3. Selection criteria for the number of experts and basis functions

In the existing literature, there still lacks theoretical guide on how to determine the best number k^* of experts or basis functions for the original and alternative ME models as well as RBF nets. According to their links to the BKYY learning as discussed in the previous subsection, with the design given by Eq. (57) we can obtain such selection criteria directly from Eqs. (48) and (49) with $F_{\text{two}}(\Theta_k,k)$ replaced by $KL^L(\{\theta_j,v_j,\alpha_j\}_{j=1}^k)$ and $KL^L(\{\theta_j\}_{j=1}^k,v)$ given by Eq. (58). After ignoring some irrelevant terms, we can get the detailed forms of the criteria for different specific cases. In the following, we list a number of major ones, in which those criteria for $J_1(k)$ were obtained first in 1996 [24,25] and the criteria for $J_2(k)$ at the special case $\gamma = 1$ were obtained first in [25,28], while the following general forms of $J_1(k)$ are newly proposed in this paper.

1. *The selection criteria for the original ME model.* That is, for $p(z|x,\theta_j)$ in general case we have

$$k^* = \min_{k\in\mathcal{K}} k, \quad \mathcal{K} = \left\{j: J_1(j) = \min_k J_1(k)\right\},$$

$$J_1(k) = KL^L(\{\theta_j^*\}_{j=1}^k,v^*) = \frac{1}{N}\sum_{i=1}^N\sum_{j=1}^k h^*(j|x_i)\ln\frac{h^*(j|x_i)}{p(j|x,v^*)} - \frac{1}{N}\sum_{i=1}^N\sum_{j=1}^k h^*(j|x_i)\ln p(z_i|x_i,\theta_j^*),$$

$$h^*(j|x_i) = p^*(j|x_i,z_i) = \frac{p(j|x_i,v^*)p(z_i|x_i,\theta_j^*)}{\sum_{j=1}^k p(j|x_i,v^*)p(z_i|x_i,\theta_j^*)},$$

$$k^* = \arg\min_k J_2(k), \tag{59}$$

$$J_2(k) = -\frac{1}{N}\sum_{i=1}^N\sum_{j=1}^k h^*(j|x_i)\ln p(j|x,v^*) - \frac{1}{N}\sum_{i=1}^N\sum_{j=1}^k h^*(j|x_i)\ln p(z_i|x_i,\theta_j^*), \quad \frac{1-\gamma}{N}\sum_{i=1}^N\sum_{j=1}^k h^*(j|x_i)\ln h^*(j|x_i), \quad 0 \le \gamma \le 1.$$

When $p(z|x,\theta_j)$ is Gaussian given by Eq. (1), by noticing $\mathrm{Tr}[\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\times[z-f_j(x,w_j)]^{\mathrm{T}}\Gamma_j^{-1}[z-f_j(x,w_j)]\approx \mathrm{Tr}[I]=d$, the above criteria actually takes the following specific form:

$$J_1(k)=\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln\frac{h^*(j|x_i)}{p(j|x,v^*)}+0.5\sum_{j=1}^{k}\alpha_j^*\ln|\Gamma_j^*|,\quad \alpha_j^*=\frac{1}{N}\sum_{i=1}^{N}h^*(j|x_i),$$

$$\begin{aligned}J_2(k)=&-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln p(j|x,v^*)+0.5\sum_{j=1}^{k}\alpha_j^*\ln|\Gamma_j^*|\\&+\frac{1-\gamma}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln h^*(j|x_i),\quad 0\le\gamma\le 1.\end{aligned}\tag{60}$$

2. *The selection criteria for the alternative ME model.* That is, for $p(z|x,\theta_j)$ in general case we have

$$k^*=\min_{k\in\mathscr{K}}k,\quad \mathscr{K}=\{j{:}J_1(j)=\min_k J_1(k)\},\quad J_1(k)=KL^L(\{\theta_j^*,v_j^*,\alpha_j^*\}_{j=1}^k),$$

$$\begin{aligned}J_1(k)=&\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln h^*(j|x_i)\\&-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln p(z_i|x_i,\theta_j^*)p(x|v_j^*)-\sum_{j=1}^{k}\alpha_j^*\ln\alpha_j^*,\end{aligned}$$

$$h^*(j|x_i)=p^*(j|x_i,z_i)=\frac{\alpha_j^*p(x_i|v_j^*)p(z_i|x_i,\theta_j^*)}{\sum_{j=1}^{k}\alpha_j^*p(x_i|v_j^*)p(z_i|x_i,\theta_j^*)},$$

$$k^*=\arg\min_k J_2(k),$$

$$\begin{aligned}J_2(k)=&-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln p(z_i|x_i,\theta_j^*)p(x|v_j^*)-\sum_{j=1}^{k}\alpha_j^*\ln\alpha_j^*\\&+\frac{1-\gamma}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln h^*(j|x_i),\quad 0\le\gamma\le 1.\end{aligned}\tag{61}$$

When $p(z|x,\theta_j)$ is Gaussian given by Eq. (1) and $p(x|v_j)$ is Gaussian given by Eq. (5), the above criteria actually takes the following specific form:

$$J_1(k)=\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln h^*(j|x_i)+0.5\sum_{j=1}^{k}\alpha_j^*(\ln|\Gamma_j^*|+\ln|\Sigma_j^*|)-\sum_{j=1}^{k}\alpha_j^*\ln\alpha_j^*,$$

$$\begin{aligned}J_2(k)=&0.5\sum_{j=1}^{k}\alpha_j^*(\ln|\Gamma_j^*|+\ln|\Sigma_j^*|)-\sum_{j=1}^{k}\alpha_j^*\ln\alpha_j^*\\&+\frac{1-\gamma}{N}\sum_{i=1}^{N}\sum_{j=1}^{k}h^*(j|x_i)\ln h^*(j|x_i),\quad 0\le\gamma\le 1.\end{aligned}\tag{62}$$

3. *The selection criteria for the RBF nets.* From $\alpha_j^* = \sqrt{|\Sigma_j^*|}/\sum_{j=1}^k \sqrt{|\Sigma_j^*|}$ given by Eq. (16), we can obtain directly from the above Eq. (62) that

$$J_1(k) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k} h^*(j|x_i)\ln h^*(j|x_i) + \sum_{j=1}^{k}\frac{\sqrt{|\Sigma_j^*|}}{\sum_{j=1}^k\sqrt{|\Sigma_j^*|}}\ln\sqrt{|\Gamma_j^*|} + \ln\sum_{j=1}^{k}\sqrt{|\Sigma_j^*|},$$

$$h(j|x_i) = \frac{e^{-0.5(x_i-m_j^*)^T(\Sigma_j^*)^{-1}(x_i-m^*{}_j)}G(z_i, r_j^*, \Gamma_j^*)}{\sum_{j=1}^k e^{-0.5(x_i-m_j^*)^T(\Sigma_j^*)^{-1}(x_i-m^*{}_j)}G(z_i, r_j^*, \Gamma_j^*)},$$

$$r_j^* = \begin{cases} c_j^*, & \text{for an ENRBF net,} \\ W_j^{*T}x_i + c_j^*, & \text{for an ENRBF net;} \end{cases}$$

$$J_2(k) = \sum_{j=1}^{k}\frac{\sqrt{|\Sigma_j^*|}}{\sum_{j=1}^k\sqrt{|\Sigma_j^*|}}\ln\sqrt{|\Gamma_j^*|} + \ln\sum_{j=1}^{k}\sqrt{|\Sigma_j^*|} + \frac{1-\gamma}{N}\sum_{i=1}^{N}\sum_{j=1}^{k} h^*(j|x_i)\ln h^*(j|x_i), \quad 0 \le \gamma \le 1. \tag{63}$$

Taking $J_2(k)$ given in Eq. (62) with $\gamma = 1$ as an example, we can intuitively understand how the above criteria work. Its first term will decrease as k increases and its second term will increase as k increases and thus trades off the best k^*. This point can be even more clearly observed by letting $\Gamma_j^* = \Gamma^*$, $\Sigma_j^* = \Sigma^*$ and $\alpha_j^* = 1/k$ such that

$$J_2(k) = 0.5(\ln|\Gamma^*| + \ln|\Sigma^*|) + \ln k, \quad \text{at } \gamma = 1. \tag{64}$$

Obviously, $\ln k$ increases as k increases, and $|\Gamma^*|, |\Sigma^*|$ decreases as k increases for a given number N of samples.

5. Experimental results

In all the experiments, the initialization of the parameters for the EM algorithms and its variants are made randomly in their corresponding domains, e.g., Σ_j is initialized at a positive-definite matrix. A large number of experiments have been made at different random initializations, and have turned out that the initialization will not obviously affect the performance, which is consistent to the advantage of the EM algorithm on initialization discussed in [31]. In contrast, for the conventional two stage algorithm, the performance of the K-means algorithm highly depends on its initialization [5,7]. In this section, only a few examples are demonstrated, and the best results we got by the conventional two stage algorithm are provided in our following comparisons.

5.1. Experiments on foreign exchange rate and stock price prediction

Experiments are made on the prediction of a foreign exchange rate and a stock price for comparing the following algorithms:

1. The conventional two-stage training algorithms [16,10] for NRBF and ENRBF nets, denoted by NRBF two-stage and ENRBF two-stage, respectively;
2. The *EM Algorithm–RBF*, denoted by EM-NRBF and EM-ENRBF for NRBF net and extended NRBF net respectively;
3. The *EM Algorithm–RBF* modified by coordinated competitive learning (CCL), denoted by EM-NRBF (CCL) and EM-ENRBF (CCL), respectively;
4. The *Adaptive EM Algorithm–RBF*, denoted by Adaptive EM-NRBF and Adaptive EM-ENRBF respectively;

A foreign exchange rate data for USD-DEM with 1112 samples (25 November 1991–30 August 1995) and a real stock price data of 380 samples from Shanghai stock market are used. On the USD-DEM Forex data, two type of partitions of the training set and the testing set are used. For the Type A, the training size is the first 1000 samples and the testing size is the subsequent 112 samples. For Type B, the training size is first 100 samples and the testing size is the subsequent 1012 samples. For the real stock price data, the first 350 samples used as the training set and the subsequent 30 samples as the testing set.

After certain experiments, our experience tells that it is the best to set the input dimension at 3, that is, we use $x = [x(t-1), x(t-2), x(t-3)]$ as our input vector at the time t.

5.1.1. Conventional algorithm, EM algorithm and coordinated competitive learning

Several experiments are made on USD-DEM-SET Type A data. Shown in Table 1 are the results of using five basis functions on both NRBF net and ENRBF net for the task of using $x = [x(t-1), x(t-2), x(t-3)]$ to predict $z = x(t)$. The figures are normalized mean square error (NMSE). The detailed prediction results on the testing set are shown in Fig. 2. For NRBF net, EM-NRBF indeed improves the conventional two stage algorithm considerably. For ENRBF net, EM-NRBF also improves, especially shown from Fig. 2. However, the conventional algorithm performs already quite good.

We further examine whether the bad result by the two stage algorithm is due to not enough number of basis functions used. Shown in Table 2 are the results of the two-stage algorithm on NRBF net with different number of hidden units, which indicates that increasing the units indeed can improve the results. So, we need to consider a large number of basis functions. Unfortunately, as the number of basis functions increases, the convergence speed reduces down considerably for EM-NRBF, which makes the experiments become too time-consuming to be implemented. Thus, we turn to compare EM-NRBF (CCL) with the two-stage algorithm. Shown in Table 3 are the results of using 20 basis functions, with the detailed prediction results on the testing set shown in Fig. 3. We again observe that EM-NRBF indeed improves

Table 1
The results of prediction on FOREX rate of USD-DEM-SET Type A (No. of units = 5)

Algorithms	NRBF two-stage	EM-NRBF	ENRBF two-stage	EM-ENRBF
Training (NMSE)	0.553	0.894	0.143	0.152
Testing (NMSE)	2.92	0.774	0.452	0.448

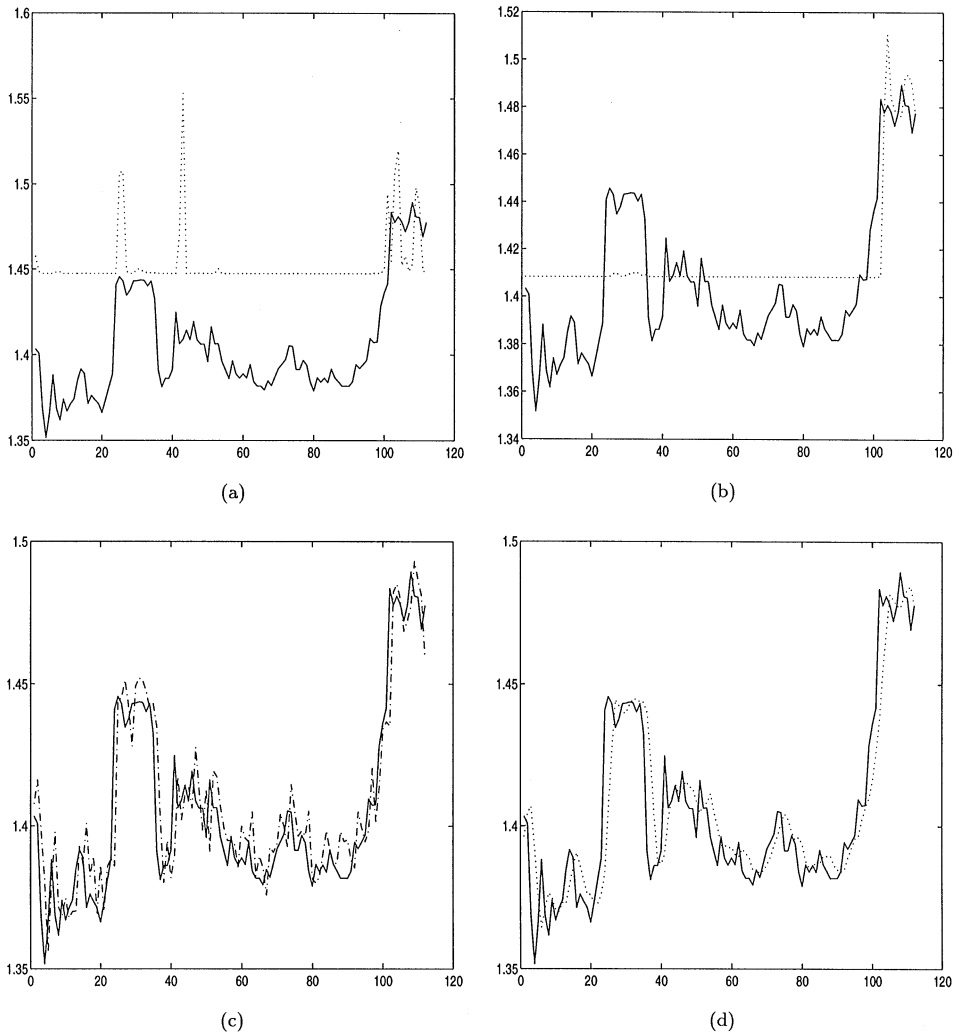


Fig. 2. The results of prediction on Forex data of USD-DEM-Set Type A (No. of units = 5), corresponding to Table 1, where the solid line is for data and the dashed line is for prediction result, and this convention is kept the same for all the figures in this paper: (a) by NRBF two-stage, (b) by EM-NRBF, (c) by ENRBF two-stage, (d) by EM-ENRBF.

Table 2
The results of prediction on FOREX rate of USD-DEM-SET Type A (by NRBF two-stage only)

No. of units	5	10	15	20
Training (NMSE)	0.553	0.647	0.514	0.396
Testing (NMSE)	2.92	4.29	3.85	1.70

Table 3
The results of prediction on FOREX rate of USD-DEM-SET Type A (No. of units = 20)

Algorithms	Training flops [a]	Training (NMSE)	*Testing* (*NMSE*)
NRBF two-stage	4.81×10^5	0.396	1.703
EM-NRBF (CCL) II	5.94×10^5	0.238	0.768
ENRBF two-stage	3.91×10^5	0.173	0.452
EM-ENRBF (CCL) II	3.96×10^6	0.151	0.445

[a]Here one flop is counted by MATLAB as an addition or multiplication operation.

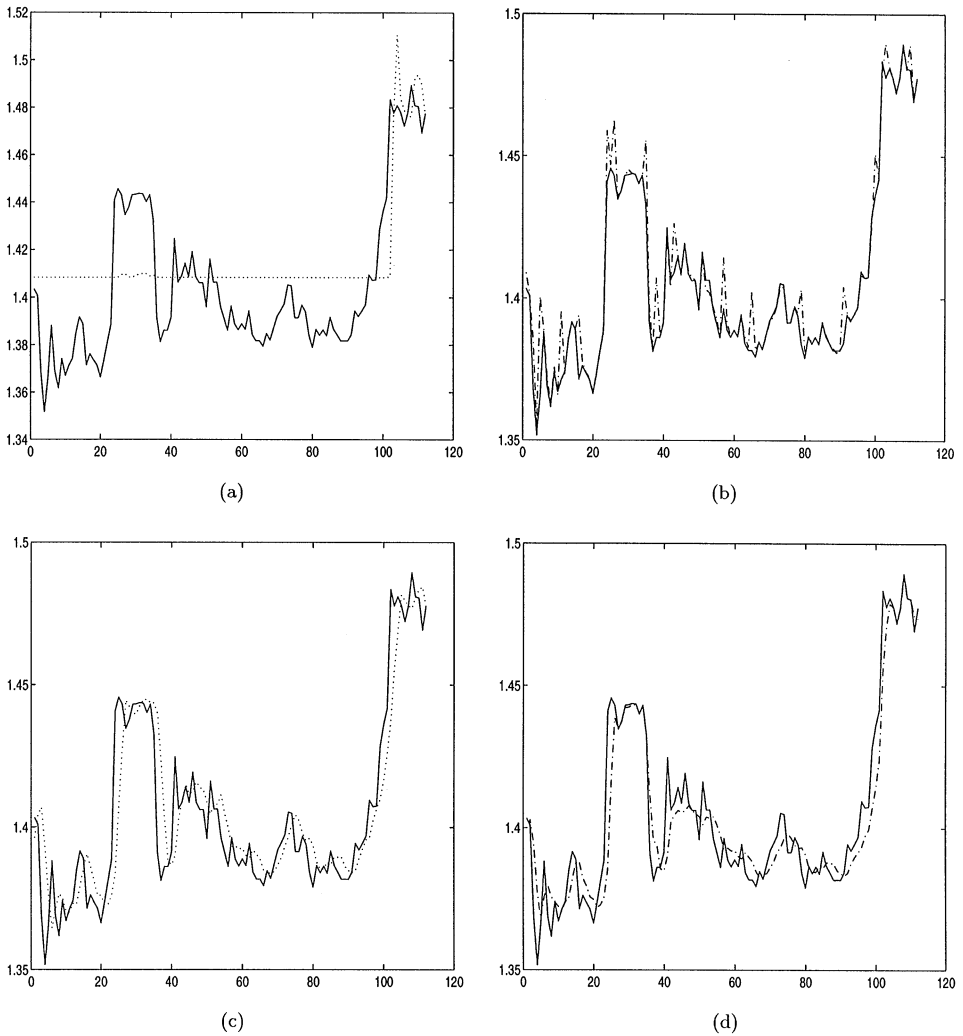


Fig. 3. The results of prediction on Forex data of USD-DEM-Set Type A (No. of units = 20), corresponding to Table 3: (a) by NRBF two-stage, (b) by EM-NRBF, (c) by ENRBF two-stage, (d) by EM-ENRBF Algorithm II.

the two stage algorithm considerably. Here, although the two stage algorithm on NRBF net got a better result than its result with 5 units, it is still far from that obtained by EM-NRBF (CCL), especially as shown in Fig. 3. We can also see that the computing cost of EM-NRBF (CCL) is almost comparable to the conventional algorithm, which is a significant speeding up from EM-NRBF that is about one or two magnitudes slower. Moreover, we found that further increasing the number of basis functions will not obviously improve the results by the two stage algorithm, but its computing cost will increase fast and become worse than EM-NRBF(CCL).

5.1.2. Adaptive algorithm for tracking temporal change

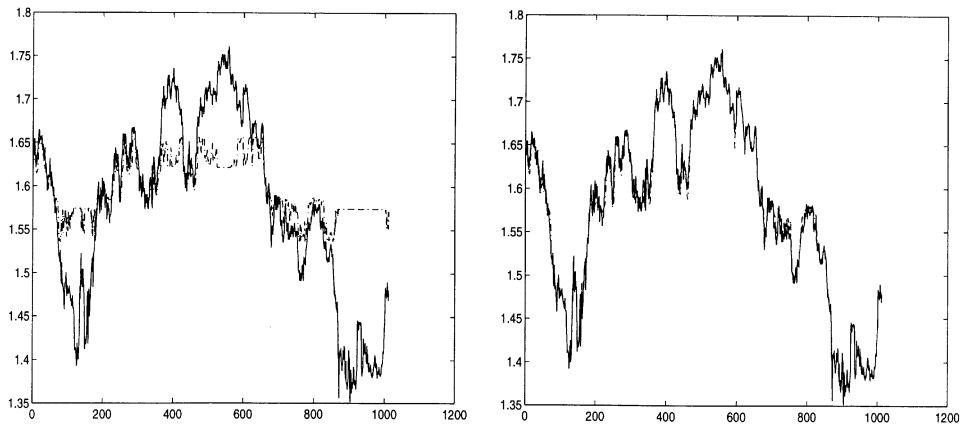
In Fig. 4, the comparison are made on Forex data of USD-DEM-SET Type B by EM-NRBF (CCL) and Adaptive EM-NRBF. The adaptive algorithm is used to track time series in such a way that the sample point at t is used to modify the network once this point is known already (i.e., once the current time t is passed into $t+1$). As shown in Fig. 4, the adaptive algorithm indeed can track the temporal change very well and outperform its corresponding batch way algorithm significantly.

5.1.3. Results on stock data prediction

Shown in Fig. 5, are the comparison results on the real stock data by EM-NRBF (CCL) and Adaptive EM-NRBF, with the other experimental setting kept the same as in Fig. 2. Again, the adaptive algorithm can outperform its corresponding batch way algorithm significantly.

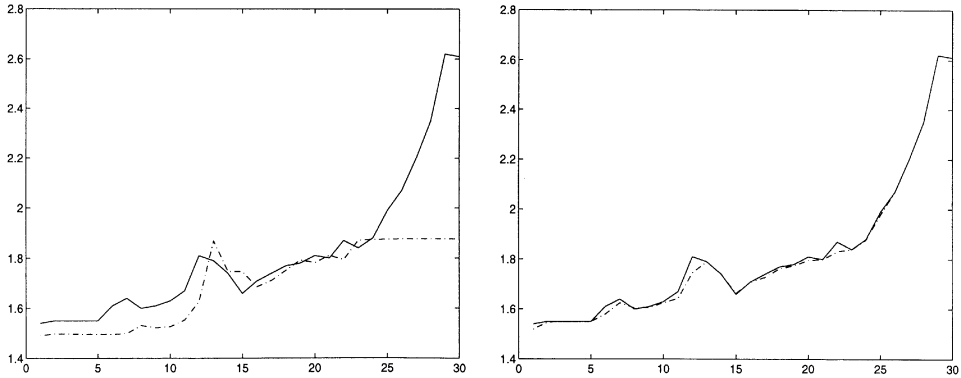
5.2. Experiments on trading investment

We compare the batch way and adaptive algorithms for both NRBF net and ENRBF net on the trading investment based on prediction. Forex data of



(a) Normalized Testing Error = 0.0840 (b) Normalized Testing Error = 0.0083

Fig. 4. The results of prediction on Forex data of USD-DEM-SET Type B (No. of units = 20): (a) by EM-NRBF (CCL), (b) by Adaptive EM-NRBF (the prediction and the real data are almost overlapped).



(a) Normalized Testing Error = 0.2320 (b) Normalized Testing Error = 0.0169

Fig. 5. The results of prediction on the stock price data (No. of units = 20): (a) by EM-NRBF (CCL), (b) by Adaptive EM-NRBF.

Table 4
The results of trading investment based on the prediction on USD-DEM-SET Type A

Algorithms	Net profit point	Profit in US$ (in 112 days)
EM-NRBF (CCL)	1425	9262.5
Adaptive EM-NRBF (CCL)	3966	25779.0
EM-ENRBF (CCL)	2063	13406.5
Adaptive EM-ENRBF (CCL)	2916	18954.0

USD-DEM-SET Type A is used again such that we can compare the result made in our previously results by the different approaches [4,30]. We use the simple trading rule proposed in [4] for trading investment. We assume that a trader can hold at most a long or short contract of foreign dollars or stock at one time. The deposit amount is fixed to be US dollar 6500 and the transaction cost rate is 0.5% of the deposit.

The results are shown in Table 4, from which we can see that adaptive algorithms can bring significant profit. Especially, Adaptive EM-NRBF NRBF net improves its non-adaptive version with the profits being as large as nearly 3 times. Also, Adaptive EM-NRBF on NRBF net gives the best result which is a considerable improvement over the one made on ENRBF net. Moreover, the batch way algorithm on ENRBF net got an obvious better result over that on NRBF net. However, the adaptive algorithm on ENRBF net performs worse than NRBF net. This is because that the adaptive process on ENRBF net is more difficult since more parameters have to be updated appropriately. In addition, the result given in Table 4 also provides a considerable improvement over the result made in [4], where only 99 days are used as testing (actually, its result on 112 days was much worse than on 99 days). Also, the result in [4] was compared with two conventional methods (Random walk and AMAR model) with significant improvements already. The profit obtained here by

Adaptive EM-NRBF is even much better, and actually has almost doubled the profit obtained in [4].

Finally, we compare the batch way and adaptive algorithms for both NRBF net and ENRBF net on the trading investment by using the recent proposed Supervised Decision Network [30] which is a direct trading system without prediction. Still, the transaction cost rate is 0.5% of the deposit. The result is given in Table 5. Again, Adaptive EM-NRBF improves its non-adaptive version with the profits being as large as nearly 3 times. Also, the result here is better than that given in Table 4, which is consistent with the conclusion in [35] that the direct trading system without prediction can provide more profit than predicting first and than making trading investment based on prediction. In addition, the result given in Table 5 also considerably improves the one given in Ref. [30] on the same data set.

5.3. Experiments on selection of radial function number

Here, we cannot directly use the previous practical data because it is hard to know its *true* k^*. Thus, we consider some piecewise nonlinear regression problems, with the data generated with the known *true* k^*.

In Fig. 6, we generate five linear segments with each having 500 samples, disturbed by Gaussian white noises of two levels of variances, namely $\sigma^2 = 0.01$ (low noise) and $\sigma^2 = 0.05$ (high noise) as shown in Fig. 6a and b. Fig. 6c and d correspond to Fig. 6a and b, respectively. In all the cases, from the curves of $J_1(k)$ and $J_2(k)$ given by Eq. (63) with $\gamma = 1$, the correct number $k^* = 5$ is detected at the minimums of the curves.

In Fig. 7, 15 linear segments are generated along the curves of $\sin(x)$ function, as shown in Fig. 7a and b. Again, each segment has 500 samples, disturbed by Gaussian white noises of two levels that are the same as in Fig. 6. The curves of $J_1(k)$ and $J_2(k)$ given by Eq. (63) with $\gamma = 1$ are given in Fig. 7c and d, corresponding to Fig. 7a and b, respectively. Still, the correct number $k^* = 15$ is detected at the minimums of the curves.

5.4. Experiments on piecewise curve fitting and detection

Another direct application of an ERBF net with the learning algorithms and the number selection criterion proposed in this paper is to make piecewise curve fitting and detection, which have many applications in the literature of image processing and computer vision. A widely used technique is the so-called *Hough transform* (HT) and

Table 5
The results of trading investment by Supervised Decision Network on USD-DEM-SET Type A

Algorithms	Net profit point	Profit in US$ (in 112 days)
EM-NRBF (CCL)	1605	10432.5
Adaptive EM-NRBF (CCL)	4237	27540.5
EM-ENRBF (CCL)	2660	17290.0
Adaptive EM-ENRBF (CCL)	3207	20845.5

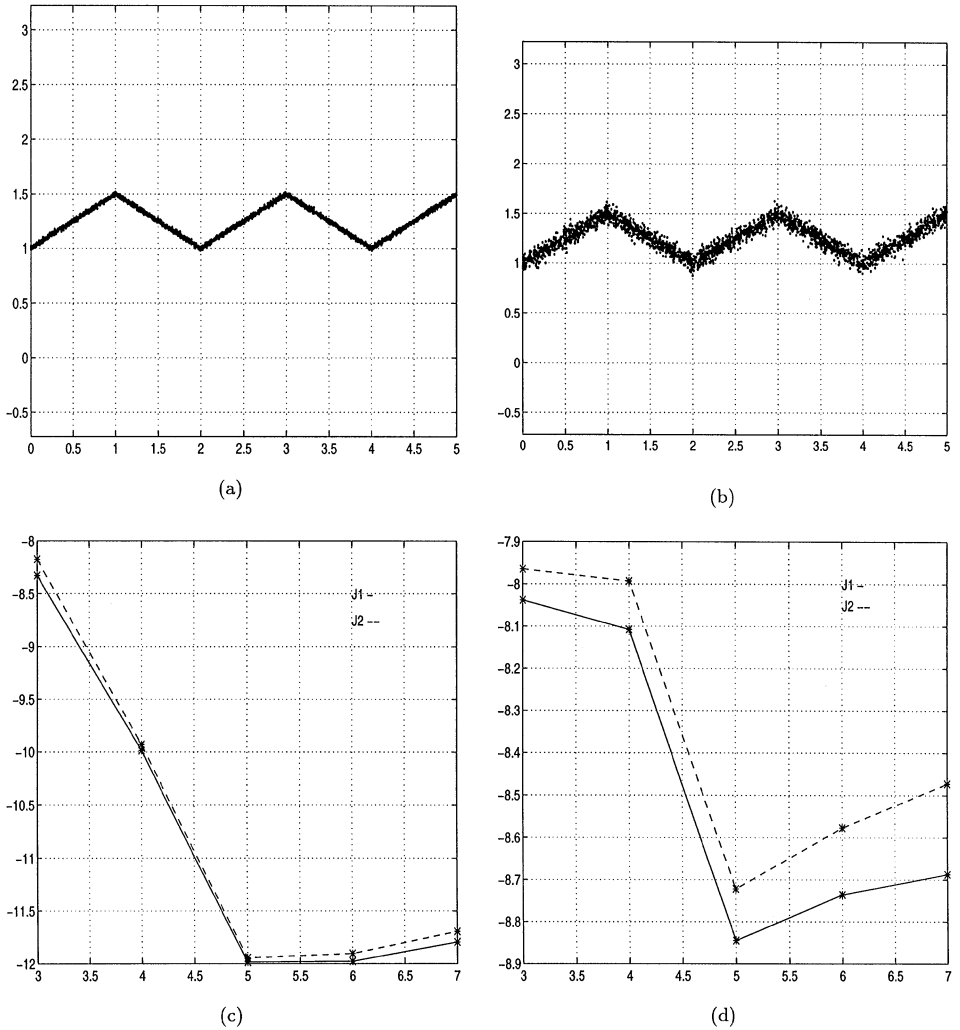


Fig. 6. The results of piecewise linear regression, with five linear segments: (a) the data disturbed by low noise, (b) the data disturbed by high noise, (c), (d) the resulting $J_1(k)$ and $J_2(k)$ with $\gamma = 1$ curves for the data in (a), (b), respectively.

its various variants. In the end of 1980s, a new Hough-like technique called *randomized Hough transform* (*RHT*) has been proposed with several advantages over the conventional HT technique [39,38,13]. However, both HT and RHT techniques do not apply well to the cases of those "fat" lines due to strong noise disturbance. Also, there is no theoretical guide to detect the number of lines in an image. These two problems can be solved here.

For a binary image with each point denoted by its coordinate (x,z), we can get a set $D_{x,z} = \{(x_i, z_i)\}_{i=1}^{N}$ as a collection of all the points on an image. Using this set to train

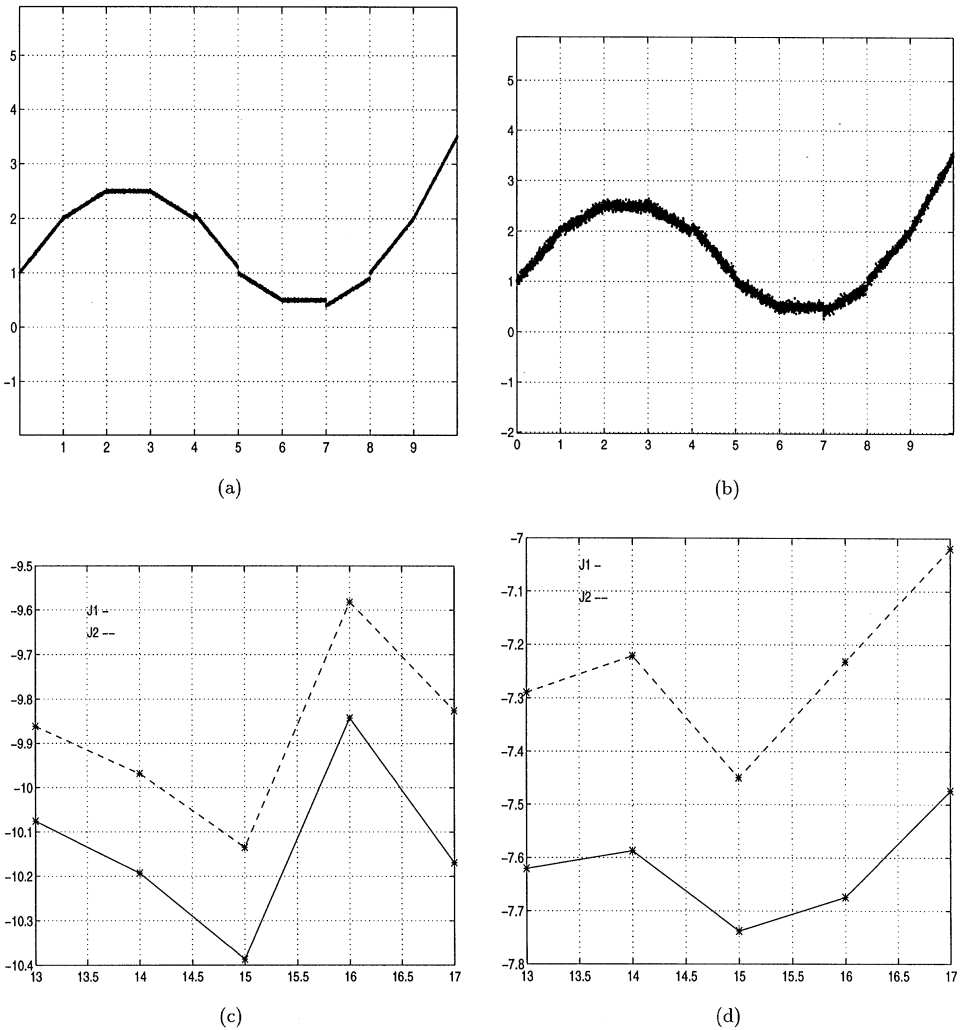


Fig. 7. The results of piecewise linear regression, with 15 linear segments: (a) the data disturbed by low noise, (b) the data disturbed by high noise, (c), (d) the resulting $J_1(k)$ and $J_2(k)$ with $\gamma = 1$ curves for the data in (a), (b), respectively.

an ENRBF net with the previously proposed learning algorithms, each of the resulted basis function with its corresponding $z = W_j^{\mathrm{T}} x + c_j$ will fit a line and the number of line is detected as the number k of basis functions, which can be made by $J_1(k)$ and $J_2(k)$ given by Eq. (63) with $\gamma = 1$.

Shown in Fig. 8 is an illustration on ten lines generated artificially. Each line is disturbed by Gaussian white noises of two levels as shown in Fig. 8a and b. An ENRBF net with Adaptive EM-ENRBF proposed in Section 3 is used. The curves of $J_1(k)$ and

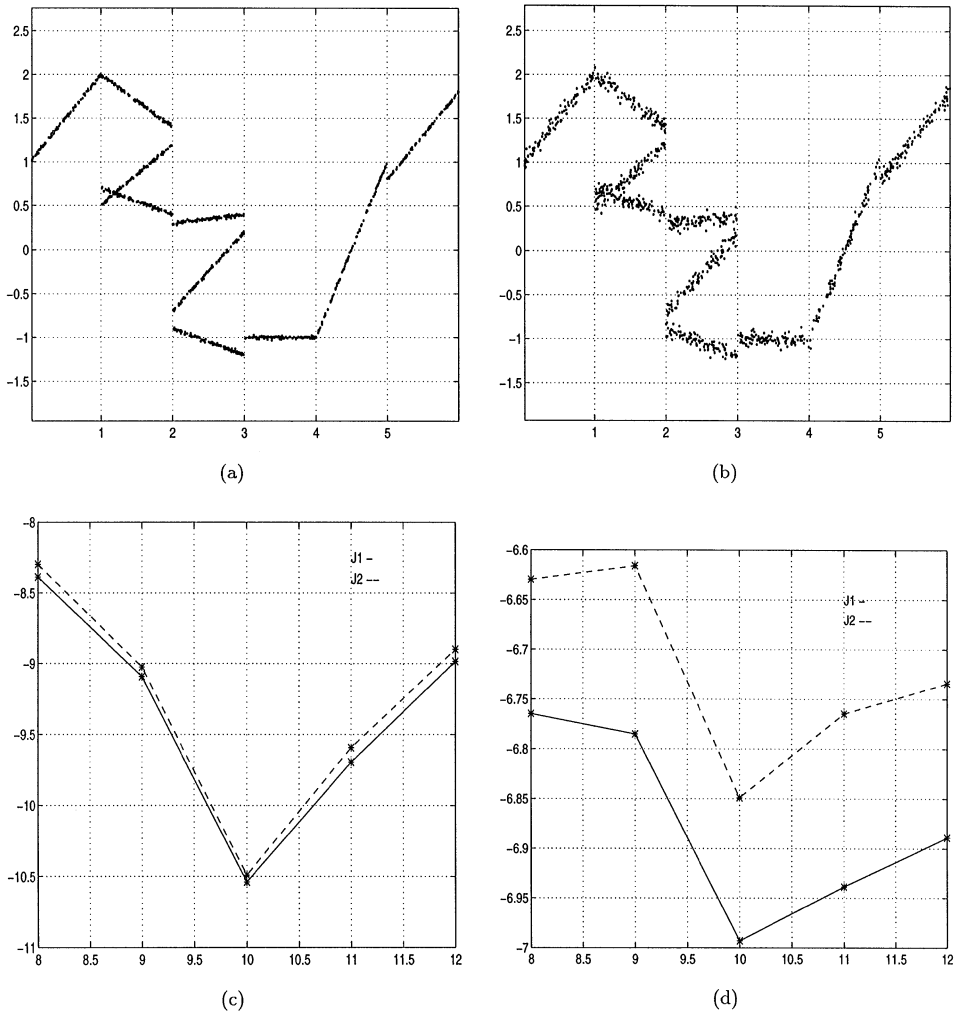


Fig. 8. The results of piecewise linear curve fitting, with ten linear segments: (a) the data disturbed by low noise, (b) the data disturbed by high noise, (c), (d) the resulting $J_1(k)$ and $J_2(k)$ with $\gamma = 1$ curves for the data in (a),(b), respectively.

$J_2(k)$ given by Eq. (63) with $\gamma = 1$ are shown in Fig. 8c and d, corresponding to Fig. 8a and b, respectively. Interestingly, the correct number of lines $k^* = 10$ is detected at the minimum of each curve again. Also, we found that each of the ten basis function fits each line with its corresponding $z = W_j^{*\mathrm{T}} x + c_j^*$ well, which are omitted from Fig. 8 for the clarity of the figure.

More generally, for each basis function in an ENRBF net, by replacing $W_j^{\mathrm{T}} x + c_j$ with a general curve $z = f_j(x, W_j)$, we can also similarly make general multiple curve fitting and detection. Even generally, we can use the alternative ME model proposed

in Section 2.1 for curve fitting and detection in such cases, with each expert $f_j(x,W_j)$ for fitting a curve and Eq. (61) for detecting the number of curves.

6. Concluding remarks

NRBF nets and ENRBF nets are shown to be special cases of the Alternative ME model and thus can be trained by the EM algorithm for determining the parameters of the input layer and the parameters of the output layer globally. Moreover, CCL and adaptive techniques can be used to approximate the EM algorithm for considerably speeding up the learning of the original and alternative ME models as well as NRBF nets and ENRBF nets. Furthermore, three-layer perceptron, the original and alternative ME models as well as NRBF nets and ENRBF nets are all shown to be special cases of the BYY learning system and theory. With this theory, we not only arrived at a new criterion for selecting the number of hidden units in three-layer perceptron, but also found that the localized architecture of the ME models and RBF nets is more preferred than the multilayer architecture. In addition, the BYY learning theory also provides us model selection criteria for determining the number of experts and basis functions.

A number of experiments are made on prediction of foreign exchange rate, trading investment, piecewise nonlinear regressions, piecewise curve fitting and detection. From the experimental results, we get the following empirical conclusions:

1. For NRBF net, the EM algorithm improves the conventional two-stage algorithm considerably. For ENRBF net, the EM algorithm can still improve the conventional two-stage algorithm although not so considerably.
2. The EM algorithm with the CCL hardcut technique can significantly speed up convergence while still keep a very good performance.
3. By using the adaptive algorithm to track time series, we can get significant improvements on financial predication and trading investment.
4. For the conventional two-stage algorithm, ENRBF net is much better than NRBF net. However, it is not the case for the EM algorithm as well as its CCL hardcut and adaptive variants.
5. The obtained criteria can detect the number of radial basis functions correctly. ENRBF net with the proposed algorithms and the number selection criteria can perform line fitting and detection well.

Acknowledgements

The author thanks Yiu-ming Cheung, Chu Ching and Tsang Sin Ma for the help in the experiments of prediction and trading investment with foreign exchange rate and stock price, Wing-kai Lam for the help in the experiments of piecewise nonlinear regression with the detection on the number of radial basis functions. The author would like also thank the reviewers' comments which are very helpful in improving the revision.

References

[1] S.M. Botros, C.G. Atkeson, Generalization properties of radial basis function, in: R.P. Lippmann et al. (Eds.), Advances in Neural Information Processing Systems 3, Morgan Kaufmann, Los Altos, CA, 1991, 707–713.

[2] D.S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Complex Systems 2 (1988) 321–323.

[3] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for Radial basis function networks, IEEE Trans. Neural Networks 2 (1991) 302–309.

[4] Y.M. Cheung, Helen Z.H. Lai, L. Xu, Adaptive rival penalized competitive learning and combined linear predictor with application to financial investment, Proc. 1996 IEEE/IAFE Conf. on Computational Intelligence for Financial Engineering, CIFEr, New York City, 1996, pp. 141–147.

[5] P.A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[6] L. Devroye, A Course in Density Estimation Birkhauser Publisher, Boston 1987.

[7] R.O. Duda, P.E. Hart,Pattern classification and Scene analysis, Wiley, New York, 1973.

[8] E.J. Hartman, J.D. Keeler, J.M. Kowalski, Layered neural networks with Gaussian hidden units and universal approximations, Neural Comput. 2 (1990) 210–215.

[9] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, G.E. Hinton, Adaptive mixtures of local experts, Neural Comput. 3 (1991) 79–87.

[10] R.D. Jones et. al., Information theoretic derivation of network architecture and learning algorithms, Proc. IJCNN91, Seattle, vol. II, 1991, pp. 473–478.

[11] M.I. Jordan, R.A. Jacobs, Hierarchical mixtures of experts and the EM algorithm, Neural Comput. 6 (1994) 181–214.

[12] M.I. Jordan, L. Xu, Convergence results for the EM approach to mixtures of experts, Neural Networks 8 (9) (1995) 1409–1431.

[13] H. Kälviäinen, P. Hirvonen, L. Xu, E. Oja, Probabilistic and non-probabilistic Hough transforms: overview and comparisons, Image and Vision Comput. 5 (4) (1995) 239–252.

[14] V. Kardirkamanathan, M. Niranjan, F. Fallside, Sequential adaptation of radial basis function neural networks and its application to time-series prediction, in: R.P. Lippmann, J.E. Moo, D.S. Touretzky (Eds.), Advances in Neural Information Processing System 3, Morgan Kaufmann, San Mateo, 1991, pp. 721–727.

[15] B.W. Mel, S.M. Omohundro, How receptive field parameters affect neural learning, in: R.P. Lippmann, J.E. Moo, D.S. Touretzky (Eds.), Advances in Neural Information Processing System 3, Morgan Kaufmann, San Mateo, 1991, pp. 757–763.

[16] J. Moody, J. Darken, Fast learning in networks of locally-tuned processing units, Neural Comput. 1 (1989) 281–294.

[17] S.J. Nowlan, Max likelihood competition in RBF networks, Tech. Rep. CRG-Tr-90-2, Dept. of Computer Sci., Univ. of Toronto, 1990.

[18] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 5 (1993) 305–316.

[19] T. Poggio, F. Girosi, Networks for approximation and learning, Proc. IEEE 78 (1990) 1481–1497.

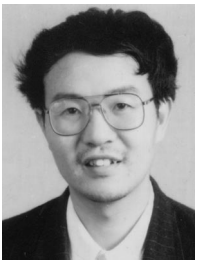
[20] M.J.D. Powell, Radial basis functions for multivariable interpolation: a review, Algorithms for Approximation, in: J.C. Mason, M.G. Cox (Eds.), Clarendon Press, Oxford, 1987.

[21] A. Saha et. al., Oriented non-radial basisv functions for image coding and analysis, in: R.P. Lippmann et al. (Eds.), Advances in Neural Information Processing Systems 3, Morgan Kaufmann, Los Altos, CA, 1991, 728–734.

[22] V.D. Sanchez, Robustization of a learning method for RBF networks, Neural Computing 9 (1) (1995) 85–95.

[23] L. Xu, YING-YANG Machine: a Bayesian–Kullback scheme for unified learnings and new results on vector quantization, Keynote talk, Proc. Int. Conf. on Neural Information Processing (ICONIP95), 30 October–3 November, 1995, pp. 977–988.

[24] L. Xu, Bayesian–Kullback YING-YANG learning scheme: reviews and new results, Proc. Intl Conf. on Neural Information Processing (ICONIP96), 24–27 September, Springer, Singapore, 1996, pp. 59–67.
[25] L. Xu, Bayesian–Kullback YING-YANG Machines for supervised learning, Invited talk, Proc. 1996 World Congress On Neural Networks, 15–18, September, San Diego, CA, 1996, pp. 193–200.
[26] L. Xu, Bayesian Ying-Yang system and theory as a unified statistical learning approach: (I) unsupervised and semi-unsupervised learning, in: S. Amari, N. Kassabov (Eds.), Brain-like Computing and Intelligent Information Systems, Springer, Singapore, 1997. (Invited paper), pp. 241–274.
[27] L. Xu, Bayesian Ying-Yang system and theory as a unified statistical learning approach: (II) From Unsupervised Learning to Supervised Learning and Temporal Modeling and (III) Models and Algorithms for Dependence Reduction, Data Dimension Reduction, ICA and Supervised Learning, *Theoretical Aspects of Neural Computation*: *A Multidisciplinary Perspective* (TANC97), K.W. Wong, I. King and D.Y. Yeung eds, Springer, Singapore pp. 25–60, 1997.
[28] L. Xu, New Advances on Bayesian Ying-Yang learning system with Kullback and non-Kullback separation functionals, Proc. IEEE Int. Conf. on Neural Networks (IEEE-INNS IJCNN97), 9–12 June, Houston, TX, USA, vol. III, 1997, pp. 1942–1947.
[29] L. Xu, Bayesian Ying-Yang machine, clustering and number of clusters, Pattern Recognition Lett., to appear.
[30] L. Xu, Y.M. Cheung, Adaptive supervised learning decision networks for traders and portfolios, Proc. IEEE/IAFE Conf. on Computational Intelligence for Financial Engineering, CIFEr, New York City, 1997, pp. 206–212.
[31] L. Xu, M.I. Jordan, On convergence properties of the EM algorithm for Gaussian mixtures, Neural Comput. 8 (2) (1996) 129–151.
[32] L. Xu, M.I. Jordan, G.E. Hinton, A modified gating network for the mixtures of experts architecture, Proc. WCNN'94, San Diego 2 (1994) 405–410.
[33] L. Xu, M.I. Jordan, G.E. Hinton, in: J.D. Cowan, G. Tesauro, J. Alspector (Eds.), An Alternative Model for Mixtures of Experts, Advances in Neural Information Processing Systems 7, MIT Press, Cambridge, MA, 1996, pp. 633–640.
[34] L. Xu, S. Klasa, A.L. Yuille, Recent advances on techniques static feedforward networks with supervised learning, Int. J. Neural Systems 3 (3) (1992) 253–290.
[35] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning clustering analysis, RBF nets and curve detection, Proc. Int. Joint Conf. on Neural Networks, Beijing, P.R. China, 3–6 November, vol.2, 1992, 665–670.
[36] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net and curve detection, IEEE Trans. Neural Networks 4 (4) (1993) 636–649.
[37] L. Xu, A. Krzyzak, A.L. Yuille, On radial basis function nets and kernel regression: satistical consistency, convergence rates and receptive field size, Neural Networks 7 (4) (1994) 609–628.
[38] L. Xu, E. Oja, Randomized Hough transform (RHT): Basic mechanisms, algorithms and complexities, Computer Vision, Graphics, and Image Processing: Image Understanding 57 (2) (1993) 131–154.
[39] L. Xu, E. Oja, P. Kultanen, A new curve detection method: randomized Hough transform (RHT), Int. J. Pattern Recognition Lett. 11 (1990) 331–338.



Lei Xu (PhD, IEEE Senior member) is currently a professor of Dept. Computer Sci. and Eng. at Chinese Univ. Hong Kong where he joined, since September 1993, as senior lecturer first and then took the current position since October 1996. He is also a professor of National Machine Perception Lab, Peking Univ. since 1992, where he started as a postdoc of Dept. of Math. in 1987 and then became one of ten exceptionally promoted young associate professors of Peking Univ. in 1988. During 1989–1993, he worked as postdoc or senior research associate in several universities in Finland, Canada and USA, including Harvard and MIT. He is a past president of Asian-Pacific Neural Networks Assembly, an associate editor for six renowned international academic journals on neurocomputing, including *Neural Networks*, *IEEE Trans. on Neural Networks*. He has published over 180 papers on neurocomputing, CVPR, Signal Processing, and AI; given over ten keynote/invited/tutorial talks as well as served as their program committee member and session cochairs in international major Neural Networks conferences in recent years, including WCNN, IEEE-ICNN, ENNS-ICANN, ICONIP, IJCNN, NNCM. Also, he was a program committee chair of ICONIP'96. He has received several Chinese national prestigious academic awards, including National Nature Science Award and State Education Council FOK YING TUNG Award, and also an 1995 International Neural Networks Society Leadership Award. He has been listed in 1995 Who's Who in Hong Kong, 1996, 1998 Marquis Who's Who in the World (13th and 14th ed.), American Biographical Institute, the Platinum Record for Exceptional Performance, and the Five Hundred Leaders of Influence (6th ed.), and England Cambridge International Biographical Center The First Five Hundreds (4th ed.) and The Men of Achievement (17th ed.).