Pruning RBF networks with QLP decomposition

EDWIRDE LUIZ SILVA , PAULO LISBOA AND ANDRÉS GONZÁLEZ CARMONA
Departamento de Matemática e Estatística
Universidade Estadual da Paraíba - UEPB
Rua Juvêncio Arruda, S/N Campus Universitário (Bodocongó).
Cep: 58.109 - 790 Campina Grande - Paraíba
BRASIL
School of Computing and Mathematical Sciences
Liverpool John Moores University
Byrom Street, Liverpool, L3 3AF
ENGLAND
Universidad de Granada
Departamento de Estadística e Investigación Operativa
ESPAÑA
http://www.uepb.edu.br
http://www.ilmu.ac.uk
http://www.ugr.es

Abstract: - The radial basis function (RBF) network is the main practical alternative to the multi-layer perceptron for non-linear modeling. This paper describes a methodology to adjust predictions models, calculated from experimental data using regression with Gaussian basis functions reduced by QLP decomposition. After introducing the concepts of linear basis function models and matrix design reduced by QLP decomposition, the method is applied to RBF networks with different choices of the hidden basis function. The QLP method is effective for reducing the network size by pruning hidden nodes, resulting in a parsimonious model which accurate out-of-sample prediction for a sinusoidal test function. Simulation results showed that Gaussian basis functions produced the best results for this bench mark problem.

Key-Words: - Radial Basis Function, Pivoted QLP Decomposition, adjust function, BIC, FPE.

1 Introduction

Radial basis function methods have their origins in techniques for performing exact interpolation of a set of data points in a multi-dimensional space [1]. Radial basis function (RBF) networks with linear outputs are often used in regression problems because they can be substantially faster to train than Multi-layer Perceptrons (MLP).

The least squares principle has been fundamental to data modeling and the training mean square error (MSE) has always played a central role in model structure construction and parameter estimation. In this paper we show how RBFs with reduction neuron from the network decomposition using QLP (a lower diagonal matrix L between orthogonal matrices Q and P [2] can result adjust problems. The performance of the RBF reduction with QLP is compared with basis functions networks Gaussian.

The rest the paper is organized as follows. Section II presents Linear Basis Function Models and section III presents Detection of the Numerical rank of the QLP, and section IV presents Design RBF Neural Regression, and section V reports on the quantitative performance comparisons for the pruned RBF networks, in the section VI simulated data to test the proposed approach to adjust. Finally, section VII offers conclusions from this study.

2 Linear Basis Function Models

2.1 Generalizing linear regression

The RBF mappings discussed so far provide an interpolating function which passes exactly through every data point. We use one specific function sine to illustrate our algorithms. In general RBF mappings are carefully designed to minimize overfitting provide an interpolating function which passes exactly through every data point [4]. The simplest regression function is a linear combination of the input variables

$$z(x,w) = w_o + w_1 x_1 + ... + w_F x_F \qquad (1)$$

where $x = (x_1,...,x_F)^T$. This is well known as linear regression. The key property of this model is that is a linear function of the parameters $w_o,...,w_F$. It is also, however, the linearity as a function of the input variables x_i imposes significant limitations on the ability of the model to fit complex functions. It is common practice in the literature to extend their class of models by considering linear combinations of fixed nonlinear functions of the input variables, of the form

$$z(x,w) = w_o + \sum_{i=1}^{N-1} w_i \theta_i(x) \qquad (2)$$

where $\theta_i(x)$ are known as basis functions with total number parameters in this model will be N. The bias w_o (not to be confused with "bias" in a statistic sense). In general used $w_o = \theta_o(x) = 1$ so that [5]

$$z(x,w) = \sum_{i=1}^{N-1} w_i \theta_i(x) = w^T \theta(x) \qquad (3)$$

where $w = (w_o,..., w_{N-1})^T$ and $\theta = (\varphi_o,..., \varphi_{N-1})^T$.

If the original variables comprise the vector x, then the features can be expressed in terms of the basis functions $\{\theta_i(x)\}$. We assume that target variable t is given by a deterministic function $z(x,w)$ with additive Gaussian noise so that $t = z(x,w) + \varepsilon$, where ε is a zero mean Gaussian random variable with precision (inverse variance) ψ. Thus we have:

$$p(t \mid x, w, \psi) = N(t \mid z(x,w), \psi^{-1}) \qquad (4)$$

Now consider a data set of inputs $X = \{x_1,..., x_N\}$ with corresponding target values $t_1,...,t_N$. Making the assumption that these data points are drawn independently from the distribution (4), we obtain the following expression for the likelihood function, which is a function of the adjustable parameters w and ψ, in the form [5]

$$\ln p(t \mid X, w, \psi) = \prod_{n=1}^{N} N(t_n \mid w^T \theta(x_n), \psi^{-1}),$$ where we have used form (3). Note that in supervised learning problems such as regression (and classification), we are not seeking to model the distribution of the input variables. Thus x will always appear in the set of conditioning variables, and so from now on we will drop the explicit x from expressions such as $p(t \mid x, w, \psi)$ in order to keep the notation uncluttered [5].

Taking the logarithm of the likelihood function, and making use of the standard form for the univariate Gaussian, we have [5]

$$\ln p(t \mid w, \psi) = \sum_{n=1}^{N} \ln N(t_n \mid w^T \theta(x_n), \psi^{-1})$$
$$= \frac{N}{2} \ln \psi - \frac{N}{2} \ln(2\pi) - \psi E_D(w) \qquad (6)$$

Where the sum-of-squares error function is defined by

$$E_D(w) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - w^T \theta(x_n)\}^2 \qquad (7)$$

The maximization of the likelihood function under a conditional Gaussian noise distribution for a linear model is equivalent to minimizing a sum-of-squares error function given by $E_D(w)$. The gradient of the log likelihood function (6) takes the form

$$\nabla \ln p(t \mid w, \psi) = \sum_{n=1}^{N} \{t_n - w^T \theta(x_n)\} \theta(x_n)^T$$

Setting this gradient to zero gives

$$\sum_{n=1}^{N} t_n \theta(x_n)^T - w^T \left(\sum_{n=1}^{N} \theta(x_n) \theta(x_n)^T \right) = 0$$

Solving for w we obtain $w = \left(\Theta^T \Theta\right)^{-1} \Theta^T t$ (8)

Which are knows as the normal equations for the least squares problem. Here Θ is an NxM matrix, called the design matrix, whose elements are given by $\Theta_{nj} = \theta_j(x_n)$, so that

$$\Theta = \begin{pmatrix} \theta_o(x_1) & \theta_1(x_1) & ... & \theta_{M-1}(x_1) \\ \theta_0(x_2) & \theta_1(x_2) & ... & \theta_{M-1}(x_2) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \theta_0(x_N) & \theta_1(x_N) & ... & \theta_{M-1}(x_N) \end{pmatrix}$$

The quantity $\Theta^t \equiv \left(\Theta^T \Theta\right)^{-1} \Theta^T$ is knows in statistic as the Moore-Penrose pseudo-inverse of the matrix Θ [6,7].

3 Detection of the Numerical rank of the QLP

The QLP (a lower diagonal matrix L between orthogonal matrices Q and P [2]) decomposition [3] is computed by applying pivoted orthogonal triangularization to the columns of the matrix design Θ in question to get an upper triangular factor R and then applying the same procedure to the rows of R to get a lower triangular matrix L. The diagonal elements of R are called the R-values of Θ; those of L are called the L-values [2]

Numerical examples show that the L-values track the singular values of Θ with considerable fidelity-far better than the R-values. At a gap in the L-values the decomposition provides orthonormal bases of analogues of row, column, and null spaces provided of Θ. The decomposition requires no more than twice the work required for a pivoted QR decomposition [2,3].

The computation of R and L can be interleaved, so that the computation can be terminated at any suitable point, which makes the decomposition especially suitable for low-rank determination problems. We will call the diagonals of R the R-values of Θ. The folklore has it that the R-values track the singular values well enough to expose gaps in the latter. For example, a matrix design of order 100 was generated in the form:

$$\Theta = QLP + 0.1\sigma_{50}\, randn(100,100) \qquad (9)$$

where L is formed by creating a diagonal matrix (of size 100x100) decreasing geometrically from one to 10^{-3} and setting the last fifty diagonal elements to zero. Thus Θ represents a matrix design of rank 50 perturbed by an error whose elements are one-tenth the size the size of the last nonzero singular value. In Figure 2, the values of $r_{50,50}$ and $r_{51,51}$ shows that there is a well-market gap in the R-values, though not as marked as the gap in the singular values. The QR and SVD showing that the evidence $r_{51,51}$ occurs close to the point of best for QLP decomposition, this is, the QLP shows us a simple gap considered between the ranges $r_{50,50}$ and $r_{51,51}$, and a highlight gap in $r_{26,26}$

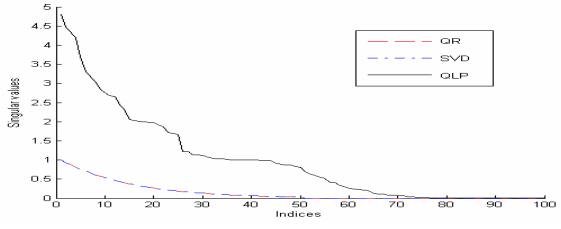

Fig. 1 Decompositions SVD, QR and QLP for the data matrix design Θ

The implementation of the Matlab package permits the QLP: [P,Q,L,pr,pl]=qlp(Θ) to determine the numerical rank of matrix design Θ applied on the Gaussian.

Thus, we have a simple QLP algorithm as follows:

1. **define** matrix design Θ, which consists of Θ (9).
2. **calculate** the orthonormal matrices Q and P which reduce the matrix design Θ to lower diagonal form.
3. **identify the** diagonal of lower-triangular matrix L
4. **sort** the diagonal elements by size.

In the Table 1 shows the computational speed advantage by over an order of magnitude from using QLP decomposition compared with the SVD algorithm.

The speed to resulting the QLP [7,8] matrix design on the training RBF network also makes them attractive for use as a component in more complex model.

Table 1 Comparison between the times required to calculate matrix design Θ using SVD and QR and QLP decompositions

Decompositions	Gau
QLP and QR	0.01
SVD	0.05

Moreover, it is likely that the pivoted QLP decomposition may also provide better approximations to the singular values of the original matrix design. Table 1 show that the QLP and QR for Gaussian matrix design RBF are on average 5 times faster than decomposition SVD.

4 Design RBF Neural Regression

Figure 2 depicts the architecture for a fully connected RBF network. The network consists of n input features x, M hidden units with center C_j and y output. The φ_j are the basis functions, and w_{kj} are the output layer weights. The basis function activations are then calculated using a method which depends on the nature of the function. Suppose at a set of fixed point $x_1,...,x_j$, $\varphi_j = \varphi_j(x)$ can be $\varphi_j(x) = \exp(-\|x-c\|^2 / 2r^2)$.

We shall write the RBF network mapping in the following form:

$$y_n(x) = \sum_{j=1}^{M} w_{nj}\varphi_j(x) + w_{no}, \quad n = 1,2,...,L \quad (11)$$

Finally the network outputs are calculated by

$$y_n(x) = \sigma\left[\sum_{j=1}^{M} w_{nj}\varphi_j(x) + w_{no}\right], \quad n = 1,2,...,L$$

where σ_n is a linear transfer function.

4.1 Equivalence to regularization

In the regularization approach to function approximation, the over-fitting problem is addressed differently in that explicit smoothness constraints are folded into the optimization process. The idea of adding a regularization term to an error function (7) in order to control over-fitting, so that the total error function to be minimized takes the form [5]

$$E_D(w) + \lambda E_W(w) \quad (12)$$

Where λ is the regularization coefficient that controls the relative importance of the data-dependent error $E_D(w)$ and the regularization term $E_W(w)$. If consider the sum-of-squares error function given by

$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w^T\varphi(x_n)\}^2$, Then the error function becomes [4]

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - w^T\varphi(x_n)\}^2 + \frac{\lambda}{2}w^T w \quad (14)$$

This particular choice of regularization on this paper it be $\lambda = 10^{-3}$, this value that relative importance of the data-dependent error $E_D(w)$ and the regularization term $E_W(w)$ before and after QLP decomposition. Specifically, setting the gradient of (14) with respect to w to zero, and solving for w as before, we obtain

$$w = \left(\lambda I + \Phi^T\Phi\right)^{-1}\Phi^T z \quad (15)$$

This represents a simple extension of the least squares solution [6].

Regularization λ allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity. There are different class smoothers, and the RBF has a particularly strong relationship to the class of statistical regression models known as kernel smoothers.

We write equation (13) emphasizing the dependence of reduction of QLP decomposition with choice the hidden units of the basis function,

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w^T\phi(x_n)\,|\,qlp\}^2 \quad (16)$$

where y_n is the target and $(\cdot\,|\,qlp)$ number is the value observed for a particular subset matrix design $N \le M$

4.2 Proposed reduction RBF to identification

This paper the problem consists of one input variable X and one target variable T with data generated by sampling X at equal intervals [0,10] and then generating target data by computing an rapidly changing continuous function $z(x) = 0.8.\exp(-0.2x).\sin(10.x) + n(x), \quad 0 \le x \le 10$ where x is obviously scalar and $n(x)$ is Gaussian noise with a maximum possible amplitude equal to 1: $n(x) \sim N(0,1)$.

The speed of training RBF networks also makes them attractive for use as a component in more complex models [4]. Each iteration with a RBF network requires a single matrix inversion. The first process is the creation of the matrix design Φ composed with inputs and centre. Initially the design matrix has constant radius and centre positioned at each data point. In comparison others values arbitrary of radio, the value 0.1 is a good choice.

5 Performance estimation

5.1 BIC Shwartz Bayesian Information Criterion

This generic function calculates the Bayesian information criterion, also known as Schwarz's Bayesian criterion (SBC), for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula $-2\log\_likehood + npar.\log(nobs)$, where $npar$ represents the number of parameters and $nobs$ the number of observations in the fitted model.

The number of parameters here considered is $\tau = p - trace(P)$ [11], where $p = 100$ is the number of rows in the design matrix Φ, and $P = I_p - \Phi A^{-1}\Phi^T$ is projection matrix, where $A = (\Phi^T\Phi + IU^TU)$ is the variance matrix, U is the upper triangular transform of

dimension 100x100 and $I = \lambda = 10^{-3}$ parameters of regularization.

On the one hand, large number of parameters (and of corresponding basis function) ensures good quality of data description but, on the other hand, it complicates the model excessively. The BIC criterion is defined as

$BIC = N_{bic}(z^T P^2 z)/p$

where $N_{bic} = \dfrac{(p + (\log(p) - 1).\tau)}{p - \tau}$ is the number of parameters and y is input training data [9].

In this paper the model is the best due to the fact that it shows the least BIC value. The value of the BIC statistic suggests also that the errors are not correlated.

5.2 Final Prediction Error (FPE)

The FPE is a network performance function is defined as $FPE = N_{fpe}(z^T P^2 z)/p$, where: $N_{fpe} = \dfrac{p+g}{p-g}$ is the number of parameters [12].

6 Simulation results

There are 1000 testing data (x_i, z_i) with randomly distributed in the range (0,10) . Here we generate 1000 data sets, independently from the sinusoidal curve z. The data set are indexed by $l = 1,...,L$, where L = 100, and for each data set we fit a model with 100 Gaussian regularized by $\lambda = 0.001$ to give a prediction function as show in Figures 3 and 4.

The Gaussian basis function was used with a kernel $r^2 = 0.1$. All the 100 training data points were used as the candidate RBF center set for c.

We assume that these input samples x_i have a uniform distribution. The design matrix from the input data, centre positions and radial factors has size of 100x100. We assume that $w = inv(\Phi'\Phi)\Phi' y$ and $output = \Phi_t w$ with 100 neurons have been obtained. The network output for an input x_i is given by

$$y_1(x) = \sum_{j=1}^{N} w_j \varphi_j(x).$$

If using all data the RBF would be close to of noise, and with an error of value 36.42 for 100 neurons. In the Figure 1 shows us that the decomposition QLP has approximately 51 neurons.

The resulting in the Fig. 3 shows us that it's clear that the network with 100 hidden units over fits the training data. It's shows that if consider all hidden units the error squared mean is bigger. In many case, however, it occur that to can construct an approximation mediate the sum of approximation local requiring a high number of neurons, which could influence negative in the capacity of generalization of the RBF.

The user can now input a reduced number of hidden units after inspecting this Fig. 1, and the network is then retrained in the number of hidden units**.** We can ask how a hidden unit is true. One method to choose this number of hidden is to use the minimum value of the BIC criterion (Table 4) shows us that 51 hidden units are better.

The Fig. 4, show us the model prediction $\hat{z}(x)$ (black points) and system output $z(x)$ RBF Gaussian (red points) and sampled data (green crosses).shows a sequence of 1000 values belonging to the test set and their predicted values, corresponding to the model generated in experiment reduced matrix design by QLP decomposition. The Fig. 4 consists of a linear combination of 51 radial functions, with smaller structure complexity than that of the model reported in the Fig. 3. It can be seen that the two sequences are partial virtually indistinguishable.

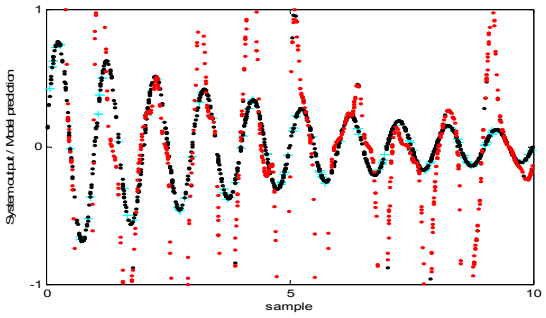

Fig. 3 RBF trained with 100 hidden units. The model prediction $\hat{z}(x)$ (black points) and system output $z(x)$ RBF Gaussian (red points) and sampled data (green crosses).

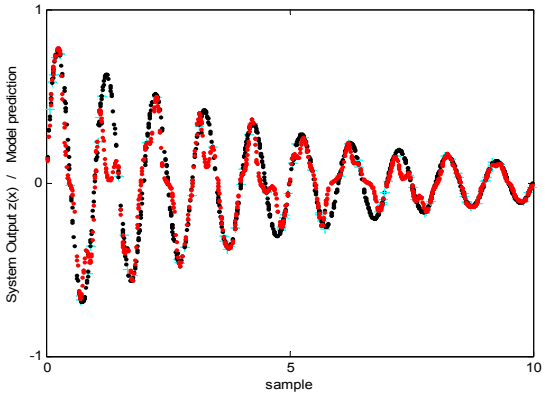

Fig. 4 Examples of the predictive for a model consisting of 51 Gaussian bases function using the z(x).

In the Fig. 5 depicts the model prediction $\hat{z}(x)$ and the prediction error $\varepsilon(x) = z(x) - \hat{z}(x)$ for the 51 select neurons.

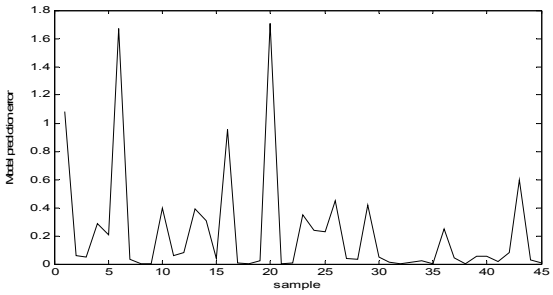

Fig.5 Model prediction error $\varepsilon(x)$

In the table 3, shows us the error squared mean with the different arquitecture about the training set and testing set. A good resulted is obtained with 51 neurons and, however, the hidden units in the network could be a significant parameter on the resolution of the problem using RBF network, then the error obtained after to get the convergence them are enough different. If consider all 100 neurons the mse would be 9.56 for training and 2.9267 for testing.

Table 3 The error squared mean of method RBF Gaussian reduced by QLP.

RBF Gau	Error train	Error test
100	9.5629	2.9267
51	**0.0013**	**0.012**
45	0.0021	0.009
40	0.0036	0.016
35	0.0070	0.023
26	0.012	0.029

Table 4 Numerical results for BIC and EPF after reduction of matrix design.

N.neuron	BIC		FPE	
	Φ	Φ_{QLP}	Φ	Φ_{QLP}
100	0.0003	0.0003	0.0001	0.0001
51	-	**0.007**	-	**0.0040**
45	-	0.010	-	0.0057
40	-	0.014	-	0.0080
35	-	0.026	-	0.0157

*FPE= final prediction error and BIC=Schwartz Bayesian Information Criterion

7 Conclusions

This paper has introduced an automatic model construction algorithm for RBF networks for regression. The main investigations in this study concern the pruning of RBF hidden nodes by QLP using different radial basis function types. The experimental results demonstrate the potential of our proposed techniques, indicating that QLP is effective when the RBF centres are not adjusted and the regularization parameters are kept fixed. Fig. 3 and 4 shows the RBF network predictions before and after the reduction in the number of hidden units It is clear that the 100 hidden units overfits the training data, while that with 51 hidden units will generalize significantly better. We fitted $z(x)$ function to data sets by RBF reduced by QLP. We also showed that mean square error of selection RBF Gaussian for training and validation were 0.0013 and 0.0012 respectively. Other point that deserves further study is the use of other basis functions.

References:

[1] Powell M.J.D. *Radial Basis Function for Multivariate Interpolation.* A Review. In: Algorithms for Approximation. J.C. Mason and M.G. Cox, (eds). Clarendon Press. Oxford, U.K., 1987.

[2] Stewart G.W, *Matrix Algorithm: Basic Decompositions*. SIAM Publications. Philadelphia, U.S.A., 1998

[3] Stewart G.W, On an Inexpensive Tringular Approximation to the Singular Value Decomposition. *Department of Computer Science and Institute for USA*, 1998, pp.1-16.

[4] Ian T. Nabney. NETLAB. *Algorithms for Pattern Recognition. Birmingham*, U.K., 2002.

[5] Bishop, C.M., *Pattern Recognition and Machine Learning*. Computer Science. Ed. Springer, U.K., 2006.

[6] Rao C.R. and S.K. Miltra. *Generalized Inverse of matrices and Its Applications.* Wiley. 1971.

[7] Edwirde L.S and Paulo Lisboa. Comparison of Wiener Filter solution by SVD with decompositions QR and QLP. Conference WEAS Corfu Greece, 2007.

[8] Edwirde L.S and Paulo Lisboa. Differentiating features for the Weibull, Gamma, Log-normal and normal distributions through RBF network pruning with QLP. Conference WEAS Corfu Greece, 2007.

[9] G. Schwarz. Estimating the dimension of a model. Annals of Statistic, Vol.6, 1978, pp 461-464.

[10] Guang-Bin Huang and P. Saractchandran. An Efficient Sequential Learning Algorithm for Growing and Pruning RBF (GAP-RBF) Networks. IEEE, Singapure, pp. 2284-2291, 2004

[11] D.J.C. MarKay. Bayesian interpolation. Neural Computation, Vol.4, N0.3, 1992, pp. 415-447.

[12] C. Mallows. Some comments on C_p. Technometrics, No.15, 1973, pp. 661-675.