Structured parameter optimization method for the radial basis function-based state-dependent autoregressive model

H. Peng†∥*, T. Ozaki‡, V. Haggan-Ozaki§ and Y. Toyoda¶

An off-line structured nonlinear parameter optimization method (SNPOM) for accelerating the computational convergence of parameter estimation of the radial basis function-based state-dependent autoregressive (RBF-AR) model is proposed. Using the method, all the parameters of the RBF-AR model may be optimized automatically and simultaneously. The proposed method combines the advantages of the Levenberg–Marquardt algorithm in nonlinear parameter optimization and the least-squares method in linear parameter estimation. Case studies on two complex time series and a nonlinear chemical reaction process show that the proposed parameter optimization method exhibits significantly accelerated convergence when compared with the classic version of the Levenberg–Marquardt algorithm, and to some hybrid algorithms such as the evolutionary programming algorithm.

1. Introduction

In nonlinear system modelling or nonlinear function approximation problems, neural networks have been widely used, for which the radial basis function (RBF) network offers a viable alternative due to its properties of simple topological structure and fast learning. Although the general approximation capabilities of the RBF network provide a theoretical foundation for representing complex systems (Chen et al. 1990), the RBF network model usually requires many RBF centres to obtain satisfactory approximation precision, so there may be some difficulty in parameter estimation.

A large class of nonlinear systems satisfying necessary smoothness conditions can be characterized by the nonlinear autoregressive (NAR or NARX or NARMAX) model (Chen and Billings 1989), of which the state-dependent autoregressive model (Priestley 1980) is a general version that includes, as special cases, many commonly used nonlinear models such as the bilinear model (Granger and Andersen 1978), the exponential autoregressive model (Ozaki and Oda 1978), the parametric Volterra model (Haber and Keviczky 1999) and the threshold model (Tong 1983), as well as standard linear models. However, the estimation of the state-dependent autoregressive model is difficult in actual applications because its coefficients do not have a concrete description.


The identification of the RBF-AR model is a major problem encompassing the determination of the model order, the selection of the centres and the scaling factor of RBF networks, and the estimation of all the weights of RBF networks corresponding to each functional coefficient of the model. Three approaches have been taken to this problem. The first approach...
optimize all the parameters of the model by using a nonlinear parameter optimization algorithm such as the Levenberg-Marquardt method (LMM) (Marquardt 1963). This kind of method is generally based on an exhaustive search in the solution space and therefore requires extensive computation. The second method is first to select the basis function centres by setting them equal to a subset of the input vectors either randomly or using some algorithm (Moody and Darken 1989, Chen et al. 1991, Ozaki et al. 1997), and then to estimate the linear weights by the standard least squares method. This approach quickly yields a solution, but clearly that solution cannot be optimum. A third approach (Shi et al. 1999), like the LMM, optimizes all the model parameters simultaneously, but is an automatic estimation method that also accelerates the computational convergence rate of the search process. The automatic estimation method optimizes all the parameters of the RBF-AR model, by a hybrid search algorithm based on evolutionary programming and the least-squares method, but their paper did not provide any illustration or comparison about the convergence rate of the search process.

In fact, as may be seen in Section 2, the RBF-AR or RBF net model belongs to a class of nonlinear model whose parameters may be separated into both linear and nonlinear parts. Golub and Pereyra (1973), Ruano et al. (1992, 2001) and Ferreira and Ruano (2000) proposed an estimation algorithm for general off-line least-squares fit of this kind of nonlinear model. The algorithm first replaced the linear parameters by their least-squares value conditional on the unknown nonlinear parameters in the quadratic objective function and then obtained a modified objective function depending only on the nonlinear parameters. The modified objective function was then minimized. In this way, the number of parameters to be estimated is much reduced. Although this approach may obtain better convergence, the price is a greater complexity in the computation of the objective function derivatives. Furthermore, in some cases large spikes of the objective function may happen in the search process.

In Section 3, we propose an off-line structured nonlinear parameter optimization method (SNPOM) for estimating the RBF-AR model or other similar models (such as an RBF network), where the model parameters to be estimated can be separated into two sets, one consisting of all the linear parameters (where the model output is linear with respect to the parameters), the other consisting of all the nonlinear parameters (where the model output is nonlinear in the parameters). The proposed optimization method is based on the LMM and the standard least-squares method (LSM), and provides an automatic estimation for all the parameters. This method uses the advantages of the LMM for nonlinear parameter optimization and the LSM for linear parameter estimation. It is different to the Golub and Pereyra (1973) and Ruano et al. (1992) methods in that it directly minimizes the quadratic objective function with respect to all the parameters and it does not bring in any extra complexity in the computation of the objective function derivatives.

As will be shown in Section 4, case studies on two complex time series and a nonlinear chemical reaction process show that the proposed nonlinear parameter optimization method exhibits significantly accelerated convergence compared with some conventional algorithms such as the LMM, and to some hybrid algorithms such as the evolutionary programming algorithm (EPA) (Shi et al. 1999).

2. Generalized RBF-AR model with an exogenous variable

2.1. State-dependent AR model

We are interested in systems that may be characterized by a NARX model as follows

\[ y(t) = f(X(t-1)) + e(t) \quad (1a) \]
\[ X(t-1) = [y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u)]^T, \quad (1b) \]

where \( y(t) \) is the output, \( u(t) \) is the input and \( e(t) \) is a white noise. Various types of function have been applied to approximate the unknown nonlinear map \( f(\cdot) \) that maximizes the likelihood of the model, for which the state-dependent AR model (Priestley 1980) may be regarded as a general description (in Priestley's state-dependent AR model, \( u(t) = 0 \)):

\[ y(t) = \phi_0(X(t-1)) + \sum_{i=1}^{p_y} \phi_{y,i}(X(t-1))y(t-i) \]
\[ + \sum_{i=1}^{p_u} \phi_{u,i}(X(t-1))u(t-i) + e(t), \quad (2) \]

where \( X(t-1) \) is the state vector at time \( t \). In model (2), \( \phi_0(X(t-1)) \), \( \{\phi_{y,i}(X(t-1)), i = 1, 2, \ldots, p_y\} \) and \( \{\phi_{u,i}(X(t-1)), i = 1, 2, \ldots, p_u\} \) are called the state-dependent functional-coefficients of the model. The basic idea of the state-dependent AR model (2) is to achieve a local linearization of the general NARX model (1) by introducing a locally linear AR model with state-dependent coefficients. Although the state-dependent model (2) provides a useful framework for general nonlinear system modelling, the problem is how to specify its functional-coefficients.
2.2. Generalized RBF-AR model

In fact, from (2) it may be seen that specifying the state-dependent coefficients of the model is actually a problem of function approximation from a multi-dimensional input space $X$ to a one-dimensional scalar space $\varphi_i = \phi_i(X)$. An efficient approach to solving the problem above, without any loss of generality, is by the neural networks approximation. In this paper, RBF networks are used to approximate the state-dependent coefficients of the model (2) since they are excellent for approximating nonlinear functions. Moreover, the locality of the basis functions makes RBF networks very suitable for learning local variations. Gaussian RBF networks are selected for the approximation of the coefficients of model (2), and the model thus derived is called the generalized RBF-AR model given by:

$$y(t) = \phi_0(X(t-1))$$

$$\quad + \sum_{i=1}^{p_r} \phi_{x,i}(X(t-1))y(t-i)$$

$$\quad + \sum_{i=1}^{p_u} \phi_{u,i}(X(t-1))u(t-i) + e(t)$$

$$\phi_0(X(t-1)) = c_0^0 + \sum_{k=1}^{m} c_0^k \exp(-\lambda_k \|X(t-1) - Z_k\|^2_2),$$

$$\phi_{x,i}(X(t-1)) = c_{i,0}^0 + \sum_{k=1}^{m} c_{i,k}^0 \exp(-\lambda_k \|X(t-1) - Z_k\|^2_2),$$

$$\phi_{u,i}(X(t-1)) = c_{i,0}^r + \sum_{k=1}^{m} c_{i,k}^r \exp(-\lambda_k \|X(t-1) - Z_k\|^2_2),$$

$$Z_k = (z_{k,1}, z_{k,2}, \ldots, z_{k,m})^T, j = y, u$$

$$y(t) = \theta_0 + \sum_{k=1}^{p} \theta_i \exp(-\lambda_k \|X(t-1) - Z_k\|^2_2) + e(t).$$

(4)

The generalized RBF-AR model (3) has a regressive-style basic structure. In fact, since model (3) includes a component like equation (4), model (3) may be regarded as a more general nonlinear model than the RBF neural network. Additionally, model (3) may be also regarded as an expansion of the classic ExpAR model (Ozaki 1985), which has only one centre. If $u(t) = 0$ in model (3), it becomes a time series RBF-AR model (Vesin 1993).

In actual applications of the RBF neural networks, a large number of RBF centres are usually necessary to obtain a satisfactory fit to a complicated nonlinear system, resulting in the ‘curse of dimensionality’, which vexes many complex problems. However, in the generalized RBF-AR model (3), the complexity of the model is partially dispersed into the AR part, so the number of RBF centres in the model does not need to be too large. Accordingly, as will be seen in the case studies, the advantage of using the RBF-AR model is that the computational load of the model parameter optimization can be largely reduced, compared with the estimation of RBF networks of the same model complexity. Another benefit of using the RBF-AR model (3) is that the local linearization of the system at any operating-point may easily be obtained, but this cannot be done using RBF networks or other nonlinear models such as the parametric Volterra or Hammerstein models.

3. Structured nonlinear parameter optimization method (SNPOM)

We propose here an off-line unconstrained SNPOM for the parameter estimation of a class of nonlinear models whose parameters can be separated into two sets: one including all the linear coefficients (such as the linear weights of the RBF-AR model or RBF networks), the other including all the nonlinear parameters (such as the RBF centres and scaling parameters of the RBF-AR). The method is based on the LMM implemented in the Matlab Optimization Toolbox and the LSM.

The main approach of the SNPOM is to separate the parameter search space into two subspaces: the linear and nonlinear parameter subspaces. The search method centres on the optimization in the non-linear subspace. At each iteration of the optimization process, the search in the nonlinear (or linear) subspace is executed based on the result of the previous search in the linear (or nonlinear) subspace. The search in nonlinear subspace uses a method similar to the LMM, and
the search in linear subspace uses the LSM. This method is implemented as follows.

3.1. Parameter classification

For the generalized RBF-AR model (3), the set of linear parameters is:

\[
\theta_L = \{ c_i^0, c_i^j, c_i^k | i = 0, 1, \ldots, m; j = 1, 2, \ldots, p_i; k = 1, 2, \ldots, p_u \} \in \mathbb{R}^{(m+1)x(p_i+p_u+1)} \]

and the set of non-linear parameters is:

\[
\theta_N = (\lambda_1, \lambda_2, \ldots, \lambda_m, Z_1^T, Z_2^T, \ldots, Z_m^T)^T \in \mathbb{R}^{m+m \times n_X}.
\]

For the RBF network (4), the set of linear parameters is:

\[
\theta_L = (\theta_0, \theta_1, \ldots, \theta_p)^T \in \mathbb{R}^{p+1}
\]

and the nonlinear set of parameters is

\[
\theta_N = (\lambda_1, \lambda_2, \ldots, \lambda_p, Z_1^T, Z_2^T, \ldots, Z_p^T)^T \in \mathbb{R}^{p+p \times n_X}.
\]

In the general case, model (1) can be rewritten as

\[
y(t) = f(\theta_L, \theta_N, X(t-1)) + e(t)
\]

or

\[
y(t) = \Phi(\theta_N, X(t-1))^T \theta_L + e(t),
\]

where \( \theta_N \) is a vector including all nonlinear parameters, \( \theta_L \) is a vector including all linear parameters and equation (10) is the regression form of (9), which is linear with respect to \( \theta_L \).

3.2. Initialization

Choose the order of the model. The approach for choosing a suitable model order is presented in Section 3.4. For the RBF-AR model (3), the order is \( p_p, p_u, m \) and \( n_X \). For the RBF network (4), the order is \( p \) and \( n_X \). Select an initial value \( \theta_0 \) appropriate for the data. For the generalized RBF-AR model (3) or the RBF network (4), a useful choice procedure for initial values of the nonlinear parameters is to choose a subset of the input vector \( X(t-1) \) as the initial centres \( Z_k (k=1, 2, \ldots, m) \). The initial scaling parameters \( \lambda_k (k=1, 2, \ldots, m) \) may be selected as:

\[
\begin{align*}
\lambda_k &= -\log \varepsilon_k / \max \{ ||X(t-1) - Z_k||_2 \} \\
\varepsilon_k &\in [0.1 \sim 0.0001]
\end{align*}
\]

so that the linear parameters become bounded and stable when the signal \( X(t-1) \) causes the centres to move far away.

Fixing \( \theta_N \) and estimating \( \theta_L^0 \) by the LSM, yields:

\[
\theta_L^0 = \left( \sum_{i=0}^M \Phi_{i,0} \Phi_{i,0}^T \right)^{-1} \sum_{i=0}^M \Phi_{i,0} \tilde{y}(t),
\]

where \( \{ \tilde{y}(t), \tilde{X}(t-1) \}_{i=1}^{\tau+1} \) is the measured data set, \( \tau \) is the largest time lag of the variables in \( X(t-1) \) and \( M \) is the number of data points.

3.3. Parameter optimization

Take the sum of squares of the model residuals as the objective function:

\[
V(\theta_N, \theta_L) = \frac{1}{2} ||F(\theta_N, \theta_L)||_2^2
\]

or

\[
F(\theta_N, \theta_L) = \begin{bmatrix}
f(\theta_L, \theta_N, \tilde{X}(\tau)) - \tilde{y}(\tau + 1) \\
f(\theta_L, \theta_N, \tilde{X}(\tau + 1)) - \tilde{y}(\tau + 2) \\
\vdots \\
f(\theta_L, \theta_N, \tilde{X}(M-1)) - \tilde{y}(M)
\end{bmatrix}
\]

The parameter optimization problem is then to compute:

\[
(\hat{\theta}_N, \hat{\theta}_L) = \arg \min_{\theta_N, \theta_L} V(\theta_N, \theta_L).
\]

Letting \( k = 0, 1, 2, \ldots, k_{\text{max}} \) be the current iteration step, denote the Jacobian matrix of \( F(\theta_N, \theta_L) \) with respect to \( \theta_N \) at \( k \) as:

\[
J(\theta_N^k) = \left( \frac{\partial F(\theta_N, \theta_L)}{\partial \theta_N} \right)^T |_{\theta_N^k, \theta_L^k}
\]

3.3.1. Updating the nonlinear parameter set

The updating algorithm of the nonlinear parameter set \( \theta_N^k \) in the SNPOM is designed to be:

\[
\theta_N^{k+1} = \theta_N^k + \beta_{k+1} d_{k+1},
\]

where \( d_{k+1} \) is the search direction, \( \beta_{k+1} \) is a scalar step length parameter measuring the distance moved towards the minimum. To make the search process robust, the LMM, a demonstrably effective algorithm, is used to determine \( d_{k+1} \). Using this method, the \( d_{k+1} \) in (15) is obtained from a solution of the linear set of the equations:

\[
[J(\theta_N^k)^T J(\theta_N^k) + \gamma_{k+1} I]d_{k+1} = -J(\theta_N^k)^T F(\theta_N^k, \theta_L^k),
\]

where the scalar \( \gamma_{k+1} \) controls both the magnitude and direction of \( d_{k+1} \). When \( \gamma_{k+1} \) tends to zero, \( d_{k+1} \) will tend towards the Gauss–Newton direction. As \( \gamma_{k+1} \) tends to infinity, \( d_{k+1} \) tends towards the steepest descent direction. Therefore, the search direction \( d_{k+1} \) for the optimization of the nonlinear parameter set is a cross between the Gauss–Newton direction and the steepest descent so that the robustness of the search process is increased.
In fact, the search direction \( \mathbf{d}_{k+1} \) is not only determined by the information relative to the nonlinear parameter set \( \theta_N \), but also depends on the information with relation to the linear parameter set \( \theta_L \). After determining \( \gamma_{k+1} \) in (16), \( \mathbf{d}_{k+1} \) is easily computed from equation (16) giving:

\[
\mathbf{d}_{k+1} = -\mathbf{J}(\theta_N^k)^T \mathbf{J}(\theta_N^k) + \gamma_{k+1} \mathbf{I}^{-1} \mathbf{J}(\theta_N^k)^T \mathbf{F}(\theta_N^k, \theta_L^k).
\]

(17)

### 3.3.2. Updating \( \gamma_{k+1} \)

The size of \( \gamma_{k+1} \) is updated in accordance with the measure of the effectiveness of the Gauss–Newton method and the linearity of the problem at each iteration (Coleman et al. 1999). The measure is represented by the difference between a linearly predicted objective function \( V_p^{k+1} \) and a cubically interpolated estimate of the minimum \( V_{k+1} \). The linearly predicted objective function is calculated from:

\[
V_p^{k+1} = \mathbf{F}(\theta_N^k, \theta_L^k) - \mathbf{J}(\theta_N^k)^T \mathbf{J}(\theta_N^k) \mathbf{d}_k,
\]

\[
\gamma_{k+1} = \frac{1}{2} (\gamma_{k+1})^T \mathbf{F}(\theta_N^k, \theta_L^k). \]

An uncorrected estimate \( \hat{V}_{k+1}^{*} \) and a step length parameter \( \beta^* \) are obtained by cubically interpolating the points \( V(\theta_N^k, \theta_L^k) \) and \( V(\theta_N^k, \theta_L^{k-1}) \) (Coleman et al. 1999). The required term \( V_{k+1}^{*} \) and corrected step length parameter \( \beta^* \) may be obtained as follows:

(A1) if \( \hat{V}_{k+1}^{*} > V(\theta_N^k, \theta_L^k) \), then \( V_{k+1}^{*} = 0.9 V(\theta_N^k, \theta_L^k) \),

else \( V_{k+1}^{*} = \hat{V}_{k+1}^{*} \);

(A2) if \( \mathbf{J}(\theta_N^k)^T \mathbf{F}(\theta_N^k, \theta_L^k) \mathbf{d}_k < 0 \),

then

- if \( \beta^* < \beta_k \), then \( \beta^* = |2\beta_k| \),
- else \( \beta^* = |\beta_k| \);

else

- if \( \beta_k > 0.9 \), then \( \beta^* = \min \{1, |\beta^*|\} \),
- else \( \beta^* = \beta_k \).

In the above, \( \beta^* \) is the estimated step to the minimum \( V_{k+1}^{*} \). If \( V_p^{k+1} > V_{k+1}^{*} \), then \( \gamma_{k+1} \) is reduced so that the search direction approaches the Gauss–Newton direction, otherwise it is increased so that the search direction approaches the steepest descent direction as follows:

(B1) if \( V_p^{k+1} > V_{k+1}^{*} \), then \( \gamma_{k+1} = -\frac{\gamma_k}{1 + \beta^*} \),

else \( \gamma_{k+1} = \gamma_k + \frac{V_{k+1}^{*} - V_p^{k+1}}{\beta^*} \).

### 3.3.3. Searching for \( \beta_{k+1} \) and updating the linear parameters simultaneously

Following the update of \( \gamma_{k+1} \), a new search direction \( \mathbf{d}_{k+1} \) is obtained by (17). Subsequently, a step length of \( \beta_{k+1} \) in (15) is then taken in the direction \( \mathbf{d}_{k+1} \), which is obtained by a line search procedure similar to the cubic polynomial interpolation and extrapolation method (Coleman et al. 1999). The optimization calculation continues on the search for \( \theta_N^{k+1} \) at each iteration. In the line search procedure, the best step length \( \beta_{k+1} \) is usually found after several (minor) iterations, and at each minor iteration, after computing an intermediate \( \theta_N^{k+1} \) from (15), the linear parameters \( \theta_L^{k+1} \) relative to \( \theta_N^{k+1} \) are updated immediately by the LSM, thus:

\[
\theta_L^{k+1} = \left[ \sum_{t=1}^{M} \Phi(t_{i,k+1}) \Phi^T(t_{i,k+1}) \right]^{-1} \sum_{t=1}^{M} \Phi(t_{i,k+1}) \vec{y}(t).
\]

(18)

After several (minor) iterations to search for the best \( \beta_{k+1} \), the line search procedure for determining the step length \( \beta_{k+1} \) ensures that

\[
V(\theta_N^{k+1}, \theta_L^{k+1}) < V(\theta_N^k, \theta_L^k)
\]

(19a)

at each major iteration (to search for \( \theta_N^{k+1} \) and \( \theta_L^{k+1} \)). Therefore, \( \theta_N^{k+1} \) and \( \theta_L^{k+1} \) should give a smaller value of \( V \) at the \( k + 1 \)th major iteration. If the search for \( \beta_{k+1} \) continues through a limited number of iterations and condition (19a) is still not adequately satisfied, that is, the decrease of \( V \) is still less than a very small pre-specified positive number \( \eta \), i.e.

\[
V(\theta_N^k, \theta_L^k) - V(\theta_N^{k+1}, \theta_L^{k+1}) \leq \eta
\]

(19b)

then the search process terminates. Obviously, (19b) is the condition for stopping the search.

**Remark 1:** In the SNPOM proposed above, the linear parameters \( \theta_L \) must be re-estimated using formula (18), which is used to adjust the search direction and step length to ensure that the objective function decreases with respect to all parameters, not only with respect to the nonlinear part \( \theta_N \) at each iteration. For computing efficiency, the SNPOM is much better than general methods of optimizing all the parameters regardless of parameter type, especially when the number of linear parameters is greater than that of nonlinear parameters in a model, because in the SNPOM the linear parameters can be easily obtained using (18) only once, whereas the nonlinear parameters have to be estimated using the iteration algorithm. In general, the more nonlinear parameters, the more search iterations and computation burden will be needed.

**Remark 2:** The difference between the SNPOM and the Golub and Pereyra (1973) method (GPM) is that the SNPOM directly minimizes the objective function (13) with respect to all the parameters simultaneously, whereas the GPM minimizes a modified appropriate objective function depending only on \( \theta_N \). This results in a heavier computational load because of the greater complexity of the derivative \( \mathbf{J}(\theta_N^k) = \{\partial \mathbf{J}(\theta_N^k, \theta_L^k) / \partial \theta_N^k\}_\theta \), since \( \theta_L \) is a function with respect.
to $\theta_N$ in the GPM, but in the SNPOM the $\theta_L$ is fixed in the computation of $J(\theta_N^k)$, i.e., $J(\theta_N^k) = \left[ \partial F(\theta_N, \theta_L) / \partial \theta_N \right]^T_{\theta_N, \theta_L}$. 

**Remark 3:** The classical LMM belongs to a restricted step type of algorithm, which in this case would set $\beta_{k+1} = 1$ in (15), leaving $\gamma_{k+1}$ in (16) to determine both the search direction and the updating amplitude of $\theta_N^{k+1}$. On occasion, in problems where the correlations among the parameter-estimates are extremely high (> 0.99), it can happen that $\gamma_{k+1}$ will increase to unreasonably large values (Marquardt 1963), causing a cessation of the parameter updating. For this reason, Marquardt also proposed a revised method using a parameter-updating rule similar to (15). In the SNPOM, $\gamma_{k+1}$ in (16) mainly determined the search direction, and $\beta_{k+1}$ in (15) determined the parameter updating amplitude. This approach is also used in the well-known steepest descent method and Gauss–Newton method, and may be used to enhance the efficiency and reliability of the search process.

**Remark 4:** For the RBF-AR model (3) or the RBF network (4), the scaling parameters $\lambda_k (k = 1, 2, \ldots, m)$ may also be determined by a heuristic approach during the search process. In this way, $\hat{\theta}_N^{k+1}$ includes only the RBF centres, and $\lambda_k$ is also computed by (11) after computing $\hat{\theta}_N^{k+1}$ at a search iteration of $\beta_{k+1}$, in order to ensure that the term $\exp(-\lambda_k \|X(t-1) - \hat{Z}_k\|^2)$ in the model approaches zero when the state $X(t-1)$ is far away from the centre $\hat{Z}_k$, which ensures that the linear parameters of the model are bounded.

3.4. *Determining the order of the model*

The order of the model (9) or (10) may be determined by using AIC (Akaike 1974) and the model dynamics. The SNPOM for model (10) is repeated using different orders, and the final model may be found by comparing AIC values and the estimated model dynamics. For the RBF-AR model (3) and the RBF network (4), the AIC is defined as follows:

$$AIC = M \log \hat{\sigma}^2 + 2(s + 1), \quad M \gg p,$$

where $\hat{\sigma}^2$ is the model residual variance after fitting the model, $M$ is the number of data points, $p$ is the largest order of the regression part and $s$ is the total number of parameters to be estimated.

4. *Case studies*

4.1. *Identification of the nitrogen oxide (NOx) decomposition process*

The purpose of the NOx decomposition process control in thermal power plants is to reduce the NOx concentration in flue gas from the plant boilers in order to protect the environment. The process has nonlinear dynamics dependent on the power load demand of the plant, and the nonlinear characteristics are mainly due to variation in gains with load. The Hammerstein model (Haber and Keviczky 1999) is usually used to describe the dynamics with nonlinear static gain. We use an RBF-AR(p,m,n) model represented by equation (20) to characterize the dynamics of the NOx decomposition process as follows:

$$y(t) = \phi_0(X(t - 1)) + \sum_{i=1}^{p} \phi_{x,i}(X(t - 1))y(t - i)$$

$$+ \sum_{i=0}^{p-1} \phi_{u,i}(X(t - 1))u(t - d - i)$$

$$+ \sum_{i=1}^{p} \phi_{v,i}(X(t - 1))v(t - i) + e(t), \quad (20)$$

where $y(t)$, $u(t)$ and $v(t)$ are the output, input and disturbance of the process, respectively, $X(t - 1) = [x(t - 1), x(t - 2), \ldots, x(t - n)]^T$ is the load demand series, $p$, $m$ and $n$ are the orders, $d$ is the pure time-delay of the process, and their RBF-net-style-coefficients are similar to those in model (3). In order to compare model performance, a Hammerstein(p) model (21) is also used to identify the nonlinear process:

$$y(t) = c_0 + \sum_{i=1}^{p} a_i y(t - i) + \sum_{i=0}^{p-1} b_i u(t - d - i)$$

$$+ \sum_{i=0}^{p-1} b_i^2 u^2(t - d - i) + \sum_{i=1}^{p} b_i v(t - i)$$

$$+ \sum_{i=1}^{p} b_i^2 v^2(t - i) + e(t). \quad (21)$$

The measured data set of an actual NOx decomposition process is showed in figure 1. The proposed SNPOM and the classic LMM are used to estimate model (20), and LSM is used to estimate model (21) since it is linear with respect to the parameters to be estimated. The estimated results for the NOx decomposition process are drawn in figures 2–5 and table 1.

Figure 2 shows the RBF centres, scaling parameters and the predictive error variance, respectively, of the RBF-AR(6,4,3) model (20) in the parameter search process using the SNPOM and the LMM. From figure 2, we can see that both parameter optimization methods are convergent, but the result using SNPOM is much better in terms of both convergence rate and precision than the results using LMM where the LMM is used to optimize all parameters simultaneously regardless of parameter type.
SNPOM for RBF-AR model

Using the same order in both cases, the performance of the RBF-AR model is far better than that of the Hammerstein model as is shown in table 1 and figures 3 and 4, where the RBF-AR model is estimated using the SNPOM. Figure 5 shows the eigenvalues of the RBF-AR(6,4,3) model (20) varying with the load demand of the power plants, which show that the dynamics of the NOx decomposition process have a nonlinear dependence on load. The local linearization of the Hammerstein model is still nonlinear, however, in the input at any operating point, but that of the RBF-AR model is a linear ARX model at some fixed load level or operating point. Therefore some linear
controller design methods may be applied to control the nonlinear process based on the RBF-AR model.

4.2. Modelling on EEG time series

Figure 6 shows a typical electroencephalogram (EEG) data taken during an epileptic seizure, which displays a nonlinear time series pattern with temporal spikes and waves. Using the RBF-AR \((p, m, d)\) model (22) and the RBF \((m, d)\) network (23) respectively to model the EEG series, we obtain:

\[
y(t) = \phi_0(X(t-1)) + \sum_{l=1}^{p} \phi_l(X(t-1))y(t-l) + e(t)
\]

\[
\phi_l(X(t-1)) = c_{i,0} + \sum_{k=1}^{m} c_{i,k} \times \exp\left(-\lambda_k \|X(t-1) - Z_k\|^2_2\right)
\]

\[
X(t-1) = [y(t-1), \ldots, y(t-d)]^T
\]

\[
y(t) = \theta_0 + \sum_{k=1}^{m} \theta_k \exp\left(-\lambda_k \|X(t-1) - Z_k\|^2_2\right) + e(t).
\]

The results of using the proposed SNPOM, the evolutionary programming algorithm (EPA) (Shi et al. 1999) and the LMM to estimate the models are showed in tables 2 and 3 and figures 7–12. Table 2 and figures 7 and 8 show that for estimating the same RBF-AR model the convergence rate using the SNPOM is much faster than that using the EPA, and the performance of the model estimated using the SNPOM is also better than that using the EPA compared with the results reported in Shi et al. (1999). Figure 9 shows that the convergence rate of the SNPOM is also much faster than that of the classic nonlinear optimization method LMM, especially in the case that the number of linear parameters is greater than the number of nonlinear parameters in the model (figure 9 and table 3, although both models have 51 unknown parameters, the RBF-AR \((8,3,4)\) has 36 linear parameters and 15 nonlinear parameters, whereas the RBF \((5,8)\) has six linear parameters and 45 nonlinear parameters).

From table 3 and figures 10 and 11 we can see that in the case where we have the same number of unknown parameters, the performance of the RBF-AR model is better than that of the RBF network estimated by using

<table>
<thead>
<tr>
<th>Model</th>
<th>Predictive error variance</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hammerstein(6)</td>
<td>0.0376</td>
<td>9775</td>
</tr>
<tr>
<td>RBF-AR(6,4,3)</td>
<td>0.0303</td>
<td>10256</td>
</tr>
</tbody>
</table>

Figure 7. Nonlinear parameters and predictive error variance during the optimization of the RBF-AR \((75,3,2)\) model (22) using the SNPOM for the EEG data.

Table 2. Comparison of the estimated results of the RBF-AR \((75,3,2)\) model (22) using the proposed method (SNPOM) and the evolutionary programming algorithm (EPA) for the EEG data.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Initial variance</th>
<th>Convergent variance</th>
<th>Total iterations</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPA</td>
<td>235.50</td>
<td>182.36</td>
<td>400</td>
<td>3900</td>
</tr>
<tr>
<td>SNPOM</td>
<td>325.43</td>
<td>172.73</td>
<td>31</td>
<td>3881</td>
</tr>
</tbody>
</table>

The results of using the proposed SNPOM, the evolutionary programming algorithm (EPA) (Shi et al. 1999) and the LMM to estimate the models are showed in tables 2 and 3 and figures 7–12. Table 2 and figures 7 and 8 show that for estimating the same RBF-AR model the convergence rate using the SNPOM is much faster than that using the EPA, and the performance of the model estimated using the SNPOM is also better than that using the EPA compared with the results reported in Shi et al. (1999). Figure 9 shows that the convergence rate of the SNPOM is also much faster than that of the classic nonlinear optimization method LMM, especially in the case that the number of linear parameters is greater than the number of nonlinear parameters in the model (figure 9 and table 3, although both models have 51 unknown parameters, the RBF-AR \((8,3,4)\) has 36 linear parameters and 15 nonlinear parameters, whereas the RBF \((5,8)\) has six linear parameters and 45 nonlinear parameters).

From table 3 and figures 10 and 11 we can see that in the case where we have the same number of unknown parameters, the performance of the RBF-AR model is better than that of the RBF network estimated by using...
Table 3. Comparison of the RBF-AR(8,3,4) model (22) and the RBF(5,8) network (23) for the EEG data.

<table>
<thead>
<tr>
<th>Models</th>
<th>Initial variance</th>
<th>Convergent variance</th>
<th>Total iterations</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBF(5,8)*</td>
<td>*</td>
<td>1.72 \times 10^4</td>
<td>*</td>
<td>6240</td>
</tr>
<tr>
<td>RBF(5,8)\textsuperscript{b}</td>
<td>2.91 \times 10^{3}</td>
<td>1.32 \times 10^{3}</td>
<td>285</td>
<td>4630</td>
</tr>
<tr>
<td>RBF-AR(8,3,4)</td>
<td>1.85 \times 10^{3}</td>
<td>1.23 \times 10^{3}</td>
<td>50</td>
<td>4595</td>
</tr>
</tbody>
</table>

(a) Estimates found by the EPA (Shi et al. 1999); and (b) RBF-AR(8,3,4) estimated by the SNPOM. * Corresponding value was not reported in the referred paper.

4.3. Modelling of a benchmark series generated from the Mackey–Glass equation

We used the Mackey–Glass equation below to generate the well-known chaotic benchmark time series in

\[
\frac{dx(t)}{dt} = \frac{r x(t - \tau) - x(t) x(t - \tau)}{1 + x(t)^2} - \frac{x(t)}{\tau}
\]

the SNPOM, which in turn is much better than that of the RBF network estimated by the EPA. Figure 12 shows the plot of the characteristic roots of the RBF-AR(8,3,4) model (23) estimated by the SNPOM for the EEG data for 9 \leq t \leq 631, which shows that the complex roots sometimes occurred outside the unit circle, resulting in the complexity of the EEG series dynamics.
order to compare the performance of the RBF-AR\((p, m, d)\) model (22) and the RBF\((m, d)\) network (23) as well as the different parameter optimization methods:

\[
\dot{x}(t) = \frac{ax(t - \tau)}{1 + x^2(t - \tau)} - bx(t),
\]

where the parameters are chosen to be \(a = 0.2, b = 0.1, c = 10\) and \(\tau = 20\) as used in Shi et al. (1999). The original Mackey–Glass series shown in figure 13 is extracted from a long run so that the initial transients effects could be ignored. The first 500 data points are used to train the RBF-AR model or the RBF network, and the last 500 data points are used to test the models.

The estimated results using the proposed optimization method (SNPOM), the evolutionary programming algorithm (EPA) (Shi et al. 1999) and the classic optimization method LMM are shown in table 4 and figures 14–18. During the search for optimal parameters using the SNPOM, the scaling parameters \(\lambda_k (k = 1, 2, \ldots, m)\) are determined by the heuristic approach described in (11),

![Figure 12. Eigenvalues of the RBF-AR(8,3,4) model (23) for the EEG data varying over time.](image1)

![Figure 13. Chaotic series generated from the Mackey–Glass equation.](image2)

![Figure 14. Predictive errors of the models estimated by the SNPOM for the test data-set of the Mackey–Glass series.](image3)

<table>
<thead>
<tr>
<th>Models</th>
<th>Number of nonlinear parameters (centres)</th>
<th>Total of unknown parameters</th>
<th>Training set Predictive error variance</th>
<th>AIC</th>
<th>Testing set Predictive error variance</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBF(5,5)a</td>
<td>25</td>
<td>31</td>
<td>2.92 × 10^{-4}</td>
<td>-4007.4</td>
<td>3.50 × 10^{-4}</td>
<td>-3916.8</td>
</tr>
<tr>
<td>RBF(5,5)b</td>
<td>25</td>
<td>31</td>
<td>1.16 × 10^{-7}</td>
<td>-7921.5</td>
<td>1.38 × 10^{-7}</td>
<td>-7835.7</td>
</tr>
<tr>
<td>RBF(20,5)a</td>
<td>100</td>
<td>121</td>
<td>4.32 × 10^{-5}</td>
<td>-4782.8</td>
<td>1.02 × 10^{-4}</td>
<td>-4353.3</td>
</tr>
<tr>
<td>RBF(20,5)b</td>
<td>100</td>
<td>121</td>
<td>9.46 × 10^{-8}</td>
<td>-7842.6</td>
<td>1.16 × 10^{-7}</td>
<td>-7739.2</td>
</tr>
<tr>
<td>RBF-AR(5,3,2)a</td>
<td>6</td>
<td>30</td>
<td>1.23 × 10^{-7}</td>
<td>-7895.5</td>
<td>1.31 × 10^{-7}</td>
<td>-7864.0</td>
</tr>
<tr>
<td>RBF-AR(5,3,2)b</td>
<td>6</td>
<td>30</td>
<td>1.08 × 10^{-7}</td>
<td>-7960.0</td>
<td>1.26 × 10^{-7}</td>
<td>-7880.3</td>
</tr>
</tbody>
</table>

(a) EPA estimates; (b) SNPOM estimates.
SNPOM for RBF-AR model

Figure 15. Histograms of predictive error of the models estimated by the SNPOM for the test data-set of the Mackey–Glass series.

Figure 17. Estimated parameters and predictive error variance for the RBF-AR(5,3,2) model (22) estimated using the SNPOM.

Figure 18. Time-varying estimated eigenvalues for the RBF-AR(5,3,2) model for the Mackey–Glass series estimated by the SNPOM.

where we fix $\varepsilon_k = 0.0001$. Table 4 shows that the performance of the models obtained by using the SNPOM is far superior to the performance of those using the EPA, especially for RBF networks. From table 4, we can also see a comparison of models with the same delay lag 5, which shows that although the RBF(20,5) network gives less predictive error variance because it has many centres, the RBF-AR(5,3,2) model still gives much better comprehensive performance (measured by the AIC). We can also see from figures 14 and 15 that the statistical properties of the RBF-AR(5,3,2) model are as good or better than the RBF(20,5) or the RBF(5,5) network.

From figure 16, it is clear that for the chaotic Mackey–Glass series, the convergence rate of the parameter search process using the SNPOM is much faster than that using the LMM both for the RBF-AR model and for the RBF network. Moreover, using the same SNPOM method, the convergence rate of the RBF-AR(5,3,2) model estimation is also much faster than that of the RBF(5,5) network estimation because the number of nonlinear parameters (centres) to be estimated for the RBF-AR(5,3,2) model (which has only six) is much less than that for the RBF(5,5) network (which has 25), and the SNPOM bases its search around the nonlinear parameters during the optimization of all the unknown parameters. The speedy convergence when we simultaneously estimate all the parameters of the RBF-AR(5,3,2) model (22) using the
SNPOM may be seen in figure 17. Figure 18 shows the characteristic roots of the RBF-AR(5,3,2) model estimated by the SNPOM for the Mackey–Glass series, showing the complicated dynamics of the time series.

5. Conclusions

Based on the classic LMM used for nonlinear parameter least-squares optimization and the standard LSM used for linear parameter optimization, an off-line SNPOM for estimating the RBF-AR model and the RBF network as well as other similar models whose parameters can be separated into linear and nonlinear parts has been presented. Compared with some optimization methods that search all the parameters simultaneously such as the LMM, and the EP (evolutionary programming) algorithm, the proposed SNPOM showed much faster convergence rate and still better modelling precision.

The SNPOM has both the merit of the LMM in having better robustness in nonlinear parameter optimization, and the merit of the LSM in that it can directly obtain the optimal solution of linear parameters. The reason for the fast convergence of the SNPOM is because it divides the search space of the unknown parameters into two subspaces, one being the nonlinear parameter subspace, in which the search is usually quite expensive of computation time, and the other being the linear parameter subspace, in which the optimal parameters can be rapidly obtained.

When SNPOM was used to estimate model parameters in cases where we had the same time lag delay, it was clear that the performance of the RBF-AR model was better than that of the RBF network or the Hammerstein model. In addition, the local linearization of the RBF-AR model at the operating-point is a linear AR or ARX model, which is not the case for the RBF network or the Hammerstein model.

Acknowledgments

The authors are grateful to the Ministry of Education, Culture, Sports, Science and Technology, Japan, for supporting H. P.’s visit to the Institute of Statistical Mathematics, Tokyo, where the joint work on this paper was undertaken. Finally, they thank the Editor and the anonymous referees for valuable comments.

References


