ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 381-386

On the invertibility of the RBF model in a Predictive Control strategy

Antoine Fache, Olivier Dubois, Alain Billat

Laboratoire d'Automatique et de Microélectronique
Université de Reims Champagne-Ardenne,
BP 1039, 51687 REIMS Cedex 2, France
Email : antoine.fache@univ-reims.fr

Abstract : This paper describes the importance of the RBF model quality in a model-based predictive control scheme. We show that a good neuronal approximator does not necessarily correctly model the intrinsic behaviour of the identified system. We have used a simulated example to show the harmful effects of a particular type of incorrect behaviour, the non-invertibility of the model relative to the control input. Lastly, we propose a derived RBF model that is slightly more complex, but which is systematically invertible.

1. Introduction

Due to their multidimensional approximation capability and their simple and fast learning, Radial Basis Function networks (RBF networks) have been used extensively for the identification and control of non-linear dynamic systems. They have been used in several strategies like classical non linear control scheme or model-based predictive control (MBPC) [1,2]. For this last, the neural net based model accuracy in term of prediction error is not sufficient to obtain acceptable performance. The objective of identification is here to define a good emulator [3] of the system to be controlled rather than a good approximator. Qualitative precision is more important than quantitative accuracy. The first part of this paper shows that the universal approximator property of RBF networks does not necessarily imply a correct system behaviour capture. We are specifically interested in model invertibility, whose definition and the way it is described are summarized. We have used a simulated system to study the effects of a non-invertible model on the performances of a generalised predictive control scheme. Lastly, we describe a model which overcomes this particular problem.

2. The RBF net and its universal approximator property

Several studies have shown that RBF networks are universal approximator [4]. Assuming we need to identify a SISO system with input u(k) and output y(k) and with dynamic $f(.) : K \rightarrow R$, $K \subset R^r$.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 381-386

The set of RBF model parametrized by θ is of the form :

$$NN_\theta\big(X(k)\big) = \sum_{i=1}^{N_h} w_i \varphi_i\big(X(k)\big) = \sum_{i=1}^{N_h} w_i \varphi\left(\frac{X(k)-C_i}{\sigma}\right) \tag{1}$$

where $X(k) = [y(k),...,y(k-n_y),u(k),...,u(k-n_u)]^T \in Y^{ny} \times U^{nu} \subseteq K$. The input and output orders n_u and n_y are assumed to be known a priori. The vector θ includes the number of hidden neurons N_h, the centres C_i, the weights w_i and the unique width σ. Under mild assumptions on the activation function φ(.), it has been proved that for a given arbitrary accuracy ε, there is always a set of parameters θ* such that [4] :

$$\left|f(X) - NN_{\theta^*}(X)\right| < \varepsilon \quad \forall X \in Y^{ny} \times U^{nu} \tag{2}$$

This result guaranties a finite bounded prediction error over the approximation domain, but it does not imply an agreement, even a biased one, between the dynamics of the system and the model. In fact, the relationship (2) means that $NN_{\theta*}(X)$ is necessarily between two hypersurfaces, centered on f(X) and distant from 2ε when X moves on $Y^{ny}\times U^{nu}$. We can try to reduce this by limiting ε, or by using a regression technique [5], but without completely cancelling it. The behaviour of the model in this region is therefore free and globally uncontrollable. Let us note that the non-linear statistical tests proposed in [6] do not always detect these undesired model behaviours.

3. Definition and measurement of invertibility

This section deals with influence of a non-invertible model used in a neural net-based GPC scheme. For this, we shall rewrite X(k) as X(k) = [u(k),ψ(k)], where u(k) is the actual control input and $\psi(k) \in Y^{ny}\times U^{nu-1}$ is the system history i.e. the set of variables not controllable at time k. The equation (2) becomes :

$$\hat{y}(k+1) = NN\big[u(k), \psi(k)\big] \tag{3}$$

We also define :

$$\hat{y}(u|k_0) = NN\big[u, \psi(k_0)\big] \tag{4}$$

as the model output at time k_0+1 for an arbitrary command $u\in U$ and a fixed history $\psi(k_0)$. The notion of invertibility relatively to the control input was proposed in [7]. A neural model is locally invertible if the output (4) is different for two any distinct inputs u_1 and u_2 taken in U. This definition implies that a model is invertible if, and only if, $\forall k_0 \geq 0$ and $\forall\ \psi(k_0) \in Y^{ny}\times U^{nu-1}$, the output $\hat{y}(u|k_0)$ is monotonic with respect to $u \in U$. We will evaluate this property at time k_0 by looking for the input interval $U_{inv}(k_0) = [u_m(k_0)\ ;\ u_M(k_0)] \subseteq U$ such that :

$$\forall\ u \in U_{inv}(k_0)\,,\ SGN\left(\frac{\partial\ \hat{y}(u|k_0)}{\partial\ u}\right) = C^{st} \tag{5}$$

where SGN(.) is the sign function and takes value ±1. If an invertible system is correctly emulated by the model, we must have $U_{inv}(k_0) = U\ \forall\ k_0$. The invertibility interval sought is centered on U, but in the general case there can be several zones of invertibility located anywhere on U. The derivative vanishes at points $u_m(k_0)$ and $u_M(k_0)$, so that the direction of change in the output model toggles. We will see later that $U_{inv}(k_0)$ can then represent a zone of inverse behaviour, where the output model moves in the opposite direction from the real output.

4. Simulation study

We have simulated a SISO system defined by [7]:

$$y(k+1) = f\big(y(k), u(k)\big) = \frac{y(k)}{1+y(k)^2} + u(k)^3 \qquad (6)$$

This non-linear system was identified with a Gaussian RBF network given by (1) with activation functions :

$$\varphi_i\big(X(k)\big) = \exp\left\{-\frac{\|X(k)-C_i\|^2}{\sigma^2}\right\} \forall i = 1 \text{ to } N_h \qquad (7)$$

The network is learned by a hybrid technique [8]. We have selected an erroneous model structure by taking n_y=1 et n_u=2 to ensure a clear demonstration. The resulting prediction errors are relatively large but remain bounded on $Y \times U^2$. The over-parameterization produces a very poor emulation of the system. We have chosen a generalised predictive control (GPC) [9], whose minimizing cost function is given by :

$$J(k,u) = \sum_{j=N_1}^{N_p}\big(y_{ref}(k+j) - \hat{y}(k+j)\big)^2 + \lambda \sum_{j=0}^{N_u-1}\big(u(k+j) - u(k+j-1)\big)^2 \qquad (8)$$

where $y_{ref}(k+j)$ is the reference trajectory to reach. The future estimated outputs $\hat{y}(k+j)$ are usually obtained by N_p iterations of the model (1).

A test was done with the GPC parameters set at $\lambda = 0.0$, $N_1 = N_p = N_u = 1$. This allowed us to ignore the model accuracy when it worked in simulation, since only the estimated output at time k+1 was required. Part of the results are shown in figure 1. The upper part shows the reference trajectory, the system response and the predicted output. The lower part shows the control input u(k) and its invertibility limits $u_m(k)$ and $u_M(k)$. The overall result was satisfactory, despite the relative inaccuracy of the model used. Since u(k) was only calculated from the output model, it almost perfectly follows the trajectory, even though the true output was biased. A more interesting phenomenon occurred between times k = 80 and k = 99, when the GPC seemed to become unstable. Figure 2 shows the situation which the numerical algorithm encounters at time $k_0 = 84$ to select the optimal control law. The upper part shows $y_{ref}(k_0+1)$ and the model output according to u. The lower part shows the corresponding change in $J(k_0,u)$. The dotted curves show the same variables calculated from the simulated system. The selected command corresponds to a local minimum (induced by the non-monotony of the model) that is in a zone of inverse behaviour.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 381-386

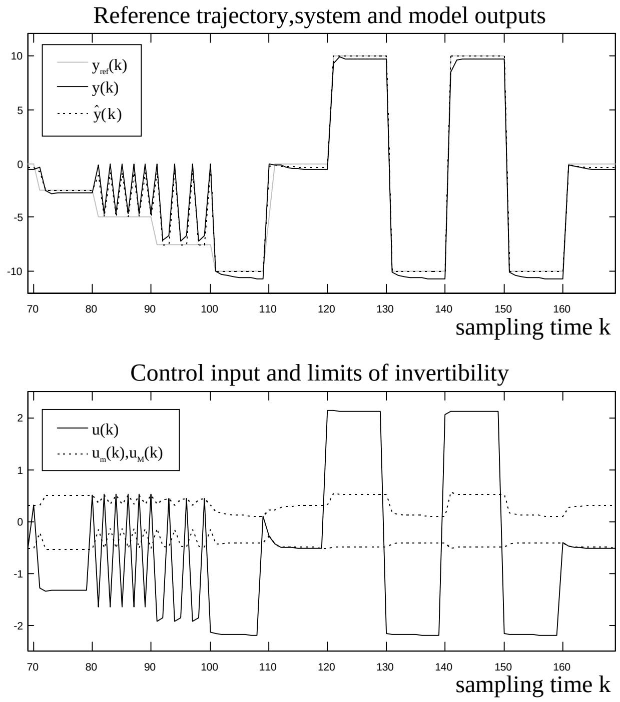

Figure 1. GPC with a non-invertible RBF net model

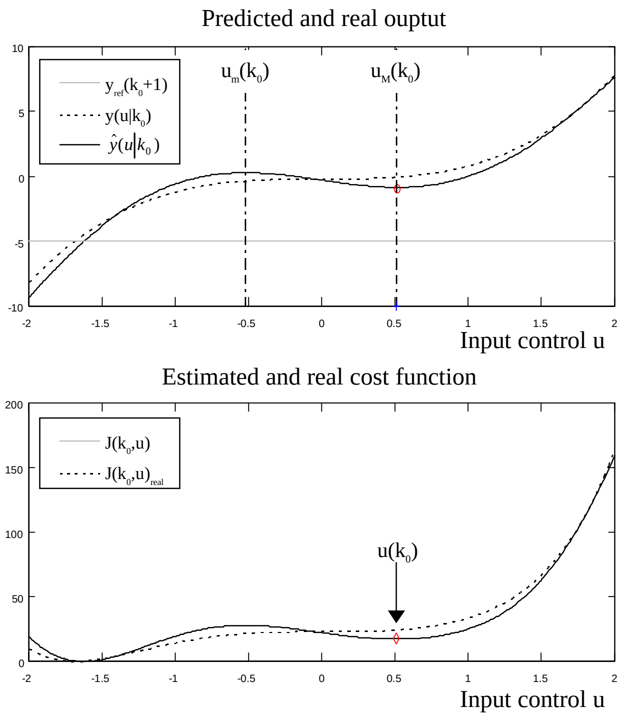

Figure 2 : Command calculus at time k=84

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 381-386

Thus, $u(k_0)$ had the effect of bringing the model output $\hat{y}(k_0+1)$ closer to the reference whereas it moved away the real output from there. If this continues, the command is likely to diverge definitively since there is no real tracking error feedback in the GPC loop. But here, the calculated control input at time k_0+1 (not shown) represents the global minimum and is outside $U_{inv}(k_0+1)$ (cf figure 1). The model then evolved in the same direction as the system, producing a correct control. Thus, the observed oscillations resulted from a succession of these two situations. This is a simple case, since there was only one zone of inverse behaviour in the vicinity of u = 0.0 and u(k) was only calculated, without penalty, from the following prediction. The general case ($N_p>1$, $\lambda>0.0$) is more complex to study. The second part of (8) can be made dominant by increasing λ, which reduces the problem. However, this solution is artificial and can be dangerous because this parameter strongly influences the GPC stability[9].

5. A systematically invertible indirect RBF model

It is mathematically difficult to introduce a monotony constraint into the RBF training to obtain a systematically invertible model and modifying the optimal control calculation to avoid the local minima can require a lot of computation. The specialised training described in [3] could be an interesting solution, but requires working with a state space model. It also does not guarantee that there will be no inverse behaviour zones. The solution we propose is to use a derived RBF model inspired by [10] and defined by:

$$\hat{y}(k+1)=\sum_{i=1}^{n_y} a_i(O_k)y(k-i)+\sum_{j=1}^{n_u} b_j(O_k)u(k-j) \tag{9}$$

where $O_k = [y(k-1)...y(k-n),u(k-d)...u(k-m)]^T$. The coefficients $a_i(.)$ et $b_j(.)$ are RBF outputs computed according to :

$$a_i(O_k)=\sum_{l=0}^{Nh} w_l^{a_i}\varphi_l\left(O_k\right) \quad , \quad b_j(O_k)=\sum_{l=0}^{Nh} w_l^{b_j}\varphi_l\left(O_k\right) \tag{10}$$

After some manipulation of the regressor O_k, it is possible to carry out the network learning by a standard least mean squares technique. In the same way, the adaptation can be simply performed on-line by recursive least squares. Choosing $d>1$ immediately shows that the output $\hat{y}(u|k_0)$ calculated for (9) is the output of a purely linear ARX model, which is therefore always invertible relative to the most recent control input. The two hidden layers of the structure mean that fewer hidden neurons are generally needed to reach the precision of a direct RBF model.

6. Conclusion

We have highlighted the importance of the qualitative aspect of an RBF model, in particular its invertibility, in a MBPC type of control strategy. In a MPBC scheme, the control law is calculated assuming the "certainty

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 381-386

equivalence" principle between the system and the model. That means that the model change in response to a control sequence is similar to the real process one. When the RBF model checks this assumption and the controller parameters are selected appropriately (i.e. produce a stable closed loop), the universal approximator property ensures a bounded tracking error. But this equivalence is no longer respected when the model is not invertible. Local minima then appear, which can be appropriately treated during computation of the optimal control, but with a waste of time that is undesirable in real-time applications. Another more serious consequence is the possible appearance of zones of opposite behaviour. These areas can produce aberrant control inputs which can destabilise the closed-loop system. We have developed a systematically invertible model to overcome this type of problem. While it seems more complex, it has the same speed of training as a direct RBF model, with a computational load that is only slightly greater.

References

[1] D. Neumerkel, J. Franz, L. Kruger, A. Hidiroglu : Real-Time Application of Neural Model Predictive Control for an Induction Servo Drive. in The 3rd IEEE Conf. on Control Appl., 24-26 August 1994, The University of Strathclyde, Glasgow, Scotland U.K. (1994)

[2] D. Soloway, P.J. Haley : Neural Generalized Predictive Control : A Newton-Raphson Implementation. NASA Technical Memorandum 110244, Langley Research Center, Hampton, Virginia (1997)

[3] S.V. Chakravarthy, J. Ghosh : Function emulation using Radial Basis Function Networks. Neural Networks 10:3, 459-478 (1997)

[4] J. Park, I.W. Sandberg : Universal Approximation Using Radial-Basis-Function Networks. Neural Computation 3, 246-257 (1991)

[5] S. Geman, E. Bienenstock, R. Doursat : Neural Networks and the bias/variance dilemma. Neural Computation 4:1, 1-58 (1992)

[6] S. Chen, S.A. Billings, C.F.N. Cowan, P.M. Grant : Practical identification of NARMAX models using radial basis functions. International Journal of Control 52:6 1327-1350 (1990)

[7] K.J Hunt, D. Sbarbaro : Neural Networks for Nonlinear Internal Model Control. IEEE Proceedings-D 138:5, 431-438 (1991).

[8] S. Chen, S.A. Billings : Neural networks for nonlinear dynamic system modeling and identification. International Journal of Control 56:2, 319-346 (1992)

[9] D. Clarke, W.C. Mothadi, P.S TUFFS : Generalized Predictive Control - part I. The basic algorithm. Automatica 23:2, 137-148 (1987)

[10] Z.Q. Wu, C.J. Harris : Modelling and Adaptive Filtering of Nonlinear Systems Using Neural Network. Technical Report ISIS-TR2, Dept of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK (1995)