Local Model Networks and Local Learning

Roderick Murray-Smith
Daimler-Benz AG, Systems Technology Research, Alt-Moabit 91b, 10559 Berlin, Germany.
E-mail: murray@DBresearch–berlin.de

Keywords: Local Model Networks, Local Learning, Basis Function Nets

Abstract: The Local Model Networks (networks composed of locally accurate models, where the output is interpolated by smooth locally active basis functions) described in this paper provide a solid basis for practical modelling tasks. The architecture benefits from being able to incorporate Fuzzy, Neural Network and conventional System Identification methodology and experience. The advantages of the architecture are described, and the tradeoff between Local and Global Learning is investigated. The Local Learning method is computationally less expensive and was found to lead to smoother and more interpretable solutions than global learning. The results are illustrated with a robot actuator modelling problem.

1. Introduction

Modelling nonlinear dynamic systems from observed data and *a priori* engineering knowledge is a major area of science and engineering. In recent years a great deal of work has appeared in new areas like Fuzzy Modelling and Neural Networks to complement the previous work in statistics and system identification. The recent work, however, often lacks a solid engineering methodology. Neural networks can learn to reproduce the behaviour evident in their training set, but they are usually unable to benefit directly from *a priori* knowledge, or to provide good estimates of their accuracy and robustness. Fuzzy systems have also been heavily used for modelling non-linear systems with their structure provided by experts for the system, but often lacked the ability to refine their structure (membership functions, rules) in a data driven manner, meaning that much of the development time is then spent 'tweaking' parameters.

The Local Model Net is proposed as a useful hybrid method incorporating the advantages of the various paradigms. This paper gives an outline of the architecture, with an overview of the literature and discusses a variation in the learning algorithm for the networks local model parameters.

2. From Basis Function Nets to Local Model Nets

2.1 Basis Function Nets

The basic BF Net is shown in Figure 1. The output is a linear combination of many locally active non-linear *basis functions*. Each unit's centre is a point in the input space, and the *receptive field* of the unit (the volume of the input space to which it reacts) is defined by its distance metric. The *basis* or *activation function* (similar to the membership function of a fuzzy set) of the unit is composed of two elements: The *distance metric d(x)*, which can scale and shape the spread of the basis function relative to its centre *C*, and the *basis function* itself

$\phi(d(x))$, which takes the distance metric as its input. They are usually chosen so that the activation monotonically decreases towards zero as the input point moves away from the unit's centre, e.g. B-Splines or Gaussian bells are common choices. Radial Basis Function (RBF) nets are the most straightforward, and most commonly used types of Basis Function networks.

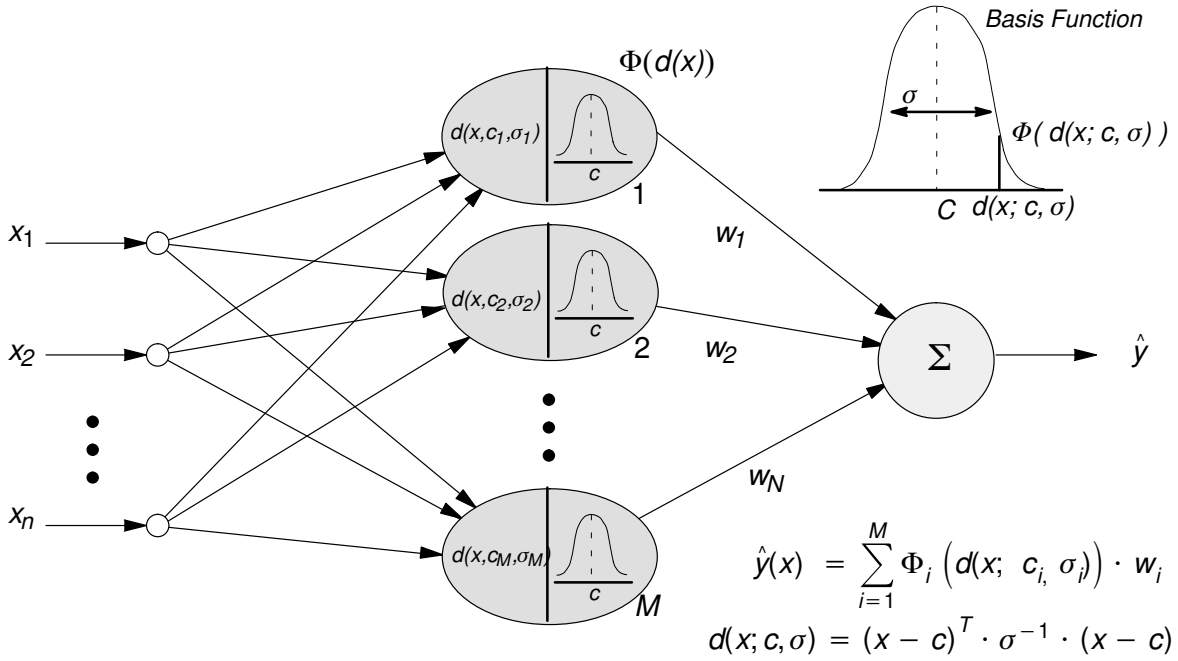

Figure 1: A Basis Function Network

If the units have localised receptive fields, and a limited degree of overlap with their neighbours, the unit's weights can be viewed as locally accurate piecewise constant models whose 'validity' for a given input is indicated by their unit's own activation functions for a given input. The simple single layered structure of Basis Function nets leads to a higher level of transparency than with Multi-Layer Perceptrons (MLP's).

The sum of the basis functions at any point in the input space should be unity. This is usually impossible to achieve in practice (given a high degree of flexibility in where the centres can be), but one option is to artificially normalise the output from the basis functions. This ensures that the basis functions perform a partition of unity, and ensures that the function is able to represent the mapping without containing oscillations due to the basis functions not adding to unity.

$$\phi_{normed}(d(x, c, \sigma)) = \frac{\phi(d(x; c, \sigma))}{\sum_{b=1}^{M} \phi\left(d(x; c_b, \sigma_b)\right)}$$

Basis Function Networks and their equivalents have been used for function approximation and modelling in various forms for many years: *Potential Functions* [Aizerman 64], *Kernels* and *Spline Models* [Wahba 92] are all similar structures. [Poggio 90][Girosi 93] describe the networks within the mathematical framework of regularisation theory for function approximation. Recently BF neural networks have received a

growing amount of attention from the neural network community [Broomhead 88][Moody 89][Jones 89][Hlavackova 92][Pantaleon 93]. Use of RBF nets for modelling and control purposes is described in [Chen 91][Sanner 92][Sbarbaro 92][Röscheisen 92]. [Leonard 92] shows how local confidence limits can be calculated with basis function nets.

Fuzzy Systems can also be viewed as Basis Function networks, as the membership functions for the rules are active in a limited area of the input space, and are analogous to the basis functions, while the weights can be viewed as the consequence part of the fuzzy rules [Jang 93][Hunt 94].

Basis Function Networks for Modelling

The modelling problem as seen in this paper is to try to robustly approximate a given non-linear dynamic system from observation data. The assumption made is that the system can be modelled as $\hat{y} = f(x)$, some function *f(x)* of the inputs x, subject to a measurement error ε, so that the true outputs y are: $y = f(x) + \varepsilon$.

For dynamic systems x will contain the previous inputs and outputs: $x_k = y_{k-1}, ..., y_{k-n}, u_{k-1}, ..., u_{k-n}$

The learning task is therefore to try to adapt the structure and parameters of the Basis Function net to minimise a cost function related to the deviation of the model from the target system. The optimisation of the weights (in this paper weights are the parameters weighting the effect of the basis functions on the output and do not include the basis function parameters, the centres and widths) is a linear process and relatively straightforward, while the optimisation of the basis functions is a difficult nonlinear problem, where *ad hoc* methods of reducing the complexity play an important role.

Partitioning the input space – The concept of locality

An intrinsic feature of the Basis Function networks is the concept of locality. In linear systems, the data, optimisation and validation are all considered to be globally relevant, i.e. any results obtained are valid over the entire input space. For non-linear systems, however, (especially when multivariable) it makes a great deal of sense to partition the input space into multiple subspaces. This can involve a reduction of the problem's dimensionality by decomposing the problem, discarding irrelevant interactions, or of simply partitioning the input spaces into subspaces which are easier to handle – the traditional 'divide and conquer' strategy inherent to local modelling techniques, such as basis function nets. The problem with standard basis function networks is that the crudeness of the local approximation forces the system to use large numbers of basis function units to approximate a given system, leading to computational, transparency and robustness problems (the training data to train all of the units has to be available!). This is especially true for higher dimensional problems. It is therefore important to be able to profit from the local nature of the basis functions while not requiring too many units. This implies that the basis functions should be associated with more powerful representations than piecewise constant models, so that a smaller number of them could cover larger areas of the input space with sufficient accuracy.

2.2 Local Model Basis Function networks

The standard basis function network can therefore be generalised to allow not just a constant weight associated with each basis function, but also a function of the inputs, so that the network can be described in the form: $\hat{y} = \sum_{i=1}^{M} \phi_i\left(d(x; c_i, \sigma_i)\right) \cdot f_i(x; w_i)$. This can then be viewed as a decomposition of the complex, nonlinear system into a set of locally accurate submodels which are then smoothly integrated by their associated basis functions. Creating black-box models from local models provides an interface between the experimentally biased research world of neural networks with the more rigorous results from statistics and system identification, creating a powerful structure which can benefit from the years of analysis and experience with linear systems.

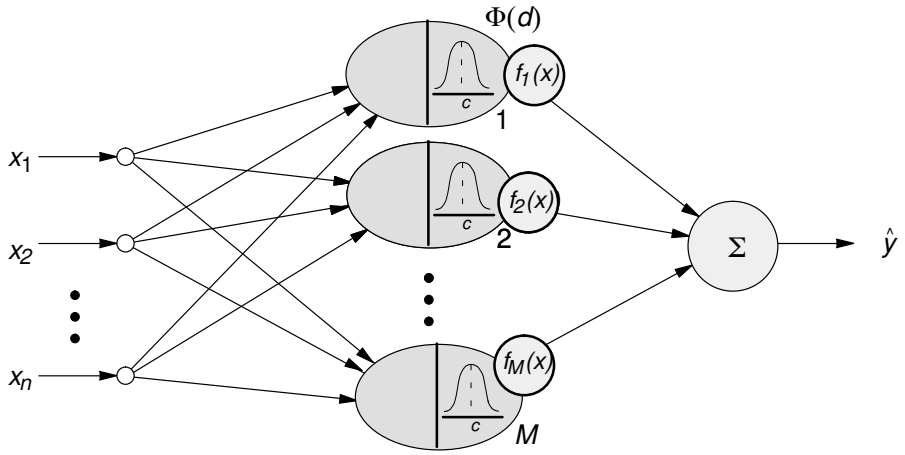

Figure 2: The Local Model Basis Function network

Literature of local model methods in learning and modelling

The idea of using local models within a basis function network has been suggested in several papers, but has only recently started to gain momentum. It was suggested in [Jones 89] and followed up by [Stokbro 90] and [Barnes 91]. The Adaptive Expert networks in [Jacobs 91] are essentially local model systems, where the local models are called 'expert networks' and the partitions are made by 'gating networks'. Bottou and Vapnik discuss the advantages of local representations in [Bottou 92], where they suggest that a proper compromise between local and global methods will usually prove most effective as varying levels of complexity are required throughout the input space, although they claim that the 'local capacity' should match the data density, which is not necessarily true – the more general goal is that the learning system should match the *'local complexity'*.

The idea of using locally accurate models is also described in the statistical literature in [Cleveland 88], where local linear or quadratic models are weighted by smoothing functions. [Atkeson 90] reviews the literature of local learning, with examples from statistics dating as far back as 1912! Two of the main workers in recent years applying local model systems for diagnosis, modelling and control have been Johansen and Foss in

Trondheim [Johansen 92A, 92B, 92C, 93A, 93B, 94] & [Foss 93, 94]. [Skeppstedt 92] also describes the use of local dynamic models for modelling & control purposes, but with step-like transfers from one model regime to the next. The methods of Takagi and Sugeno [Takagi 85] for Fuzzy Systems are also effectively piecewise linear models, with the fuzzy interpolation between models provided by the membership functions. Similar applications are reported in [Sugeno 88] and [Foss 93].

3. Structure Initialisation & Optimisation

Structure optimisation is the most important aspect of the learning process for local model networks, but is not emphasised in this paper due to lack of space. The most relevant aspects will be briefly described.

The advantage of Basis Function networks is that the nonlinearity is a *localised* one. This provides advantages for learning efficiency, generalisation and transparency. It is, however, very difficult to automatically find the 'correct' level of locality for a given subspace of an arbitrary problem, because he concept of 'locality' is obviously relative, depending on the complexity of the system, the availability of training data, the importance of the given area of the input space, and, importantly for models, *a priori* knowledge of internal structures within the given system.

If the user already has *a priori* knowledge about the system being modelled, this could be used to define or initialise the basis function partition. (The use of such physically based knowledge makes the model more easily interpretable and also makes on-line adaptation of the system's parameters much more feasible). As mentioned earlier, some classes of fuzzy systems can also be viewed as Basis Function models. The relationship to fuzzy systems is interesting, as an initial partition of the input space can then be supplied in the form of linguistic rules and membership functions. Once this initial partition of the input space has been completed, the consequence of each node can be a local model network which learns the remaining structural details by a data-driven structure adaptation algorithm.

4. Weight Optimisation: Local Learning versus Global Learning

4.1 Estimating the pseudoinverse with Singular Value Decomposition

The optimisation of the weights in a Basis Function network is a straightforward application of Linear Regression techniques, and as the optimisation problem is linear in the parameters, the global minimum should always be found. The generalised form of linear parameter identification used for systems such as Basis Function nets is *General Linear Least Squares fitting*. Here, a linear combination of M basis functions (which can be as nonlinear as you like!) of the actual inputs x is optimised with respect to a given cost function. The general BF net system is shown below:

$\hat{y}(x) = \sum_{k=1}^{M} w_k \ \Phi_k\left(d(x;\ c_k,\ \sigma_k)\right)$, or in matrix form, for all inputs in the training set: $\hat{Y} = \Phi W$. The training problem for a given set of basis functions can be viewed as finding the solution of W to the linear equations, where Φ is the *design matrix*, and where we have to find the best set of weights W. One way of finding

the weights is to calculate the Moore-Penrose pseudoinverse of Φ, $\Phi^+ = \left[\Phi^T\Phi\right]^{-1}\Phi^T$. The weights W can then be calculated as $\hat{w} = \Phi^+ \cdot y$. For local model networks with linear local models, the basis functions are extended (producing *M(n+1)* basis functions) by multiplying the basis function by the inputs:

$$\Phi_{k:k+n} = \Phi_k\left(d(x;\ c_k,\ \sigma_k)\right) \cdot \left[w_0 + \sum_{k=0}^{n} x_k w_{1_k}\right]$$

Using Singular Value Decomposition to find the weights

The task of matrix inversion is important for the optimisation process. The algorithm used in this work, *Singular Value Decomposition* (SVD) of a matrix of observed input data, is an important step to the robust estimation of the parameters for a generalised linear least squares system. It is robust because it can cope with singular or poorly conditioned matrices. The algorithm also delivers important information about the significance of the result and the importance of the various basis functions used in the model. The SVD algorithm decomposes any *NxM* matrix Φ to matrices U (*N*x*M* column orthogonal), S (*M*x*M* diagonal) and V (*M*x*M* orthogonal), such that $\Phi = U \cdot S \cdot V^T$. More detailed descriptions of the algorithm are given in [Golub 89] or [Press 93]. The decomposition has the form:

$$\Phi = \begin{pmatrix} \Phi_1(x_1) & \Phi_2(x_1) & \dots & \Phi_M(x_1) \\ \Phi_1(x_2) & \Phi_2(x_2) & \dots & \Phi_M(x_2) \\ & \vdots & & \\ \Phi_1(x_N) & \Phi_2(x_N) & \dots & \Phi_M(x_N) \end{pmatrix} = \begin{pmatrix} \\ \Phi \\ \\ \end{pmatrix} = \begin{pmatrix} \\ U \\ \\ \end{pmatrix} \cdot \begin{pmatrix} s_1 & & & \\ & s_2 & & \\ & & \dots & \\ & & & s_n \end{pmatrix} \cdot \begin{pmatrix} \\ V^T \\ \\ \end{pmatrix}$$

allowing the pseudoinverse to be easily calculated: $\Phi^+ = V \cdot \left[diag(1/s_i)\right] \cdot U^T$, so the weights are: $\hat{w} = \Phi^+ \cdot y = V \cdot \left[diag(1/s_i)\right] \cdot U^T \cdot y$

4.2 Local learning

The optimisation process described above assumed that all of the weights would be optimised simultaneously with a single pseudoinverse calculation. This is not always computationally feasible if a large number of training patterns or local models are needed for a particular problem. A further problem is that the global nature of the observation can lead to the trained network being less transparent, as the parameters of the local models cannot be interpreted without considering the effect of neighbouring nodes. Even with a robust method such as SVD, the 'optimal' network parameters may consist of delicately balancing large positive and negative weights which minimise the output error on the training set, but which are not robust when confronted with new examples.

The alternative to global learning is to optimise each local model independently. The potential advantages are:

- there is less computational effort involved, so learning is faster.
- the final network will perform worse on the training data, but will be a smoother approximation to the physical system than the globally trained model.
- different local models can use different optimisation algorithms (especially important for varying *a priori* algorithms which are not linearly optimisable).
- there is less interference and cooperation between models during the learning process, making locally trained models more independent, which could possibly be useful for algorithms for on-line learning.

4.2.1 Weighted least squares optimisation for local learning

In many learning situations there are areas of the input space which are less important than others, either because the system spends most of its time in one particular operating regime, or because a particular aspect of the system is more interesting than others. It is also common to have varying levels of measurement accuracy in different areas of the input space. It is therefore important to be able to weight points in the training set to have more or less significance, depending on their location. The general term for this type of optimisation for linear optimisation techniques is *weighted least squares*. The cost function is scaled by the weighting function $Q(x)$, with cost J: $J = \frac{1}{N_l}\left[\left(Y - \hat{Y}\right)^T Q\left(Y - \hat{Y}\right)\right]$, so that the form of the optimisation equation is: $\hat{W} = \left[\Phi^T Q \Phi\right]^{-1} \Phi^T Q Y$.

Weighted Least Squares optimisation is useful for local learning as a given local model's basis function can be used to define that model's relevance for any given input. The weighting function is therefore directly based on the model's basis function $Q(x) = \phi(d(x))$. The local learning method is therefore to compute M pseudoinverses, one for each local model, using only the training data within the model's receptive field. The analogous method for straightforward RBF networks would be to set the weight of a unit to the average of the data points in its receptive field, as in [Pantaleon 93]. It should be noted that the local optimisation method requires the partition of unity property which can be achieved by normalising the basis functions.

Computational effort

The effort needed to find the pseudoinverse using SVD for a ($p \times q$) matrix is roughly $O(p^2 \bullet q + p \bullet q^2 + min(p,q)^3)$ [Noble 88]. In terms of the information matrix for a basis function network, p relates to the number of training points and q is the number of basis terms (in a local model structure M, the number of local models, $\times n$, the number of parameters in the local models). The cubic term shows the importance of the smallest dimension of the matrix on the complexity of the calculation. As the set of linear equations should be overdetermined, the smaller number is going to be q, representing the number of basis elements, and this implies that the number and complexity of local models is the crucial factor with regard to computational effort.

The computational effort required for the two possibilities is shown by O_{global} where $p = N$ and $q = M \bullet n$ and O_{local} where $p=N_{local}$, $q=n$ and which is repeated M times. It is difficult to compare the methods exactly, as the reduction in the number of training points in a particular area is dependent on the problem in question and will vary for each local model. $O_{global} = O(N^2 (M \bullet n) + N(M \bullet n)^2 + min(N,(M \bullet n))^3)$ and $O_{local} = O(M \bullet (N^2 \bullet n + N \bullet n^2 + min(N,n)^3))$. Even ignoring the speed-up gained by the reduced number of points, the local variant will be faster for all M greater than 1.

Modelling Robustness

The most important feature of a learning algorithm is that it robustly delivers a solution which has as high a level of accuracy as possible, and which is likely to generalise well. The global methods for linear optimisation will give the optimum fit to the data in the training set, but will not necessarily generalise robustly. Large changes in one area of the input space will affect the model's performance in other areas.

The smoothness of the resulting model is also important for many applications. In many cases, a smooth model which has a poorer least squares cost is better than a 'more' exact but 'wrinkled' model. For example, in model based predictive control, the optimal control setting is often found using gradient methods, which would then be subject to many local minima and would lead to unreliable control.

Transparency

An oft-cited advantage of local model networks is that the trained local models are easier to interpret for the engineer than other network types because they are already in a form which is more similar to the conventional representations used for dynamic systems. If global training is used, the parameters can often have no local meaning, as they depend on interaction with their neighbours to produce the correct model behaviour. Locally trained local models can be interpreted independently of all other models, as they do not depend on their neighbours. For any point in the input space the system can show the nearest local models with parameters, which should help the designer to better understand the physical system and the model.

5. Experimental results

5.1 Static example

To give an impression of the effect of local learning on the final solution of a learning problem, a simple 1-dimensional example is shown below in figure 3. The left hand side shows the target function and the trained network's response. The right hand side shows the normalised basis functions and the associated local models. The global learning solution, part (a), produces the lowest mean squared error, but the effects of the rapid change in the system output are not contained locally, but affect the model over the whole input space. This means that more of the globally trained models are less representative of the local behaviour of the model than the locally trained ones (b), making the network's parameters less interpretable.

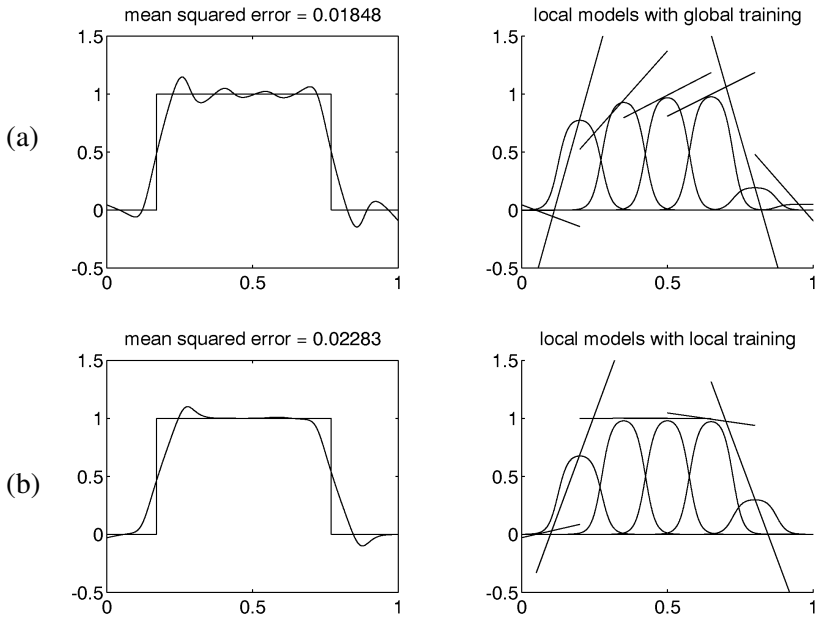

Figure 3: (a) Globally trained network (b) Locally trained network

5.2 Robot actuator modelling

The robot application described is based on data sampled from a physical system, supplied by Tom Kavli, SINTEF, Oslo. The application and the data sets have been described for modelling work in [Kavli 92] and in [Johansen 93B]. A brief overview is given here:

Many industrial robot applications now demand high dynamic accuracy. Model based control schemes have the potential to improve performance, but the use of hydraulic manipulators has suffered due to the lack of good nonlinear models for the hydraulic components. Model based control schemes have been more successful on electric direct-drive arms which generally have low friction and linear actuators. The goal of the learning task is to form a model of the servo valve/actuator system of a hydraulic robot. The robot is an ABB Trallfa TR4000 Robot, specially designed for spray painting, where tracking accuracy over a desired trajectory is extremely important. The control signal u is described as a function of the joint position q, velocity $\dot{q}$ and acceleration $\ddot{q}$: $\hat{u} = u(q, \dot{q}, \ddot{q})$.

The nonlinearities are due to: (1) the changing moment arm of the cylinder over the operating range, (2) the nonlinear damping coefficient due to the quadratic flow/pressure relation for turbulent flow and (3) the changing pressure gain characteristics for the servo valve at different flow rates.

The data was sampled by logging the data at 100Hz, while the manipulator moved along a randomly generated path. The velocity and acceleration signals were calculated by low pass filtering the data and differentiating the joint positions. The linear effects in the system were subtracted from the data to emphasise the nonlinearity of the system. The training data consisted of 8000 training points and the test set had 1000 points.

5.2.1 Experimental results

The ASMOD (Adaptive Spline MODel) results * are taken from [Kavli 92]. LSA (Local Search Algorithm) results + are from [Johansen 93B]. The RBF results were obtained by using a clustering algorithm for the structure optimisation, and an active learning algorithm to reduce the number of points in the training set [Murray-Smith 94]. The training set was reduced to a maximum of 3000 points. The measure used is the normalised root-mean-square error, for comparison with the earlier work (RMS error divided by standard deviation of outputs). Error results for the RBF models are given for the full training set (8000 points) and the test set (1000 points).

Network	*Error (NRMS)*
ASMOD(Quadrat.)*	*15% test*
ASMOD(Linear)*	*17% test*
LSA+	*17% test*
RBF globally trained model	*14% train 17% test*
RBF locally trained model	*16% train 19% test*

Visualisation of the system.

The following surfaces are representations of the input space as seen through different slices through the space. The areas in the corners of the plots are areas where the system had no training data, and are therefore unreliable. The local model net surfaces below, show the apparent smoothness of the nonlinearity:

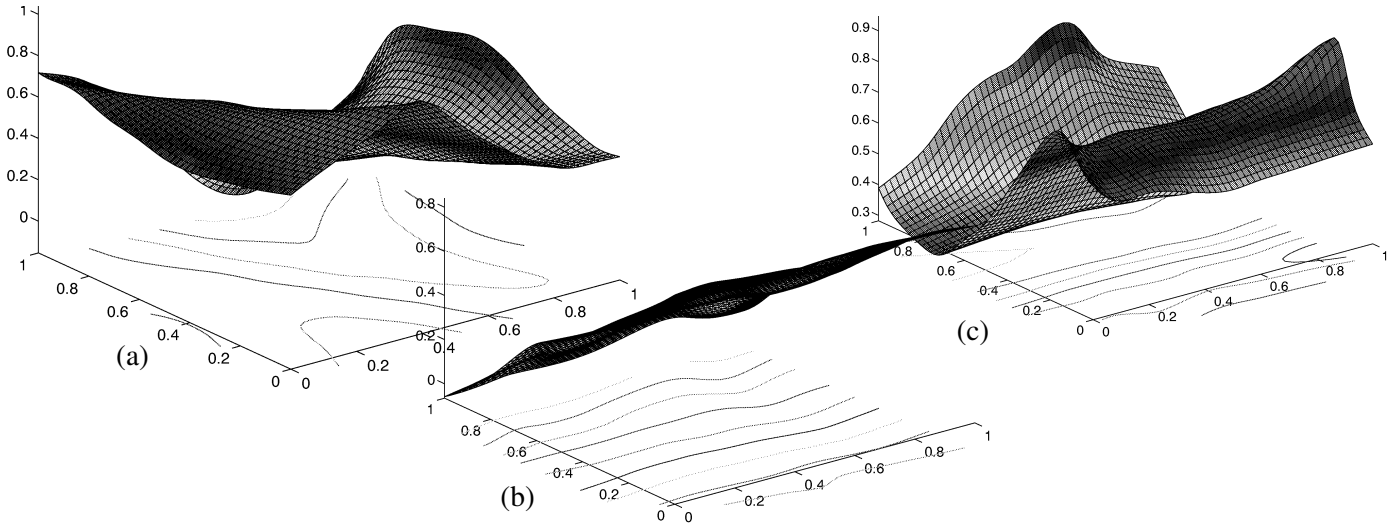

Figure 4: (a) Speed and acceleration (b) Position and acceleration (c) Position and speed

The results obtained by the Local Model Net are comparable with the results quoted in the original work [Kavli 92] and also with those in [Johansen 93B]. The interesting point is that the local learning implementation hardly seems to suffer from the simpler optimisation, when the test results are examined.

6. Conclusions

The local model network provides a useful framework for integrating various areas of modelling research:

- The network structure is more transparent than other architectures (e.g. Multi-layer Perceptrons).
- It allows easy integration of existing models or model structures.
- It can be pre-structured using fuzzy rules.
- The weight optimisation stage can use standard linear least squares techniques.
- The local nature of the basis functions allows the estimation of local confidence limits on the output.

This paper investigated two ways of optimising the local model parameters, given a particular basis structure. Global and Local singular value decomposition algorithms were used. The analysis of the computational complexity shows that local learning is faster than global learning. The structure also allows more flexibility in the use of optimisation algorithms, which will be especially useful with hybrid model structures which require nonlinear local optimisation, or on-line learning. A further point is that the locally trained networks are more interpretable than the globally trained ones, and although producing worst least squares statistics, deliver smoother models without having to resort to expensive cost functionals as in [Poggio 90][Girosi 93].

7. References

[AIZERMANN 64] M. AIZERMAN, E. BRAVERMAN, L. RONZONOER
Theoretical foundations of the potential function method in pattern recognition learning
Automatika i Telemekhanika, (in Russian) 25 pp 147-169, 1964.

[ATKESON 90] CHRISTOPHER G. ATKESON, *Memory-Based Approaches to Approximating Continuous Functions*
Workshop on Nonlinear Modelling & Forecasting, Sept. 1990, Santa Fe Institute, Publ. Addison-Wesley.

[BARNES 91] CHRIS BARNES, STAN BROWN, GARY FLAKE, ROGER JONES, MICHAEL O'ROURKE, Y. C. LEE
Applications of Neural Networks to Process Control and Modelling
Artificial Neural Networks, Proceedings of the 1991, Internat. Conf. Artificial Neural Networks, Vol 1, pp 321-326, June 1991.

[BOTTOU 92] LÉON BOTTOU, VLADIMIR VAPNIK, *Local Learning Algorithms*, Neural Computation 4, p888-900, 1992.

[BROOMHEAD 88] D. S. BROOMHEAD, DAVID LOWE, *Mutlivariable Functional Intrpolation and Adaptive Networks*
Complex Systems 2, p321-355, 1988

[CHEN 91] S. CHEN, C. F. N. COWAN, P. M. GRANT
Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks
IEEE Trans. Neural Networks, Vol 2, No 2, March 1991.

[CLEVELAND 88] WILLIAM S. CLEVELAND, SUSAN J. DEVLIN, ERIC GROSSE
Regression by Local Fitting, Journal of Econometrics 37, pp 87-114, 1988.

[FOSS 93] BJARNE FOSS, TOR A. JOHANSEN, *On Local and Fuzzy Modelling*,
Proc. 3rd Int. Industrial Fuzzy Control and Intelligent Systems, Houston, Texas, 1993

[FOSS 94] BJARNE FOSS, TOR A. JOHANSEN, AAGE V. SØRENSON, *Nonlinear Predictive Control using Local Models – applied to a Batch Process,* IFAC ADCHEM 94, Kyoto, Japan, 1994.

[GIROSI 93] FEDERICO GIROSI, MICHAEL JONES, TOMASO POGGIO
Priors, Stabilizers and Basis Functions: from regularization to radial, tensor and additive splines
A.I. Memo No. 1430, MIT, June 1993.

[HLAVÁCKOVÁ 93] KATERINA HLAVÁCKOVÁ, ROMAN NERUDA, *Radial Basis Function Networks*,
Neural Network World 1, p93-101, 1993.

[HUNT 94] KEN HUNT, ROLAND HAAS, RODERICK MURRAY-SMITH, *Extending the Functional Equivalence of Radial Basis Function Networks and Fuzzy Inference Systems*, Submitted for publication.

[JACOBS 91] R. A. JACOBS, M. I. JORDAN, S. J. NOWLAN, G. HINTON, *Adaptive mixtures of local experts*, Neural Computation 3, p29-87.

[JANG 93] J.S. RODGER JANG, C.T. SUN, *Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems*, IEEE Trans. on Neural Networks, Vol. 4, No. 1, Jan 1993.

[JOHANSEN 92A] TOR A. JOHANSEN, BJARNE A. FOSS, *Non-linear Model Representation for Adaptive Systems*, IEEE SICICI, Feb1992.

[JOHANSEN 92B] TOR A. JOHANSEN, BJARNE A. FOSS, *Constructing NARMAX models using ARMAX models* Technical Report 92-36-W, Institute for Technical Cybernetics, Univ. Trondheim, November 1992.

[JOHANSEN 92C] TOR A. JOHANSEN, BJARNE A. FOSS, *Representing and Learning Unmodeled Dynamics with Neural Network Memories*, American Control Conf., Chicago, June 1992.

[JOHANSEN 93A] TOR A. JOHANSEN, BJARNE A. FOSS, *State-space Modelling using operating Regime Decomposition and Local Models*, 12th IFAC World Congress, Sydney, 1993.

[JOHANSEN 93B] TOR A. JOHANSEN, BJARNE A. FOSS, *Identification of non-linear System Structure and Parameters using Regime Decomposition*, Univ. of Trondheim Tech. Report, 1993.

[JOHANSEN 94] TOR A. JOHANSEN, *Adaptive Control of MIMO Non-linear Systems using Local ARX Models and Interpolation*, IFAC ADCHEM 94, Kyoto, Japan, 1994.

[JONES 89] R. D. JONES, Y. C. LEE C. W. BARNES G. W. FLAKE, K. LEE P. S. LEWIS, S. QIAN, *Function Approximation and Time Series Prediction with Neural Networks,* IEEE, Vol 1, 1989.

[KAVLI 92] TOM KAVLI, *Learning Principles in Dynamic Control*, Dr. Scient Thesis, University of Oslo, 1992.

[LEONARD 92] J. A. LEONARD, M. A. KRAMER, L. H. UNGAR,, *A Neural Network Architecture that Computes its Own Reliability*, MIT Industrial Liason Program Report, 3–7–92, Dept. of Chemical Engineering.

[MOODY 89] J. MOODY, C. DARKEN, *Fast Learning in Networks of Locally Tuned Processing Units* Neural Computation 1, p281–294, 1989.

[MURRAY-SMITH 94] RODERICK MURRAY–SMITH, HENRIK GOLLEE, *Constructive Techniques for Modelling with Local Model Ellipsoidal Basis Function Nets*, Submitted for publication.

[NOBLE 88] BEN NOBLE, JAMES W, DANIEL, *Applied Linear Algebra*, Prentice-Hall International, 1988.

[PANTALEON 93] CARLOS J. PANTALEÓN-PRIETO, FERNANDO DÍAZ-DE-MARIA, ANÍBAL FIGUEIRAS-VIDAL *On training RBF Networks*, Neuro-Nimes, 1993.

[POGGIO 90] THOMAS POGGIO, FEDERICO GIROSI, *Networks for Approximation and Learning,* Proc. IEEE, Vol 78, No. 9, Sept. 1990.

[RÖSCHEISEN 92] MARTIN RÖSCHEISEN, REIMER HOFMANN, VOLKER TRESP *Neural Control for Rolling Mills: Incorporating Domain Theories to Overcome Data Deficiency* Advances in Neural Information Processing , Vol 4., Morgan Kaufman, p659-666, 1992.

[SANNER 92] ROBERT M. SANNER, JEAN-JAQUES E. SLOTINE, *Gaussian Networks for Direct Adaptive Control* IEEE Trans. on Neural Networks, Vol. 3, No. 6, Nov. 92.

[SBARBARO 92] DANIEL SBARBARO-HOFER, *Connectionist Feedforward Networks for Control of Nonlinear Systems* Ph.D. Thesis, Dept. Mech. Eng., Glasgow University, 1992.

[SKEPPSTEDT 92] ANDERS SKEPPSTEDT, LENNART LJUNG, MILLE MILLNER, *Construction of composite models from observed data*, Int. Journal of Control, Vol. 55, No. 1, pp 141-152, 1992.

[STOKBRO 90] K. STOKBRO, D. K. UMBERGER, J. A. HERTZ, *Exploiting Neurons with Localized Receptive Fields to Learn Chaos*, Complex Systems 4, pp 603-622, 1990.

[SUGENO 88] M. SUGENO, G.T. KANG, *Structure Identification of fuzzy model*, Fuzzy Sets and Systems, Vol. 26., p15-33, 1988.

[TAKAGI 85] TOMOHIRO TAKAGI, MICHIO SUGENO *Fuzzy Identification of Systems and Its Applications to Modeling and Control.* IEEE Trans. on Systems, Man and Cybernetics, Vol. 15, No. 1, p116-132, 1985.

[WAHBA 90] GRACE WAHBA, *Spline Models for Observational Data*, SIAM, Philadelphia, PA, March 1990.