A Constructive Learning Algorithm for Local Model Networks

Roderick Murray-Smith, Henrik Gollee

Daimler-Benz AG,
Alt-Moabit 91b, D-10559 Berlin, Germany.
E-mail: {murray}{gollee}@DBresearch-berlin.de

Abstract

Local Model Networks are flexible architectures for the representation of complex non-linear dynamic systems. The local nature of the representation leads to a modular network which can integrate a variety of paradigms (neural nets, statistics, fuzzy systems and *a priori* mathematical models), but because of the power of the local models, the architecture is less sensitive to the curse of dimensionality than other local representations, such as Radial Basis Function networks. The concept of 'locality' is a difficult one to define, and tends to vary over a problem's input space, so a constructive structure identification algorithm is presented which automatically defines a suitable model structure on the basis of the observed data from the process being identified. Local learning algorithms are introduced for the local model parameter optimisation, which save computational effort and produce more interpretable and robust models.

1. Introduction

Computationally intensive learning systems capable of representing multivariable nonlinear systems have been the centre of a great deal of attention in the last decade. In many cases, however, there has been little attempt to provide any link to conventional modelling and identification theory, or to make the resulting trained systems interpretable to the user. There has been a recent upsurge of interest in neural networks and spline or kernel based architectures with local representation. This popularity is based to a great extent on the local nature of the approximation which makes the system more interpretable, and the fact that a great deal of the classical optimisation and regularisation theory can be applied to such architectures. A major problem, however, is that as one would expect with *local* representations, the architectures are highly susceptable to the *curse of dimensionality*. This paper describes an architecture which is flexible enough for the designer to limit the effect of high dimensionality, especially for dynamic systems, and a constructive structure identification algorithm which automatically determines the 'suitable' level of locality throughout the input space. A further reduction in the effect of high dimension is the application of *local learning* methods to the parameter optimisation phase of learning, which are outlined in this paper.

2. Local Model Methods

In this paper we discuss variations on one main class of network, the Basis Function Network (BF Net) because of its suitability for modelling continuous nonlinear systems. BF nets are basically local methods, which are inherently modular representations and thus allow the easy integration of ideas and structures from other modelling paradigms.

2.1. BF Nets for Modelling

Basis Function Networks and their equivalents have been used for function approximation and modelling in various forms for many years: The original Radial Basis Function Nets came from Interpolation theory and are described in [1], where a basis function is associated with each training point. Potential Functions, Kernels and Spline Models are all similar structures. Smoothing methods such as Gaussian Kernel methods like other local averaging methods quickly become impractical in high-dimensional spaces. Recently BF neural networks have received a growing amount of attention from the neural network community, [2, 3, 4, 5, 6, 7]. [8] gives a brief review of RBF nets. The basic structure is defined in more detail in our companion paper [9]. We are trying to model a process, where the underlying system can be described as

$$y = f(\boldsymbol{\psi}) + e \qquad (1)$$

where the input vector $\boldsymbol{\psi} \in \mathcal{R}^{n_\psi}$.

The output of the BF net is a weighted (by parameters $\boldsymbol{\theta}_i$) linear combination of many (n) locally active non-linear *basis functions* $\rho_i(\cdot)$,

$$\hat{y} = \hat{f}(\boldsymbol{\psi}) = \sum_{i=1}^{n} \theta_i \rho_i(\boldsymbol{\psi}). \qquad (2)$$

Each unit's centre is a point in the input space, and the *receptive field* of a unit i (the volume of the input space to which it reacts) is defined by its distance metric $d(\boldsymbol{\psi}; \mathbf{c}_i, \sigma_i)$, e.g. the Euclidean distance metric divided by a scalar value (σ_i) representing the basis function's radius,

$$d(\boldsymbol{\psi}; \mathbf{c}_i, \sigma_i) = \left\| \frac{\boldsymbol{\psi} - \mathbf{c}_i}{\sigma_i} \right\|^2 \qquad (3)$$

The *basis* or *activation function* (similar to the *membership function* of a fuzzy set) of the unit is usually designed so that the activation is local (i.e limited to the area near the centre), and that it smoothly and monotonically decreases towards zero as the input point moves away from the unit's centre ($\mathbf{c}_i$), e.g. B-Splines or Gaussian bells are common choices,

$$\rho_i(\boldsymbol{\psi}) = \rho(d(\boldsymbol{\psi}; \mathbf{c}_i, \sigma_i)) = \exp(-d(\boldsymbol{\psi}; \mathbf{c}_i, \sigma_i)), \quad (4)$$

There are a number of advantages to this style of structure, where the nonlinear representation is linearly weighted to the output. The optimisation of the parameters is based on standard regression techniques, and methods such as regularisation, experiment design and recursive estimation are easier to transfer to such structures.

2.1.1 Are BF Nets too local?:

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) the problem can be simplified by partitioning 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.

The 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 *a priori* knowledge of internal structures within the given system. The form of 'locality' utilised depends on the representation used; in decision trees, locality is introduced by partitioning the input space into hypercubes, in RBF nets locality is hyperspherical and in multilayer perceptrons or Projection Pursuit nets 'locality' is a given projection of the input space.

The problem with standard basis function networks is that the crudeness of the local approximation (weighted piecewise constant models) 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 exist!). It is therefore important to be able to profit from the local nature of the basis functions while not having to have 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 while achieving the desired modelling accuracy.

2.2. Local Model Basis Function Nets

The representational ability of the normal Basis Function (BF) net can be extended to a generalised form of BF network, the *Local Model Network* described in equation (5), where the basis functions are used to weight general functions of the inputs as opposed to straightforward weights,

$$\hat{y} = \hat{f}(\boldsymbol{\psi}) = \sum_{i=1}^{n} \hat{f}_i(\boldsymbol{\psi}) \rho_i(\tilde{\boldsymbol{\phi}}). \qquad (5)$$

The basis, or model validity functions used in this paper are radial, i.e. they use a distance metric (equation (3)) $d(\tilde{\boldsymbol{\phi}}; \mathbf{c}_i, \sigma_i)$ which measures the distance of the current *operating point* $\tilde{\boldsymbol{\phi}}$ (which is not necessarily the same as the full model input $\boldsymbol{\psi}$, but usually a subset of $\boldsymbol{\psi}$) from the basis function's centre $\mathbf{c}_i$, relative to the width variable σ_i. If the units have localised receptive fields, leading to *a limited degree of overlap with their neighbours* the functions $\hat{f}_i(\boldsymbol{\psi})$ can be viewed as locally accurate models, whose validity for a given input $\boldsymbol{\psi}$ is indicated by their basis function's value for that input. The basis functions are *normalised*, so that they sum to unity. See [10] for a discussion of the side-effects of normalisation.

This means that the structure has the advantages inherent to the local nature of the basis functions while, because of the more powerful local models associated with the basis functions, not requiring as many basis functions as before to achieve the desired accuracy. The improvement is more significant in higher dimensional problems. This style of representation was suggested in [3], followed up by [11] and [12]. The idea of using locally accurate models is also described in the statistical literature in [13], where local linear or quadratic models are weighted by smoothing functions, and Priestley's *State Dependent Models* [14] have many similarities to LMN's. LMN's for modelling and control of dynamic systems have been used extensively by Johansen and Foss [15, 16, 17].

The n local models used are represented by general functions $\hat{f}_i(\boldsymbol{\psi})$, but in many cases simple linear models are chosen

$$\mathcal{M}_i(\boldsymbol{\theta}_i) = \hat{f}_i(\boldsymbol{\psi}) = \left[1 \; \boldsymbol{\psi}^T\right] \boldsymbol{\theta}_i, \qquad (6)$$

where the parameters local model i, are $\boldsymbol{\theta}_i \in \mathcal{R}^{n_\psi+1}$ The network form of equation is shown in Figure 1. The trained network structure can be viewed as a decomposition of the complex, nonlinear system into a set of locally active submodels, which are then smoothly integrated by their associated basis functions.

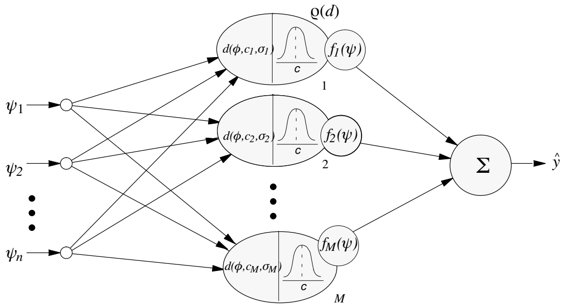


Figure 1: Local Model Basis Function network

To give an impression of the workings of a local model network, a one dimensional function is mapped using local models in Figure 2. The top plot (a) shows the target function and the model's approximation, while the basis functions and associated local models are shown in (b).

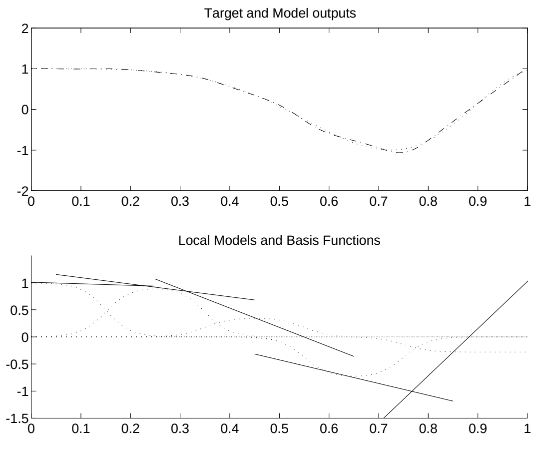


Figure 2: An example of local models representing a one dimensional function

3. Pre-structuring the Local Model

A major advantage of the local model nets is their ability to allow the introduction of *a priori* knowledge to define the model structure for a particular problem. This leads to more interpretable models which can be more reliably identified from a limited amount of observed data.

3.1. Incorporating local models based on *a priori* knowledge

The most general form of information is the expected dynamic order of the system, and the form of model to be identified (e.g. simple linear ARX models etc). If more knowledge is available, the local models could be physically oriented models, possibly with only a subset of their variables to be identified, thus allowing the engineer to create *grey-box* models. A generalised form would allow the designer to specify a pool of feasible local models, which could be locally tested for suitability in the various operating regimes defined by the basis functions.

In many cases, there will not be sufficent data to train the model throughout the entire input space. This is especially true in areas outside normal operation, where the model may have to be very robust, and well understood, but where no data can be acquired. These situations can be covered by fixing *a priori* models in the given areas, and applying learning techniques only where the data is available and reliable.

3.2. Incorporating *a priori* knowledge of nonlinearity into the Model Structure

Locality of representation 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. The problems of automatically discovering structure in high dimensional spaces can be reduced in many systems, as there are often combinations of input dimensions which are known *a priori* to be of no interest, or which are additively or linearly related, thus allowing the user to treat the system as an additive combination of lower dimensional sub-models. (The use of such physically based knowledge makes on-line adaptation of the system's parameters much more feasible). In [9] we make use of the strong links between Fuzzy membership functions and Basis Functions to initially decompose the problem by expressing structure as linguistic rules with accompanying basis functions.

The decomposition of the input space is especially interesting for nonlinear, high order dynamic systems, as the input space can be very large (and in practice such high dimensional input spaces are often impossible to fill with data), but the nonlinearity may only be dependent on a small number of the inputs. Local model nets are well suited for modelling such systems because although the system may be globally strongly nonlinearly dependent on the inputs, if the important subset of the inputs is used to partition the input space, the system can then be locally approximated sufficiently accurately by simple (possibly linear) models which use the entire input vector. In the dynamic systems case, an important reduction in the input space used for the nonlinear partition could be achieved by not including all of the delayed values of the inputs and state in $\tilde{\boldsymbol{\phi}}$, while all are present in $\boldsymbol{\psi}$ (i.e. the dimension of $\tilde{\boldsymbol{\phi}}$ is smaller than that of $\boldsymbol{\psi}$, as shown in figure 3).

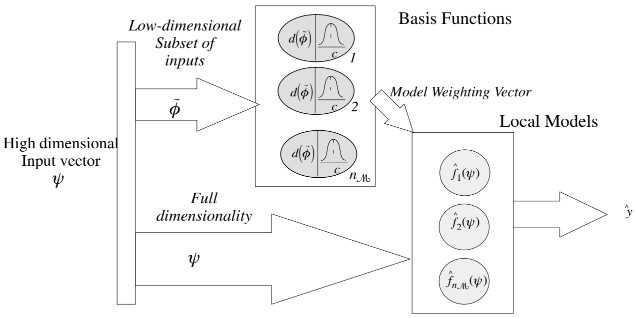


Figure 3: A mixed order hybrid system

4. Learning in Local Model Basis Function Networks

The learning task for Local Model Networks is to try to adapt their structure and parameters to minimise a cost functional related to the deviation of the model from the target system. The optimisation of the parameters[1]. is a linear process and relatively straightforward (assuming uncorrelated noise on the outputs), while the optimisation of the number of basis functions and their position and determining a suitable level of 'locality' is a difficult nonconvex problem, where *ad hoc* methods of reducing the complexity play an important role.

We will therefore split up the description of the learning process into these two highly interrelated stages, *structure identification* (Section 4.2) and *parameter identification* (Section 4.1), which are repeated until the desired structure and parameters are found.

4.1. Parameter Estimation

Parameter identification for local model nets, where the local models are linear in the parameters, is reasonably well understood, with a variety of efficient optimisation algorithms existing to solve the problem of optimising the parameters $\boldsymbol{\theta}$ to minimise the cost functional $J(\boldsymbol{\theta}, \mathcal{M}, \mathcal{D})$ for a given local model structure $\mathcal{M}$ and training set $\mathcal{D} = (\boldsymbol{\psi}(t-1), y(t)), t = 1..N$. Parameter optimisation for a given model structure finds the optimal cost J^* such that

$$J^*(\mathcal{M}, \mathcal{D}) = \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}, \mathcal{M}, \mathcal{D}). \tag{7}$$

The optimisation process generally used for the parameters of the local models is usually a conventional linear regression technique, where all of the parameters are optimised simultaneously using a single pseudoinverse calculation (see companion paper [9] for details). The linear regression problem is well understood, and there is a range of excellent algorithms for this task. The problem is that it is not always computationally feasible if a large number of parameters are to be identified from a large number of training patterns. 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 independently of neighbouring nodes. Also, even with a robust method such as SVD, because of ill-conditioning, the 'optimal' (in a least squares sense) network parameters will often fail to produce a robust model. In [18] we suggested a local learning method to get round these problems, which optimises the local model parameters for each local model independently (again, see the companion paper [9] for further details), leading to computational cost scaling linearly with the number of local models, rather than to the cube of the number of parameters, as in the global case.

4.2. Structure Identification

The use of existing *a priori* knowledge, as described earlier, to define the model structure is important, but as many problems are not well enough understood for the model structure to be fully specified in advance, it will often be necessary to adapt the structure for a given problem, based on information in the training data. The optimisation of the network structure $\mathcal{M}$ is, however, a difficult nonconvex optimisation problem, and is probably the most important area of research for basis function networks, if they are to be applied to demanding modelling problems where little is known about the model structure in advance.

The goal of the structure identification procedure is to provide a problem-adaptive learning scheme which automatically relates the density of basis functions and the size of their receptive fields to the local complexity and importance of the system being modelled. The aim is to find a model structure $\mathcal{M}$ which allows the network to best minimise the given cost function in a robust manner. Minimising $J^*(\mathcal{M}, \mathcal{D})$, from equation (7), over the possible model structures leads to the 'super-optimal' cost, using *a priori* knowledge about the process structure $\mathcal{K}_{\mathcal{S}}$,

$$J^{**}(\mathcal{D}, \mathcal{K}_{\mathcal{S}}) = \min_{\mathcal{M}} J^*(\mathcal{M}, \mathcal{D}). \tag{8}$$

[1] In this work the term *parameters* refers to the parameters of the local models and does not include the basis function parameters, the centres and widths, which are deemed to define the *model structure* $\mathcal{M}$

The robustness is an important aspect, as constructive structure identification algorithms can obviously be very powerful, enabling the network to represent the training data very accurately by using a large number of parameters, but usually then leading to a high variance. The choice of model structure plays a major role in the bias-variance trade-off (see [19] for details about the trade-off), and this should be reflected in the cost functions $J$and J^* in the form of regularisation terms for J and terms which penalise over-parameterisation in the structure functional J^*.

5. Constructive Algorithm for Structure Identification

As a result of the above considerations we suggest a constructive structure identification algorithm where the model structure is gradually improved, forming a coarse representation of the system initially, and refining it at each step by adding a new local model where it will bring the greatest improvement in accuracy. This involves a loop of the following manner:

1. Identify model parameters from training data.
2. Determine where model structure most needs improvement.
3. Improve model structure, if necessary and feasible.
4. Goto 1

To produce efficient, practical algorithms the following observations should be noted:

1. Although highly desireable, the distribution of training data will probably not be directly related to the complexity of the observed process.
2. The mapping will probably have varying levels of complexity in different areas of the input space.
3. The training data will not be uniformly distributed, and there will be areas of the input space which *cannot* be filled with data.

The first point implies that we should consider the local complexity of the system in the selection of cluster points, as opposed to unsupervised learning techniques, which only consider the density of the input data, regardless of the output response. The second point implies the need for a multi-resolution technique which will find clusters representing varying volumes of the input space (varying levels of locality). The last point lets us reduce the volume of the input space we consider for new local models to only points covered by the available training data.

5.1. Complexity Detection – Where Are Extra Units Needed?

The multi-resolution cluster algorithm uses a 'complexity detection' heuristic to place new local models. The heuristic was inspired by the *Vector Field Approach to Cluster Analysis* [20]. Observation 3 above indicates that we should simplify the search task by assuming that the training data covers the significant areas of the input space adequately for the initial search (an assumption which must be true for any degree of learning to take place). Initially, therefore, each training point is viewed as a possible centre ($\mathbf{c}$) for a basis function, where $\mathbf{c} \in \mathcal{R}^{n_\phi}$, and the 'complexity' of the mapping in a windowed area ($\rho(\cdot)$ is the windowing function) around this point is measured, using

$$F_{total}(\mathbf{c}, \sigma) = \frac{1}{N} \sum_{i=1}^{N} \rho(d(\tilde{\boldsymbol{\phi}}_i, \mathbf{c}, \sigma)) |\Delta y_i| |e_i| \quad (9)$$

where N is the number of neighbouring data points

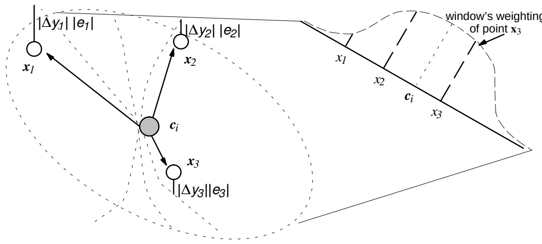


Figure 4: Windowed Complexity Estimate

used, and

$$\Delta y_i = y_i - y_{centre} \quad (10)$$

and

$$e_i = y_i - \hat{y}_i. \quad (11)$$

The function d is a distance measure. The complexity is estimated by an analogy to the concepts of forces acting on a mass in physics. The weighting of the forces depends on their associated residuals (e_i), and the difference from the prototype centre's output (Δy_i). The windowing function focusses the heuristic's attention on the level of locality currently being examined. The larger the level of F_{total}, the larger the estimated complexity.

The major disadvantage of the complexity heuristic is its computational load. It can require a maximum of N^2 calculations of the weighting function and the associated offset. This can be reduced by limiting the search by using only a subset of the training data as potential centres, or by the use of hierarchical decomposition of the input space to limit the number of data points under consideration, as in [21].

5.2. Overlap Determination

Given a set of basis function centres we need to define the basis function widths, which determine the level of overlap between neighbouring local models. When basis functions are used where centres can be distributed unevenly throughout the input space finding the 'correct' degree of overlap is a difficult problem. The conventional method is to set the radius σ_j proportional to the average distance of the k nearest neighbours from the centre $\mathbf{c}_j$,

$$\sigma_j \propto \frac{1}{k} \sum_{i=1}^{k} ||\mathbf{c}_j - \mathbf{c}_i||. \qquad (12)$$

In many cases this will be unsatisfactory, as the immediately neighbouring units could be widely varying distances apart in different directions, meaning that the resulting level of overlap with the neighbouring basis functions would vary greatly. Too much overlap and there are problems with poor estimation and singularities in the regression process. With too little overlap, and normalisation the mapping loses smooth interpolation between local models, and the behaviour further away from the centre becomes unpredictable, because of the interaction with other basis functions. The overlap problem is further complicated when basis functions are normalised.

To avoid this problem, more powerful distance metrics, e.g. the ellipsoidal distance metric in equation (13), where $\boldsymbol{\sigma}_i$ is a positive definite square matrix, as in [5, 22] can be used.

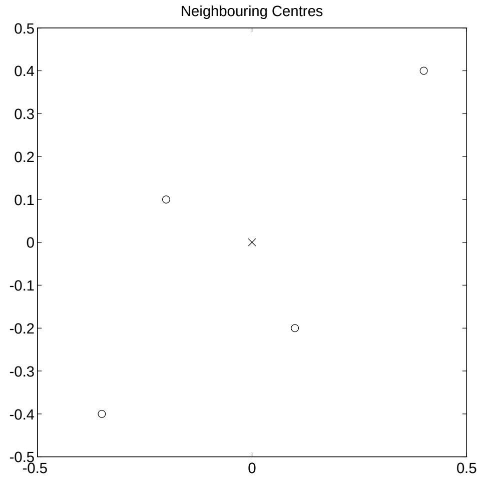
$$d(\tilde{\boldsymbol{\phi}}; \mathbf{c}_i, \boldsymbol{\sigma}_i) = \left|\tilde{\boldsymbol{\phi}} - \mathbf{c}_i\right|^T \boldsymbol{\sigma}_i{}^{-1} \left|\tilde{\boldsymbol{\phi}} - \mathbf{c}_i\right| \qquad (13)$$

This potentially allows a more even level of overlap with the neighbouring units[2].

The problem is therefore to robustly determine the matrix $\boldsymbol{\sigma}$ for each basis function from the distance of the neighbouring units. We used a heuristic which calculated the 'covariance' of the centres $\mathbf{c}_n$ surrounding the chosen centre $\mathbf{c}_i$ (the 'mean'), and using the inverse of this to define the distance function, as in equation (13).

$$\boldsymbol{\sigma}_i \propto E_n\left[(\mathbf{c}_n - \mathbf{c}_i)(\mathbf{c}_n - \mathbf{c}_i)^T\right], \qquad (14)$$

where E_n is the expected value over the chosen neighbouring centres $\mathbf{c}_n$. The results of this technique are shown in figure 5. 'x' is the centre of the unit being adjusted, and the 'o' points are the centres of neighbouring units. The surface plot infigure 5 shows the form of the basis function produced. The problem with this technique is that we are interested in distance to the nearest neighbours in all directions around the centre, so it is not sufficient to take the k nearest, as they could all be in the same area. To avoid this, we used a heuristic which demanded a minimum angle between neighbouring centres before they were included in $\mathbf{c}_n$ to be used for the covariance estimate, as shown in figure 6. While producing intuitively pleasing results for lower dimensional problems, this technique becomes less robust with increasing input dimension. The ellipsoidal basis functions demand an extra $\frac{n(n+1)}{2}$ free parameters, and the variance inherent in this increase in parameterisation is not trivial. To reduce the variance we used singular value decomposition (SVD) to determine the inverse of the 'covariance' matrix $\boldsymbol{\sigma}$. This allows us to apply simple regularisation ideas to the distance metric by altering the singular values, and thus reducing the



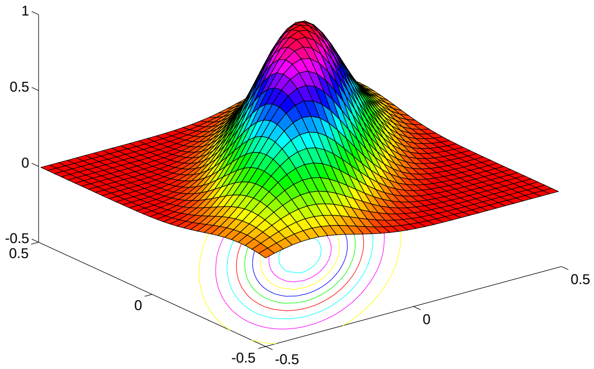


Figure 5: Using the 'covariance' measure to determine the basis functions size and orientations

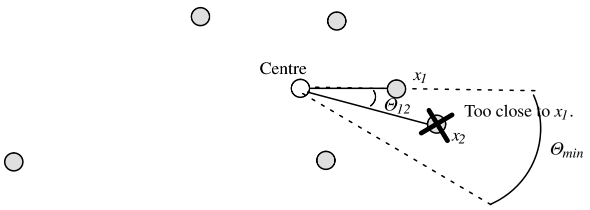


Figure 6: Eliminating centres with a common direction

[2]The use of such distance metrics becomes more significant where more powerful local models are used, allowing the model to be composed of fewer, but more powerful units.

degrees of freedom in the final distance metric. In our experiments, however, we found that the minimal improvement in performance compared to the simple RBF net was usually not enough to justify their use. Axis-orthogonal elliptical basis functions, where $\boldsymbol{\sigma}$ is a diagonal matrix, can be easily derived by only considering the diagonal terms of the covariance matrix, and tended to be lead to more robust networks. Figures 7 and 8 show the contours (at a level of 0.5) of the basis functions found for a two-dimensional problem with the given centres, using the three styles of distance metric with and without normalisation. However, when the basis

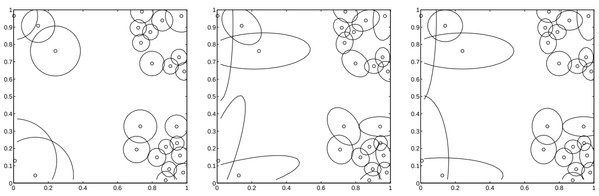


Figure 7: Radial, Ellipsoidal and Axis-orthogonal ellipsoidal basis functions

functions are normalised, there is often little difference between the various options: The heuristic

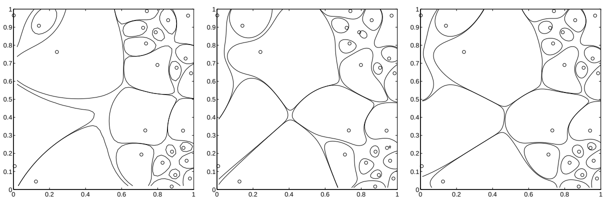


Figure 8: Normalised Radial, Ellipsoidal and Axis-orthogonal ellipsoidal basis functions

method described in this section could be seen as a fast initialisation routine, which could be further optimised by iterative gradient techniques, as in [5].

5.3. Preventing Overfitting

The problem of differentiating between errors due to noise on the training data and errors due to bias caused by an inadequate model structure is often a difficult one. Overtraining is the result of extending the structure so far that noise is learned, instead of system structure. A simple constraint on the structure identification algorithm is to require a minimum number of data points within the receptive field of a given local model for it to be considered viable, i.e. only if $\rho_{total_l} > N_{min}$, where N_{min} is the minimum number of training points needed and $\rho_{total_l} = \sum_{i=1}^{N} \rho_l(\tilde{\boldsymbol{\phi}}_i)$ is a heuristic 'count' of the local data points for smoothly overlapping basis functions. N_{min} is dependent on the level of noise on the data and the complexity of the local model.

Further pruning techniques, which merge neighbouring local models if their parameters are similar enough have also been successfully applied to simplify the networks. These rely on the use of local learning techniques [18], to ensure that the parameters have a strictly local interpretation.

5.4. Local Model Structure Selection

The structure optimisation extends also to the model structures of the individual local models. These need not be homogeneous, the user can define a pool of possible local models which can then be inserted into a given operating regime, with the 'best' one being chosen. This would allow a more robust fitting of models to operating regimes, taking the amount of data and the local process complexity into account.

5.5. The Constructive Process

In order to robustly determine the model structure a constructive approach is proposed. The algorithm starts off with a minimal representation (perhaps only one linear model) and searches for 'coarse' complexity (use of a large windowing function) initially and then refines the model at ever increasing levels of resolution until the desired accuracy has been achieved, or the data is exhausted. To determine where to add the extra representation the 'complexity' heuristic is used to decide where new models should be placed. Given the new model centre, which must be a minimum distance (which depends on the size of the 'complexity window') from existing centres, the desired overlap with neighbouring regions is determined, and the local model parameters for the new mdoel structure are re-estimated. If the model is still not accurate enough, the search for the next most 'complex' area of the input space is then restarted. This is repeated until the added models do not bring any improvement, whereupon the scale of the 'complexity window' is reduced (although how rapidly to reduce the windowing size is still a matter for experimentation), and the search is restarted at the finer resolution.

Such a gradual approximation has several advantages:

- The learning process is automated somewhat, in that the model structure is determined by the learning algorithm.
- It allows the network to first allocate representation where most needed, according to the complexity functional.
- Generalisation tends to be improved, as the required model structure is created, while

the overfitting protection inherent to the constructive algorithm limits overtraining.

- *A priori* knowledge can be introduced in the form of a pool of local model structures, so that the model best suited to a local area of the input space is chosen.
- The proportion of the training set used can also be selectively extended during learning to have a density matching the density of the basis functions, improving the quality of the parameter estimates, and speeding up the learning process. If there is insufficient data in certain areas of the input space, the constructive algorithm could be linked to an *active learning* procedure.

The structure identification problem is, however, far from solved, and this should remain a major research area in the near future.

6. Conclusions

The local model networks described in this paper allow the introduction of *a priori* knowledge about the process being modelled which reduces the dimensionality of the problem. The learning process can usefully be seen as being composed of two parts, *structure identification* and *parameter identification*, the structure identification phase of which is more difficult. A constructive algorithm for the automatic determination of the 'locality' of the model structure was described. A local method for parameter estimation was outlined which is more robust and far more efficient than the global method. The algorithms have been applied to real industrial data sets with success. Details of the applications will be given in the talk.

References

[1] M. J. D. Powell, "Radial Basis Functions for multivariable interpolation: A review," in *In: J.C. Mason & M.G. Cox, eds., Algorithms for Approximation*, (Oxford), pp. p143–167, Clarendon Press, 1987.

[2] J. Moody and C. Darken, "Fast-learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.

[3] R. D. Jones, Y. C. Lee, C. W. Barnes, G. W. Flake, K. Lee, P. S. Lewis, and S. Qian, "Function approximation and time series prediction with neural networks," Tech. Rep. 90-21, Los Alamos National Lab., New Mexico, 1989.

[4] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.

[5] T. Poggio and F. Girosi, "Networks for approximation and learning," in *Proceedings of the IEEE*, vol. 78, pp. 1481–1497, 1990.

[6] R. Murray-Smith, D. Neumerkel, and D. Sbarbaro-Hofer, "Neural Networks for Modelling and Control of a Non-linear Dynamic System," in *IEEE Symposium on Intelligent Control, Glasgow*, pp. p404–409, 1992.

[7] D. Neumerkel, R. Murray-Smith, and H. Gollee, "Modelling dynamic processes with clustered time-delay neurons," in *Proc. International Joint Conference on Neural Networks, Nagoya, Japan*, 1993.

[8] K. Hlaváčková and R. Neruda, "Radial Basis Function networks," *Neural Network World*, vol. 1, pp. 93–101, 1993.

[9] K. J. Hunt, R. Haas, and R. Murray-Smith, "Dimensionality reduction in basis-function networks: exploiting the link with fuzzy systems," in *Proc. IEEE Workshop on Computer-intensive methods in control and signal processing, Prague, Czech Republic*, 1994.

[10] R. Shorten and R. Murray-Smith, "On Normalising Basis Function networks," in *4th Irish Neural Networks Conf., Univ. College Dublin*, Sept. 1994.

[11] K. Stokbro, D. K. Umberger, and J. A. Hertz, "Exploiting neurons with localized receptive fields to learn chaos," *Complex Systems*, vol. 4, no. 3, pp. 603–622, 1990.

[12] C. Barnes, S. Brown, G. Flake, R. Jones, M. O'Rourke, and Y. Lee, "Applications of neural networks to process control and modelling," in *Artificial Neural Networks, Proceedings of 1991 Internat. Conf. Artif. Neur. Nets*, vol. 1, pp. p321–326, 1991.

[13] W. S. Cleveland, S. J. Devlin, and E. Grosse, "Regression by local fitting," *Journal of Econometrics*, vol. 37, pp. 87–114, 1988.

[14] M. B. Priestley, *Non-linear and Non-stationary Time Series Analysis.* Academic Press, 1988.

[15] T. A. Johansen and B. A. Foss, "Nonlinear local model representation for adaptive systems," in *Proceeding of the Singapore Int. Conf. on Intelligent Control and Instrumentation*, vol. 2, pp. 677–682, February 1992.

[16] T. A. Johansen and B. A. Foss, "Constructing NARMAX models using ARMAX models," *Int. J. Control*, vol. 58, no. 5, pp. 1125–1153, 1993.

[17] T. A. Johansen and B. A. Foss, "Identification of non-linear system structure and parameters using regime decomposition." To be presented at the IFAC Symposium on System Identification, Copenhagen, 1994.

[18] R. Murray-Smith, "Local Model Networks and Local Learning," in *Fuzzy Duisburg, '94*, pp. p404–409, 1994.

[19] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992.

[20] H. C. Andrews, *Introduction to Mathematical Techniques in Pattern Recognition.* Robert E. Krieger, 1983.

[21] R. Murray-Smith, "A Fractal Radial Basis Function network for modelling," in *Inter. Conf. on Automation, Robotics and Computer Vision, Singapore*, vol. 1, pp. NW–2.6.1 – NW–2.6.5, 1992.

[22] M. Röscheisen, R. Hofmann, and V. Tresp, "Neural control for rolling mills: Incorporating domain theories to overcom data deficiency," in *Advances in Neural Information Processing*, vol. 4, pp. 659–666, Morgan Kaufmann Publishers, San Mateo, Ca., 1992.