2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

On the Smoothness in Local Model Networks

Benjamin Hartmann and Oliver Nelles

Automatic Control, Mechatronics
Department of Mechanical Engineering
University of Siegen
D-57068 Siegen, Germany
benjamin.hartmann@uni-siegen.de

***Abstract*— This paper compares flat and hierarchical model structures in local model networks and discusses the side effects of normalization. A new algorithm for automatic transition adjustment between local models avoids undesirable effects that occur with the hierarchical approach and leads to a suitable model structure with better interpretability of local models. Demonstration examples illustrate the advantages over the existing approaches.**

I. INTRODUCTION

In the last two decades, architectures based on the interpolation of local models have attracted more and more interest as static function approximators and particularly as nonlinear dynamic models. Local *linear* models allow the transfer of many insights and methods from the mature field of linear control theory to the nonlinear world. Recent advances in the area of convex optimization and the development of efficient algorithms for the solution of linear matrix inequalities have contributed significantly to the boom on local linear model structures.

The output $\hat{y}$ of a local model network with p inputs $\underline{u} = [u_1\ u_2\ \cdots\ u_p]^T$ can be calculated as the interpolation of M local model outputs $\hat{y}_i$, $i = 1, \ldots, M$, see Fig. 1 [10],

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i(\underline{u})\Phi_i(\underline{u}) \tag{1}$$

where the $\Phi_i(\cdot)$ are called interpolation or validity or weighting functions. These validity functions describe the regions where the local models are valid; they describe the contribution of each local model to the output. From the fuzzy logic point of view (1) realizes a set of M fuzzy rules where the $\Phi_i(\cdot)$ represent the rule premises and the $\hat{y}_i$ are the associated rule consequents. Because a smooth transition (no switching) between the local models is desired here, the validity functions are smooth functions between 0 and 1. For a reasonable interpretation of local model networks it is furthermore necessary that the validity functions form a *partition of unity*:

$$\sum_{i=1}^{M} \Phi_i(\underline{u}) = 1\,. \tag{2}$$

Thus, everywhere in the input space the contributions of all local models sum up to 100%.

This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft (DFG), project code NE 656/3-1).

In principle, the local models can be chosen of arbitrary type. If their parameters shall be estimated from data, however, it is extremely beneficial to choose a linearly parameterized model class. The most common choice are polynomials. Polynomials of degree 0 (constants) yield a neuro-fuzzy system with singletons or a normalized radial basis function network. Polynomials of degree 1 (linear) yield local linear model structures, which is by far the most popular choice. As the degree of the polynomials increases, the number of local models required for a certain accuracy decreases. Thus, by increasing the local models' complexity, at some point a polynomial of high degree with just one local model ($M = 1$) is obtained, which is in fact equivalent with a global polynomial model ($\Phi_1(\cdot) = 1$).

Besides the possibilities of transferring parts of mature linear theory to the nonlinear world, local *linear* models seem to represent a good trade-off between the required number of local models and the complexity of the local models themselves. Due to the overwhelming importance and for simplicity of notation the rest of this paper will deal only with local models of linear type:

$$\hat{y}_i(\underline{u}) = w_{i,0} + w_{i,1}u_1 + w_{i,2}u_2 + \ldots + w_{i,p}u_p\,. \tag{3}$$

However, an extension to higher degree polynomials or other linearly parameterized model classes is straightforward.

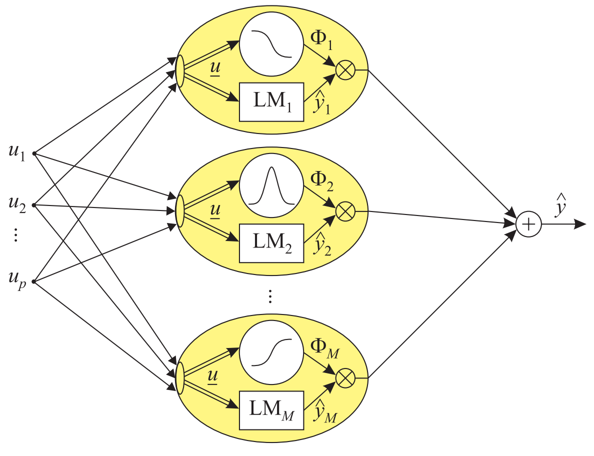

Fig. 1. Local model network: The outputs $\hat{y}_i$ of the local models (LM$_i$) are weighted with their validity function values $\Phi_i(\cdot)$ and summed up.

One of the key features of local model networks is that the input spaces for the local models and for the validity functions can be chosen independently. In the fuzzy interpretation this means that the rule premises (IF) can operate on (partly) other variables than the rule consequents (THEN). With different input spaces (1) has to be extended to, see Fig. 2:

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i(\underline{x})\Phi_i(\underline{z}) \tag{4}$$

with $\underline{x} = [x_1 \; x_2 \; \cdots \; x_{nx}]^T$ spanning the consequent input space and $\underline{z} = [z_1 \; z_2 \; \cdots \; z_{nz}]^T$ spanning the premise input space. This feature enables the user to incorporate prior knowledge about the strength of nonlinearity from each input to the output into the model structure. Or the other way round, the user can draw such conclusions from a black-box model which has been identified from data.

Especially for dynamic models where the model inputs include delayed versions of the physical inputs and output, the dimension nx becomes very large in order to cover all dynamic effects. In the most general case (universal approximator) this is also true for nz. However, for many practical problems a lower-dimensional $\underline{z}$ can be chosen, sometimes even one or two scheduling variables can yield sufficiently accurate models.

If the validity functions once are determined, it is easy to efficiently estimate the parameters of the local linear models w_{ij} by local or global least squares methods. The decisive difference between all proposed algorithms to construct local linear model structures is the strategy to partition the input space spanned by $\underline{z} = [z_1 \; z_2 \; \cdots \; z_{nz}]^T$, i.e., to choose the validity regions and consequently the parameters of the validity functions. This strategy determines the key properties of both: the construction algorithm and the finally constructed model.

Section II analyzes the influence of covariance matrix and side effects of normalization in local model networks with Gaussian basis functions. In Sect. III a strategy of hierarchical modeling is presented. Besides the discussion of the key features of global estimation, local estimation and interpolation smoothness in local model networks, Section IV introduces a new algorithm for the adjustment of the transitions between local models in hierarchical model networks. The performance of the hierarchical model strategy in combination with an automatic smoothness adjustment is evaluated with demonstration examples in Sect. V. This paper ends by summarizing the important conclusions.

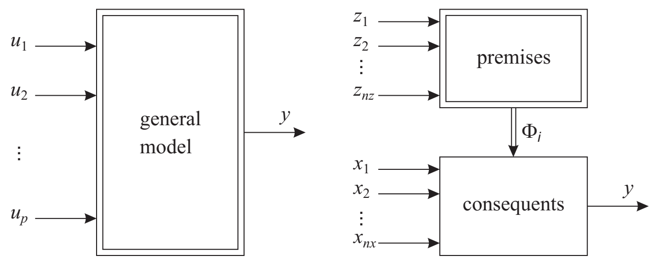

Fig. 2. For local model networks the inputs can be assigned to the premise and/or consequent input space according to their nonlinear or linear influence on the model output.

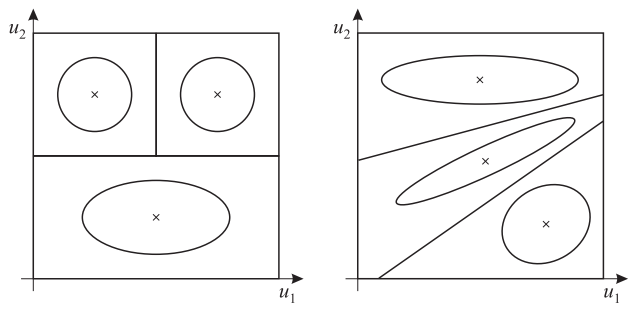

Fig. 3. The circles and ellipses represent the contour lines of the multidimensional membership functions μ_i (i.e., before normalization) in case of axes-orthogonal (left) and axes-oblique partitioning (right).

II. FLAT MODEL STRUCTURE WITH GAUSSIAN BASIS FUNCTIONS

This section gives a theoretical investigation about axes-orthogonal and axes-oblique input space partitioning strategies in local model networks with Gaussian basis functions. One key feature of this network type is its flat structure, i.e., all validity functions can be computed in parallel. Side effects of normalization in local model networks are analyzed.

A. Influence of the covariance matrix

To construct a Gaussian basis function the distance x_i from a data point to each center $\underline{c}_i$ is calculated with the help of the covariance matrix $\underline{\Sigma}_i$, which scales and rotates the axes:

$$x_i = ||\underline{z} - \underline{c}_i||_{\underline{\Sigma}_i} = \sqrt{(\underline{z} - \underline{c}_i)^{\mathrm{T}} \, \underline{\Sigma}_i^{-1} \, (\underline{z} - \underline{c}_i)} \, . \tag{5}$$

The membership functions $\mu_i(\cdot)$ of a Gaussian basis function network are given by:

$$\mu_i(\underline{z}) = \exp\left(-\frac{1}{2}||\underline{z} - \underline{c}_i||^2_{\underline{\Sigma}_i}\right) \, . \tag{6}$$

To achieve a *partition of unity* the membership functions have to be normalized to obtain the validity functions:

$$\Phi_i(\underline{z}) = \frac{\mu_i(\underline{z})}{\sum\limits_{j=1}^{M} \mu_j(\underline{z})} \, . \tag{7}$$

Depending on the covariance matrix two cases for partitioning the input space can be distinguished, see Fig. 3:

1) *Axes-orthogonal partitioning:* If the input space is divided into rectangular regions by axes-orthogonal splits the covariance matrix becomes diagonal:

$$\underline{\Sigma}_i = \begin{bmatrix} \sigma_{i,1}^2 & 0 & 0 & 0 \\ 0 & \sigma_{i,2}^2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \sigma_{i,nz}^2 \end{bmatrix} , \tag{8}$$

where $\sigma_{i,1}^2, \ldots, \sigma_{i,nz}^2$ are the variances of each input direction. In case of univariate membership functions the tensor product

$$\mu(z_1) \cdot \mu(z_2) \cdot \ldots \cdot \mu(z_{nz}) = \mu(z_1, z_2, \ldots, z_{nz}) \quad (9)$$

is utilized as construction mechanism. The multidimensional membership function realized by each neuron can be described with nz center values $c_{i,j}$ and nz variances $\sigma_{i,j}^2$. One big advantage of axes-orthogonal partitioning is its easy interpretability in terms of fuzzy logic. This way of partitioning allows a projection of the validity regions to the one-dimensional input variables. Undesirable normalization side effects and extrapolation behavior can be improved compared to clustering or data-based strategies [13], [10]. Common approaches are e.g. CART [1] and LOLIMOT [12], see also [14], [7].

2) *Axes-oblique partitioning:* In case of an axes-oblique partitioning strategy that is used for e.g. Gustafson-Kessel [6] or Gath-Geva clustering [5] the covariance matrix becomes symmetric ($nz \times nz$). A fuzzy interpretation of the model is not really possible, because the projection to the input axes is not possible without loss of information.

B. Side effects of normalization

The normalization (7) can lead to some very unexpected and usually undesirable effects, which are illustrated in [13]. In case of a diagonal covariance matrix these effects do *not* occur if all basis functions (BFs) possess identical standard deviations for each dimension, i.e.,

$$\sigma_{1j} = \sigma_{2j} = \ldots = \sigma_{Mj} \,. \quad (10)$$

The side effects are caused by the fact that for all very large input values the activation of the Gaussian BF with the largest standard deviation becomes higher than the activation of all other Gaussian BFs. As shown in Fig. 4, the normalization then results in a reactivation of the basis function with the largest width. This reactivation makes the validity functions non-local and multi-modal – both are properties usually not intuitively assumed and not desired in such networks.

These problems translate to the multidimensional case. If axes-orthogonal partitioning is used, these side effects can always be analyzed by examining the one-dimensional projections to the input variables. However, in case of scaled and rotated Gaussian BFs (axis-oblique partitioning) the normalization can lead to highly nontransparent reactivation regions in the input space, see Fig. 5. These side effects are typically not very significant for the performance of a Gaussian BF network, but they are of fundamental importance with regard to its interpretation.

The next section introduces a hierarchical model structure that can circumvent the above mentioned drawbacks of the flat structure.

III. HIERARCHICAL MODEL STRUCTURE

Almost all of the common partitioning strategies yield flat models. Even if the *algorithm* is hierarchically organized like e.g. LOLIMOT, the constructed *model* itself is flat in the sense that all validity functions $\Phi_i(\cdot)$ can be calculated in parallel. This is an important feature if the network really should be realized in hardware or by some parallel computer. For any standard software implementation a hierarchical model structure as shown in Fig. 6 is not disadvantage. Strictly speaking hierarchical model networks do not belong to the group of neural networks because the parallelization of neurons is not applicable. But in case of sequential computer architectures hierarchical model approaches can be favorably. Truly hierarchical model structures are e.g. pursued with CART [1], MARS [4], hinging hyperplane trees [3], and the hierarchical local model network [8], see also [11].

As an example, a model with four rules shall be constructed in a hierarchical manner. The model is represented by a binary tree as shown in Fig. 6. At each knot i of the tree the input space is softly partitioned into two areas by two splitting functions $\Psi_i(\cdot)$ and $\widetilde{\Psi}_i(\cdot)$ which sum up to one:

$$\Psi_i(\underline{z}) + \widetilde{\Psi}_i(\underline{z}) = 1 \,. \quad (11)$$

These regions are further subdivided by succeeding knots (if they exist). Each leaf of the tree realizes a local model and its contribution to the overall model output is given by the multiplication of all splitting functions from the root of the tree to the corresponding leaf. For the tree with seven knots and four leaves in Fig. 6 this means:

$$\begin{aligned}\Phi_3(\underline{z}) &= \widetilde{\Psi}_1(\underline{z}) \,, \quad \Phi_4(\underline{z}) = \Psi_1(\underline{z})\Psi_2(\underline{z}) \,, \\ \Phi_6(\underline{z}) &= \Psi_1(\underline{z})\widetilde{\Psi}_2(\underline{z})\Psi_5(\underline{z}) \,, \quad \Phi_7(\underline{z}) = \Psi_1(\underline{z})\widetilde{\Psi}_2(\underline{z})\widetilde{\Psi}_5(\underline{z}) \,.\end{aligned}$$

Once these validity functions are determined, the overall model output is given similarly to (1) as the sum the local

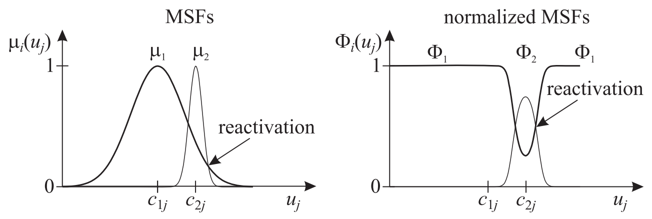

Fig. 4. Reactivation of basis functions occurs if the activation of the Gaussian BF with the largest standard deviation becomes higher than the activation of all other Gaussian BFs.

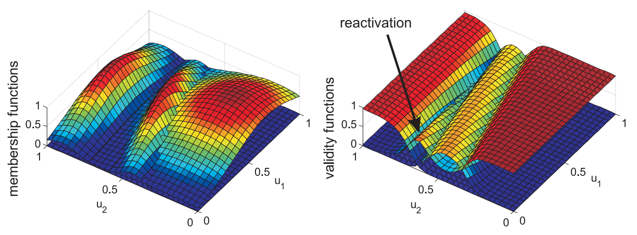

Fig. 5. Normalization of axes-oblique membership functions leads to unexpected reactivation regions in the input space. Projection to the one-dimensional input variables delivers no adequate information about these side effects.

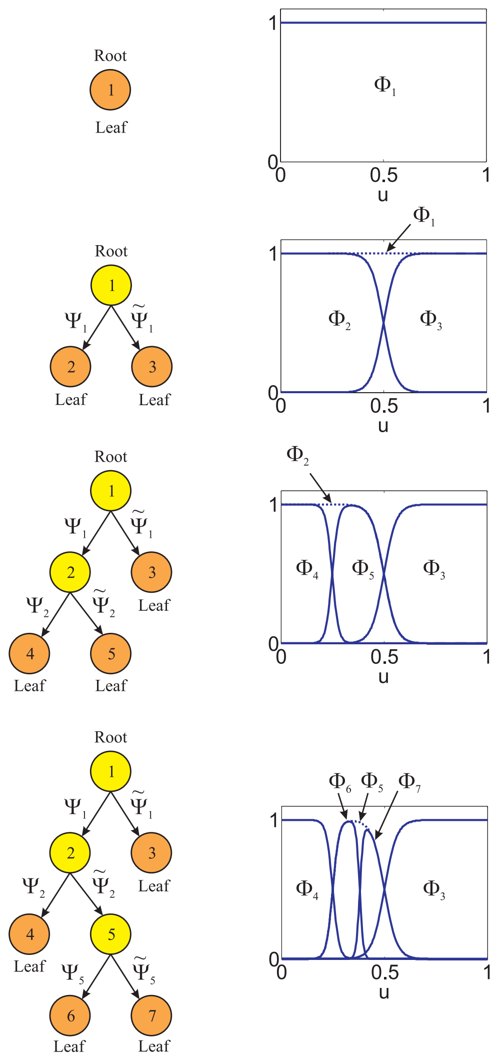

Fig. 6. Hierarchical tree construction algorithm. Every leaf of the tree represents one validity function Φ_i and is generated by multiplying the corresponding splitting functions Ψ_i.

models weighted with their associated validity functions:

$$\hat{y} = \sum_{i \in \mathcal{L}} \hat{y}_i(\underline{x})\Phi_i(\underline{z}) \qquad (12)$$

where $\mathcal{L}$ is the set of indices of the leaf knots, which is in the above example $\mathcal{L} = \{3, 4, 6, 7\}$.

One of the major advantages of such hierarchical model structures compared to flat ones is that the partition of unity in (2) holds automatically if only (11) is fulfilled. This also avoids all undesirable normalization side effects (comp. sect. II-B) because the normalization in (7) is not needed to create a partition of unity [13], [10]. A further motivation for this approach is the fact that both model leaves and whole model sub-trees could be changed or adapted without taking any effect on tree regions of higher hierarchical levels. However, the erasement of the disadvantages of flat model structures goes along with a problem that concerns the adjustment of the overlaps between local models. This is discussed in the next section.

IV. SMOOTHNESS ADJUSTMENT

The goal of approximation with local model networks is i) the interpretability of local models and ii) a smooth global model behaviour. The necessity of adjustment of overlaps between local models depends on the used parameter estimation approach. Two different approaches for optimization of the local model parameters can be distinguished: *global* and *local* estimation. While global estimation represents the straightforward application of the least squares algorithm, local estimation neglects the overlap between the validity functions in order to exploit the local features of the model [2], [8].

A. Global and local estimation:

In case of global estimation the following characteristics can be listed:

- High training effort.
- High flexibility in case of large overlaps of validity functions, but then interpretability of local models is lost.
- High accuracy if data has low noise level.
- In case of hard switching between local models global and local estimation tend to the same results.

Local estimation seems to be advantageous to global estimation in most applications [10], [9]. The following benefits can be expected:

- Fast training.
- Regularization effect.
- Good interpretability of local models.
- Advantages when applied in a recursive algorithm for online learning.

B. How smooth should the interpolation be?

For the choice of the interpolation smoothness it is interesting to consider two special cases:

1) *smoothness too small:* hard switches between local models.
2) *smoothness too large:* big overlaps of validity functions and loss of locality.

Hence the smoothness should be a compromise between these cases. In Fig. 7 it is depicted how the interpolation smoothness influences the validity functions Φ_i and the transition between two local linear models $\hat{y}_1$ and $\hat{y}_2$.

In most cases it seems to be sufficient to roughly choose some reasonable *a priori* smoothness value for all local models. The activation of all validity functions should reach almost 100% in order to ensure a proper interpretation of the associated local models (in particular true for local parameter estimation). It has been observed that the model quality is only insignificantly dependent on the exact smoothness value [10], see also Fig. 8. Therefore a fine tuning is not necessary.

C. Smoothness optimization or a priori fixation?

With local estimation an optimization of the smoothness does not work satisfactorily in most cases. The reason is that a hard switching between local models also minimizes the

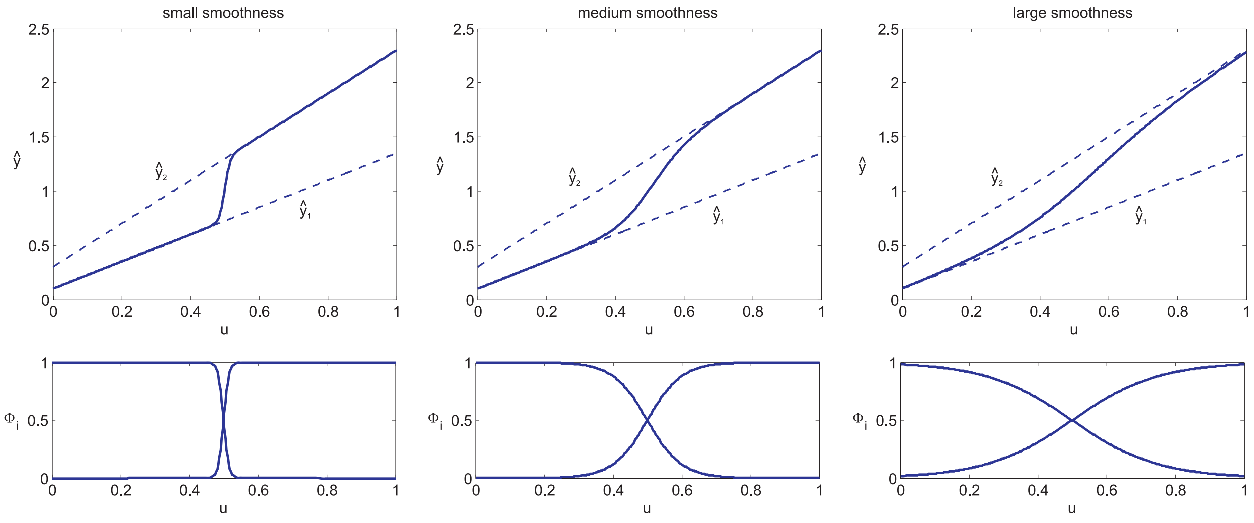

Fig. 7. Influence of interpolation smoothness on the model and relation between model output $\hat{y}$ and validity function Φ.

error caused by neglecting the overlap between the validity functions. However, even if optimal performance is achieved such small overlaps should not be realized in practice where the model is usually required to be smooth. In contrast with a global estimation approach a smoothness optimization tends to yield very large smoothness values ($\rightarrow\ \infty$), in order to overcome the restrictions in model flexibility due to the locality of the validity functions. Considering the low sensitivity of the model quality on the smoothness and all the difficulties arising during nonlinear optimization of the smoothness it can be generally recommended to fix it a priori. But in case of networks with hierarchical structure an additional unpleasant effect occurs: The overlap of validity functions depends on the hierarchy level of the local models. Since the splitting functions Ψ_i have to be multiplied with each other to generate the validity functions Φ_i, the activity of subsequent neurons is reduced with each hierarchy level. For example, the validity functions associated with knot 6 and 7 in Fig. 6 are always less active than the validity function associated to their mother knot 5. A reliable fixation of the smoothness a priori is only possible if also the final depth (or hierarchy level) of the hierarchical model structure would be known. To circumvent this drawback a new algorithm has been developed where the smoothness is adapted automatically after adding a local model to the hierarchical tree structure.

D. Automatic transition adjustment algorithm:

For each local model split the smoothness of the overall model is adjusted. The following loss function can be defined:

$$J = \left|\Phi_{\text{threshold}} - \min\left(\Phi_i(\underline{c}_i), \Phi_j(\underline{c}_j)\right)\right| , \qquad (13)$$

where $\Phi_{\text{threshold}}$ is an user defined value that gives a threshold for the minimal activation of all validity functions. Depending on the underlying process this threshold should be chosen as high as the locality of the validity regions needs to be high. The validity functions Φ_i and Φ_j are generated by division of an existing local model one level up in hierarchy. As long as J is larger than a residuum ε the transition between all local models has to be sharpened. This feature can be implemented with a single-parameter optimization or simply by an iterative sharpening of the transitions between local models. After manipulation of the model smoothness actually the parameters of all local models have to be re-estimated consecutively. But since the smoothness influences the model parameters only insignificantly the additional re-estimation procedure can be skipped in most applications. Experimental results showed a relative parameter difference of less than 0.1%.

The advantages of this smoothness adjustment approach are shown with demonstration examples in the following section.

V. DEMONSTRATION EXAMPLES

In order to illustrate the operation of the transition adjustment algorithm the following simple static SISO process will be utilized with input value interval $0 \leq u \leq 1$:

$$y = \frac{1}{0.1 + u} . \qquad (14)$$

In Fig. 8 the hierarchical modeling of this process with sigmoidal splitting functions is demonstrated. The first case shows the model without adjustment of transitions. The smoothness is calculated with respect to the center distances of local models and therefore yield validity functions with large overlaps. As a consequence of the hierarchical model structure (see Fig. 6) the last added local models have a smaller influence on the overall global model. The interpretability and locality tends to become lost.

In contrast all validity functions generated with inherent smoothness adjustment ($\Phi_{\text{threshold}} = 0.95$) have maximum activities over 95%. The normalized root mean squared error of the model drops from 5.49% to 3.79%.

As a second even more drastic demonstration example the process in Fig. 9 is approximated with 10 local linear models.

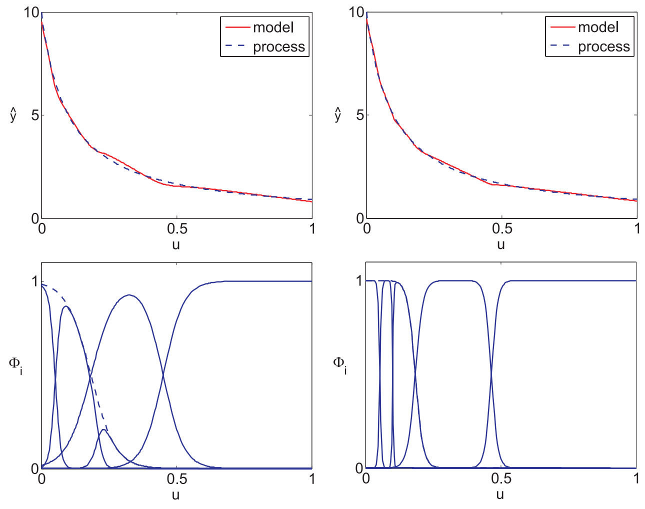

Fig. 8. Trained network without (left) and with adjustment of the transition (right) between local models and influence on model output (top) and validity functions (bottom).

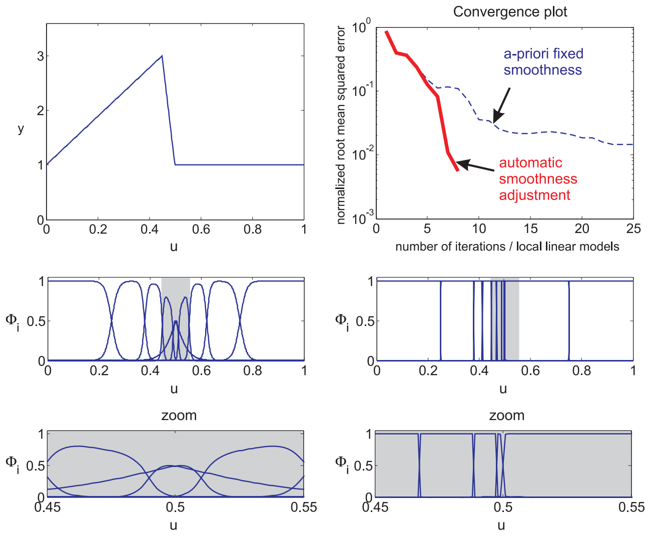

Fig. 9. Modeling of a test-process (top left) with hierarchical local model networks. Validity functions generated with a-priori fixed smoothness (middle/bottom left) and automatic transition adjustment algorithm (middle/bottom right). The convergence plot (top right) demonstrates the advantages of automatic smoothness adjustment.

The standard method using a-priori fixed smoothness values (left) is compared with the proposed automatic smoothness adjustment algorithm (right). The process has an almost step-like behavior (high gradient) at $u \approx 0.5$. So the model resolution has to be high in this region.

The convergence plot shows that the models of both methods yield to nearly the same normalized root mean squared error in case of 5 local models. But if the number of local models increases the model with a-priori fixed smoothness becomes too smooth. The refinement of the model around $u \approx 0.5$ achieves no improvement. In other words the local model network tends to loose the characteristic of an universal approximator. This unsatisfactory approximation behavior is a consequence of the too small validity function values. They directly result from the multiplications of the splitting functions due to the hierarchical model structure that is not known beforehand.

The here proposed smoothness adjustment algorithm automatically adapts the smoothness of the overall model as long as each neuron has an activity more than 95% at its center. So the global model smoothness is adapted to the resulting hierarchical model structure. As in the convergence plot illustrated, the model with automatic smoothness adjustment can be improved significantly with each iteration in contrast to the a-priori fixed smoothness approach.

VI. CONCLUSIONS

Normalization of basis functions in local model networks can lead to undesirable side effects. Especially networks with scaled and rotated membership functions are tending to reactivation of neurons so that interpretability and locality of the validity regions are lost. One remedy to overcome this drawback is the use of a hierarchical model structure. In combination with a new algorithm for automatic smoothness adjustment between local models this approach leads to reasonable results.

REFERENCES

[1] L. Breiman and C.J. Stone J.H. Friedman R. Olshen R. *Classification and Regression Trees*. Chapman & Hall, New York, 1984.

[2] W.S. Cleveland, S.J. Devlin, and E. Grosse. Regression by local fitting: Methods, properties, and computational algorithms. *Journal of Econometrics*, 37:87–114, 1996.

[3] S. Ernst. Hinging hyperplane trees for approximation and identification. In *IEEE Conference on Decision and Control (CDC)*, pages 1261–1277, Tampa, USA, 1998.

[4] J.H. Friedman. Multivariate adaptive regression splines (with discussion). *The Annals of Statistics*, 19(1):1–141, March 1991.

[5] I. Gath and A.B.. Geva. Unsupervised optimal fuzzy clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(7):773–781, 1989.

[6] D.E. Gustafson and W.C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *IEEE Conference and Decsion and Control*, pages 761–766, San Diego, USA, 1979.

[7] T.A. Johansen. Identification of non-linear system structure and parameters using regime decomposition. *Automatica*, 31(2):321–326, 1995.

[8] R. Murray-Smith. *A Local Model Network Approach to Nonlinear Modeling*. PhD thesis, University of Strathclyde, Strathclyde, UK, 1994.

[9] R. Murray-Smith and T.A. Johansen. Local learning in local model networks. In *IEE International Conference on Artificial Neural Networks*, pages 40–46, 1995.

[10] O. Nelles. *Nonlinear System Identification*. Springer, Berlin, Germany, 2001.

[11] O. Nelles. Axes-oblique partitioning strategies for local model networks. In *International Symposium on Intelligent Control (ISIC)*, Munich, Germany, October 2006.

[12] O. Nelles, S. Sinsel, and R. Isermann. Local basis function networks for identification of a turbocharger. In *IEE UKACC International Conference on Control*, pages 7–12, Exeter, UK, September 1996.

[13] R. Shorten and R. Murray-Smith. Side-effects of normalising basis functions in local model networks. In R. Murray-Smith and T.A. Johansen (Eds.), editors, *Multiple Model Approaches to Modelling and Control*, chapter 8, pages 211–229. Taylor & Francis, London, 1997.

[14] M. Sugeno and G.T. Kang. Structure identification of fuzzy model. *Fuzzy Sets & Systems*, 28(1):15–33, 1988.