Increasing the Performance of a Training Algorithm for Local Model Networks

Torsten Fischer, Benjamin Hartmann and Oliver Nelles

***Abstract*—In this paper the improvement of the established training algorithm HILOMOT is presented. HILOMOT is a hierarchical tree-construction method based on the ideas of neuronal networks and fuzzy-systems and an advancement of the well-known LOLIMOT-Algorithm. During the training the input space is divided into subregions with the help of validity functions. This is done iteratively by splitting the current worst local model, until an specified exit condition is reached. For every subregion one local linear model is estimated by a weighted least squares method. The main purpose is keeping the number of local models low. Therefore the split position is optimized during the training. The optimization problem is nonlinear, so a gradient-based nonlinear local optimization method, called Quasi-Newton method, is used. The main drawback of this approach is its long calculation time. The most time consuming part is the numerical calculation of the gradient done by the finite difference technique.To avoid this problem the numerical approach is replaced by the analytical gradient. This leads to a significant reduction of the training time without decreasing the approximation quality.**

***Index Terms*—neuronal networks, fuzzy-logic, system identification, nonlinear optimization.**

I. Introduction

MOST of the commonly used methods in automatic control engineering require a well formulated model of the system in order to be controlled. Many real processes are too complex to use an analytical model. In these cases the experimental modeling, called *identification* is employed. Commonly used identification methods are neuronal networks and neuro-fuzzy systems. A method based on this ideas is the HILOMOT (**HI**erachical **LO**cal **MO**del **T**ree)-Algorithm [6], which can be seen as an advancement of the well-known LOLIMOT (**LO**cal **LI**near **MO**del **T**ree)-Algorithm [5], [7]. In Contrast to LOLIMOT, which uses orthogonal Gaussian membership functions for the partition, HILOMOT uses arbitrarily orientated Sigmoids. Consequently they are more flexible during the training and a better indication of the nonlinear characteristics of the process is afforded [3]. The price of a lager flexibility is a required nonlinear optimization, which is necessary in order to get the best segmentation of the input space with respect to the training error. One way to improve the performance of HILOMOT significantly in terms of training time is the application of an analytical gradient instead of a numerically derived gradient information. This paper increases the performance of the algorithm by improving this optimization procedure. Therefore, first a brief overview of the mode of operation of the HILOMOT Algorithm is given. Then the optimization of the split function used for partitioning is discussed and the difficulties are identified. The implemented approaches to overcome this problems are illustrated and the modifications are compared and evaluated to the optimization method used in an empirical examination so far. Different example processes are investigated.

T. Fischer, B. Hartmann and O. Nelles are with the Department of Mechanical Engineering, University of Siegen, Germany, email: torsten.fischer@uni-siegen.de, benjamin.hartmann@uni-siegen.de, oliver.nelles@uni-siegen.de

II. Principle Operation of the Algorithm

The basic idea of HILOMOT is the description of the training data with local sub-models of polynomial type. These local models are interpolated with so called validity functions. The local models are linearly parameterized, i.e. can be estimated with weighted least squares. In contrast to the local models, the validity functions contain nonlinear parameters. Therefore, these parameters only can be found by iterative methods. Other algorithm like CART [2] or LOLIMOT [5], [7] are popular methods that also follow this strategy too. These algorithms are heuristic tree-construction methods. They use heuristical approaches for finding the structure parameters in contrast to the HILOMOT, which performs a nonlinear optimization of its structure parameters. Therefore, they are much faster, but have a lower flexibility concerning the split position and direction. The general principle of all of them (Fig. 1) can be interpreted as a fuzzy model in Takagi-Sugeno form, which are very important for approximation strategies for nonlinear static and dynamic processes [1], [4]. The global model output $\hat{y}$

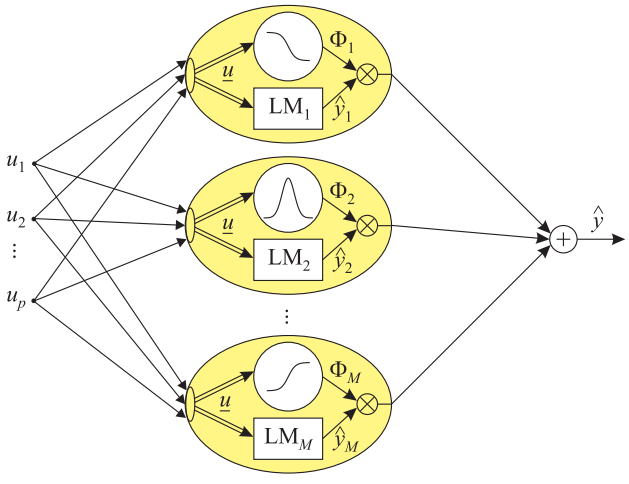

Fig. 1. Local model network: The output $\hat{y}_i$ of each local model is weighted by its validity function Φ_i and summed up to the global model output $\hat{y}$.

is a superposition of weighted local models, where each of the M local models is a rule in terms of fuzzy logic:

$$\hat{y} = \sum_{k=1}^{M} \hat{y}_k(\underline{x})\Phi_k(\underline{z})\,. \tag{1}$$

ISBN: 978-988-19252-4-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Here, the local models $\hat{y}_k$ are the associated rule consequents and the validity functions Φ_k represent the rule premises, with the vector $\underline{z}$ spanning the premise input space and the vector $\underline{x}$ spanning the consequent input space. They result from the input vector $\underline{u}$ of the real process, as illustrated in Fig. 2. By dealing with independent input spaces for the rule

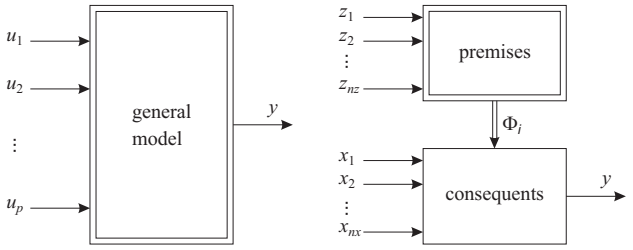

Fig. 2. The input vector $\underline{u}$ of the general nonlinear model, can be assigned to the premise (nonlinear) and/or consequent (linear) input space according to their influence of the output behavior.

premises and consequents, it is possible to incorporate prior knowledge about the nonlinear behavior between each input and the output into the model structure. Each local model $\hat{y}_k$ is defined by its parameter vector $\underline{w}_k$ and the vector $\underline{x}$:

$$\hat{y}_k = \underline{x} \cdot \underline{w}_k \ . \tag{2}$$

The parameters of the local models can be easily estimated by an local or global least-squares method, if the validity functions Φ_k are known. The local estimation of the parameter vector $\underline{w}_k$ is:

$$\underline{w}_k = \left(\underline{X}^T \cdot \underline{Q}_k \cdot \underline{X}\right)^{-1} \cdot \underline{X}^T \cdot \underline{Q}_k \cdot \underline{y} \ . \tag{3}$$

Here matrix $\underline{Q}_k$ is a diagonal matrix including the weights of the k-th local model, which are its validity function Φ_k, and matrix $\underline{X}$ is the input matrix of the consequent input space, whose columns represent one data point.

As mentioned in the introduction, HILOMOT uses sigmoids as splitting functions in contrast to LOLIMOT, which applies orthogonal gaussians. The arbitrary orientation of the sigmoids in the premise input space is their advantage [3]. A sigmoid is described by:

$$\Psi(\underline{z}) = \frac{1}{1 + e^{(v_0 + z_1 \cdot v_1 + \ldots + z_{nz} \cdot v_{nz})}} \ , \tag{4}$$

where vector $\underline{v}$ contains the parameters of the sigmoid and the vector $\underline{z}$ spans the premise input space, whose number of inputs is nz [5], [6]. A typical partitioning done over the first training iterations of HILOMOT is shown in Fig. 3. A split is realized by dividing the validity area of the worst local model into two sub-models by a sigmoid Ψ_1 and its complementary function $\tilde{\Psi}_1$. Each validity function is a product of all sigmoid function along the path beginning at the root and ending in the leafs. The complexity of the global model increases in each iteration, which means that a better approximation is achieved step by step. In order to get a good approximation with as less local models as possible, in each iteration the parameters of the new sigmoid are optimized. To guarantee a good initialization for the nonlinear optimization, all orthogonal splits are done in the premises input space and the best one with respect to the used global loss function is chosen as starting point. To

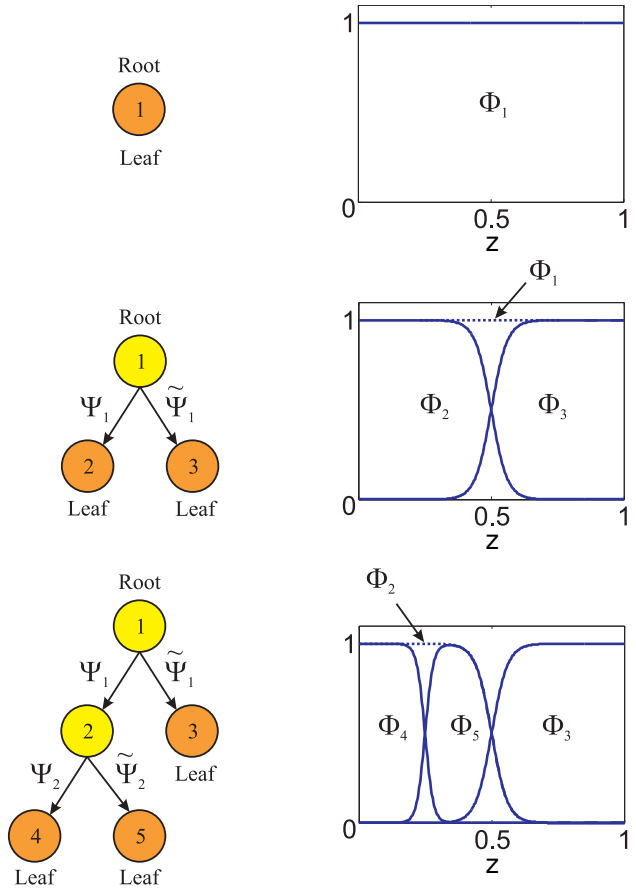

Fig. 3. Tapical tree structure: Splitting the worst local model in each iteration. The validity function of each leaf results from the multiplication of the all sigmoids functions along the path.

assess the optimization problem precisely, a closer look on the properties of a sigmoid has to be taken. The parameters of the sigmoid influence its position and steepness. The position of the sigmoid corresponds to the position of the split, which is taken during the training. The steepness specifies the smoothness of the transition between the adjoining local models. In order to discuss these statements in detail, Fig. 4 shows a schematic illustration of a two-dimensional Sigmoid. The level curve at $\Psi(u_1, u_2) = 0.5$ of the sigmoid

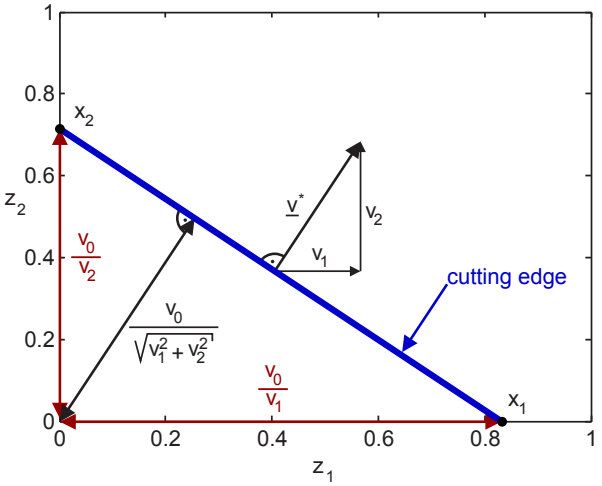

Fig. 4. Two-dimensional sigmoide with its parameters v_0, v_1 and v_2.

describes the splitting line, where the change of the bigger validity between both local models is given. For a higher dimensional premise input space the partitioning is done by hyperplanes. The direction of the split is described by the vector $\underline{v}^* = [v_1, v_2, \dots v_{nz}]$, which is called direction vector. The difference between the parameter vector $\underline{v}$ and the direction vector $\underline{v}^*$ is the parameter v_0. The orthogonal distance of the hyperplane to the origin is described by the ratio $v_0/\|\underline{v}^*\|$. With the direction and the distance the position of the split is defined accurately. Another possible representation can be done by vectors along the axes of the premise input space. They define the intersection x_j of the splitting line with the corresponding axis. Here we got two input dimensions and according to this the points x_1 und x_2, with $x_1 = v_0/v_1$ and $x_2 = v_0/v_2$. It is obvious that the position of the sigmoid can be defined only by the ratios of the parameters. The length of the vector specifies the steepness of the sigmoid, and therewith the character of the continuous transition between the local models. This relationship is shown in Fig. 5. Here several one-dimensional

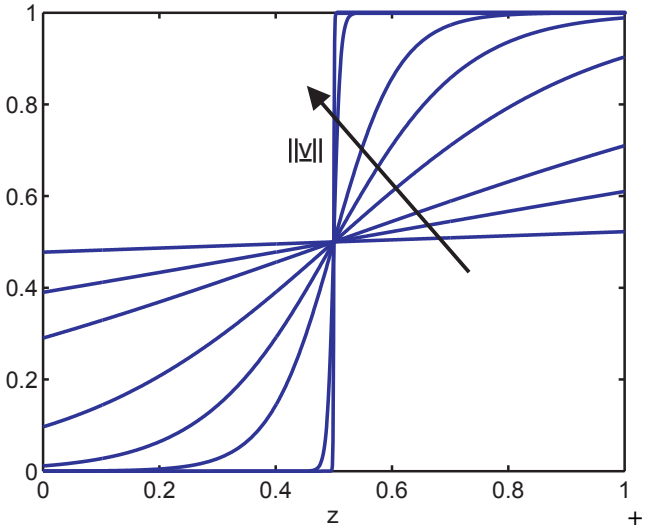

Fig. 5. One-dimensional sigmoids with different transitions.

sigmoids are illustrated, that all have the same point of intersection. The parameter vectors of the sigmoids differ only in their length, whereupon the ratios between the single parameters stay the same. The length of the parameter vectors of the sigmoids increases along the direction of the arrow. In summary, the ratio between the single parameters indicates the position of the sigmoid, which is the same as the position of the split and the length of the parameter vector defines the steepness of the sigmoid, which describes the smoothness of the transition between the adjoined local models. In LOLIMOT, the choice of the parameters of the gaussian results from heuristic geometrical considerations, which are easy to perform [5]. HILOMOT however uses arbitrary orientated sigmoids, whose nonlinear parameters can not result from heuristically approaches. Thats why they need to be optimized to reach a minimal number of local models to describe the process. The optimization is performed by the Quasi-Newton-Method, which belongs to the group of nonlinear local gradient-based optimization methods [5], [9]. Optimizing the parameter vector $\underline{v}$ leads to a very steep slope and a very hard transition between the local models. Using local estimation, this is the best solution in terms of the considered loss function. A hard transition is not intended. Therefore, the influence of the parameters on the steepness of the sigmoid needs to be removed by a standardization of the length of the vector $\underline{v}$ [6]. Hence, a parameter κ is introduced, which directly influences the sharpness of the transition:

$$\kappa = \frac{20}{\sqrt{\underline{v} \cdot \underline{v}^T} \cdot \sqrt{\underline{\Delta c} \cdot \underline{\Delta c}^T} \cdot \sigma} = \frac{20}{\|\underline{v}\| \cdot \|\underline{\Delta c}\| \cdot \sigma} \quad . \tag{5}$$

Combined with Eq. (4) follows:

$$\Psi(\underline{z}) = \frac{1}{1 + e^{\kappa \cdot (v_0 + z_1 \cdot v_1 + \dots + z_{nz} \cdot v_{nz})}} = \frac{1}{1 + e^{\kappa \cdot (v_0 + \underline{z} \cdot \underline{v}^*)}} \quad . \tag{6}$$

The distance of the centers of the adjoined local models Δc and the smoothness parameter σ, which is used to calibrate the overall transition behavior, are responsible for the steepness of the sigmoids [6]. The influence of the parameter vector $\underline{v}$ on the steepness is abrogated by the division with the norm of the vector included in κ.

III. PROBLEM STATEMENT AND APPROACHES

As mentioned before, the parameter κ neglects the influence of the parameter vector $\underline{v}$ on the smoothness of the transition between the local models by normalizing it. Thus, only the position of the split remains affected by the vector $\underline{v}$. Needing just the ratios of the parameters to define the exact position, one parameter becomes redundant. If the optimizer considers the whole vector $\underline{v}$, it handles a over-determined optimization problem. This proceeding causes a long calculation time and possibly a worse result. To avoid this, one parameter must be fixed during the optimization. Generally it is irrelevant which parameter is chosen. A good one is the offset v_0, because it is clearly separated form the premise input space matrix and the other parameters in Eq. (6). By fixing the parameter v_0, the problem is not over-determined anymore and reduced by one dimension. Hence, the performance of the optimizer and of the whole algorithm increases. In order to optimize the parameters, the Quasi-Newton-Method is used, which needs the gradient of the loss function I. If an analytical gradient is not given, a numerical approach is needed. The numerical calculation of the gradient, which the Quasi-Newton-Method typically uses, is done by the finite difference technique. Its drawback among other issues is its high computational effort with respect to number of parameters to be optimized [5], [9]. In order to avoid this problem, an analytical gradient should be implemented. The loss function appropriated by HILOMOT is the NRMSE (**N**ormalized **R**oot **M**ean **S**quared **E**rror):

$$I(\underline{\theta}) = \frac{\sqrt{\sum_{i=1}^{N} e_i^2(\underline{\theta})}}{\sqrt{\sum_{i=1}^{N} (y_i - \overline{y})^2}} \quad . \tag{7}$$

Therefore, the derivative of loss function I with respect to the direction vector $\underline{v}^*$ of the sigmoid needs to be calculated.

ISBN: 978-988-19252-4-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

This means the general equation of the gradient has to be changed into:

$$\underline{g}(\underline{\theta}) = \frac{\partial I(\underline{\theta})}{\partial \underline{\theta}} = \underline{g}(\underline{v}^*) = \frac{\partial I(\underline{v}^*)}{\partial \underline{v}^*} \ . \tag{8}$$

Here, the gradient consists of the derivatives with respect to each sigmoid parameter. Thus, the j-th entry of the gradient for the j-th sigmoid parameter can be specified by:

$$g_j = \left(\sum_{i=1}^{N} (y_i - \overline{y})^2 \right)^{-\frac{1}{2}} \cdot \frac{\partial}{\partial v_j} \left(\sum_{i=1}^{N} e_i^2(v_j) \right)^{\frac{1}{2}} \ , \tag{9}$$

where N is the number of data samples, y_i is the output at the i-th data sample, $\overline{y} = \frac{1}{N} \sum y_i$ the mean of all data samples and $e_i(v_j) = y_i - \hat{y}_i(v_j)$ is the error at the i-th data sample. Because just the global model is addicted to the vector $\underline{v}^*$ Eq. 9 becomes:

$$g_j = -F_1 \cdot \sum_{i=1}^{N} e_i \cdot \frac{\partial \left(\hat{y}_i(\underline{v_j}) \right)}{\partial v_j} \ , \tag{10}$$

In Eq. (10) a placeholder F_1 is used for a more compact formulation:

$$F_1 = \left(\sum_{i=1}^{N} (y_i - \hat{y})^2 \right)^{-\frac{1}{2}} \cdot \left(\sum_{i=1}^{N} e_i^2 \right)^{-\frac{1}{2}} \ . \tag{11}$$

The estimation of the parameters of the local models is performed by a weighted least squares approach and these weights result from the validity areas of the local models [6]. Furthermore the normalization of the parameters is done, which leads to additional terms in the gradient equation. After considering this addictions the following expression for each entry of the gradient results:

$$\begin{aligned} g_j = \frac{\partial I}{\partial v_j} = \\ - F_1 \cdot \sum_{i=1}^{N} e_i \cdot \Phi_i^* \cdot \Bigg\{ (\hat{y}_{a,i}(\underline{\theta}) - \hat{y}_{b,i}(\underline{\theta})) \cdot \frac{\partial \Psi_i}{\partial v_j} \\ + \underline{X}_i \cdot \left(\underline{X}^T \cdot \underline{Q}_a \cdot \underline{X} \right)^{-1} \cdot \underline{X}^T \cdot \frac{\partial \underline{Q}_a}{\partial v_j} \cdot \\ \left[\underline{y} - \hat{\underline{y}}_a \right] \cdot \Psi_i(\underline{\theta}) \\ - \underline{X}_i \cdot \left(\underline{X}^T \cdot \underline{Q}_b \cdot \underline{X} \right)^{-1} \cdot \underline{X}^T \cdot \frac{\partial \underline{Q}_a}{\partial v_j} \cdot \\ \left[\underline{y} - \hat{\underline{y}}_b \right] \cdot \tilde{\Psi}_i(\underline{\theta}) \Bigg\} \ . \end{aligned} \tag{12}$$

Φ_i^* is the validity function of the local model, which is split. $\hat{y}_{a,i}$ and $\hat{y}_{b,i}$ are the new local models arising out of the currently worst local model by splitting. The split is done by the sigmoid Ψ_i and its complementary function $\tilde{\Psi}_i$:

$$\tilde{\Psi}_i = 1 - \Psi_i \ . \tag{13}$$

$\underline{Q}_a$ and $\underline{Q}_b$ are the weighting matrices of the new local models. The calculation of the derivatives of the weighting matrix $\underline{Q}$ and of the sigmoids are done separately, because the equation would be too long and confusing. The derivatives of the weighting functions only vary in their sign:

$$\frac{\partial \underline{Q}_a}{\partial v_j} = -\frac{\partial \underline{Q}_b}{\partial v_j} \ . \tag{14}$$

Hence, it is just necessary to calculate the derivation of the diagonal weighting matrix $\underline{Q}_a$ as:

$$\frac{\partial \underline{Q}_a}{\partial v_j} = \begin{bmatrix} \Phi_1^* \cdot \frac{\partial \Psi_1}{\partial v_j} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \Phi_N^* \cdot \frac{\partial \Psi_N}{\partial v_j} \end{bmatrix} \ . \tag{15}$$

In Eq. (12) and (15) the derivative of the sigmoid needs to be calculated. Eq. (16) shows the associated formula. Because of the normalization of the parameter vector inside the exponential function, the equation is not really well formulated. Hence, two more placeholders are added in order to achieve a shorter expression:

$$F_{2,i} = \frac{e^{\kappa \left(v_0 + \underline{Z}_i \cdot \underline{v}^* \right)}}{\left(1 + e^{\kappa \left(v_0 + \underline{Z}_i \cdot \underline{v}^* \right)} \right)^2} \ , \tag{17}$$

and

$$F_3 = \frac{-20}{\sigma} \cdot \left(\|\underline{v}\| \cdot \|\underline{\Delta c}\| \right)^{-2} \ . \tag{18}$$

In Eq. 17 the matrix $\underline{Z}$ is the input matrix of the premise input space and its i-th column $\underline{Z}_i$ represent the i-th data point. With the equations above, the derivative of each parameter of the vector $\underline{v}^*$ is clearly defined. This incorporates the derivative of the sigmoid and the derivative of the weighting matrix. Obviously the calculation of the analytical gradient depends on the number of data points. The formula is analytical, but it is evaluated at discrete points. Thereby its vulnerable to low number of data points or scattered data points, which at worst causes the optimizer to crash. To guarantee a robust modeling and therewith a deterministic approximation, the orthogonal initial split has to be used, if the nonlinear optimization fails.

IV. Validation

To demonstrate the increased performance of the algorithm by the modification explained in the last section, three example processes are modeled and the results are compared against each other. The examples are a symmetric hyperbola and parabola, and the so called "radcos"-function, which was used during a dissertation to verify its results [8]. The formula for two input dimensions of the "radcos"-function is

$$\begin{aligned} y = \cos \left(9 \cdot \sqrt{u_1^2 + u_2^2} + 2 \right) + \\ \frac{1}{2} \cdot \cos \left(11 \cdot u_1 + 2 \right) + \\ 15 \cdot \left((u_1 - 0.4)^2 + (u_2 - 0.4)^2 \right)^2 \ , \end{aligned} \tag{19}$$

and Fig. 6 illustrates its curve progression. In the following, two optimization methods will be compared. The first one uses the reduced parameter vector for the optimization, but with a numerical gradient. The second method in this comparison employs the reduced parameter vector and the proposed analytical gradient. By comparing these methods, it is possible to get an idea which improvement is achieved by this modification. The used quality criterions are the global loss function of the training and the number of local models. The limit of the NRMSE, which is used as the training error, is 5% and the number of the local models is constrained to

ISBN: 978-988-19252-4-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$$\frac{\partial \Psi_i}{\partial v_j} = -F_{2,i} \cdot \left[\kappa \cdot Z_{i,j} + (v_0 + \underline{Z}_i \cdot \underline{v}^*) \cdot \frac{v_j \cdot \frac{\|\underline{\Delta c}\|}{\|\underline{v}\|} - 2 \cdot \frac{\|\underline{v}\|}{\|\underline{\Delta c}\|} \cdot \sum_{l=1}^{n} \Delta c_l \cdot \frac{1}{N} \cdot \sum_{k=1}^{N} u_{l,k} \cdot \Phi_k^* \cdot F_{2,k} \cdot \kappa \cdot Z_{k,j}}{\frac{1}{F_3} + 2 \cdot \frac{\|\underline{v}\|}{\|\underline{\Delta c}\|} \cdot \sum_{l=1}^{n} \Delta c_l \cdot \frac{1}{N} \cdot \sum_{k=1}^{N} u_{l,k} \cdot \Phi_k^* \cdot F_{2,k} \cdot (v_0 + \underline{Z}_k \cdot \underline{v}^*)} \right] . \quad (16)$$

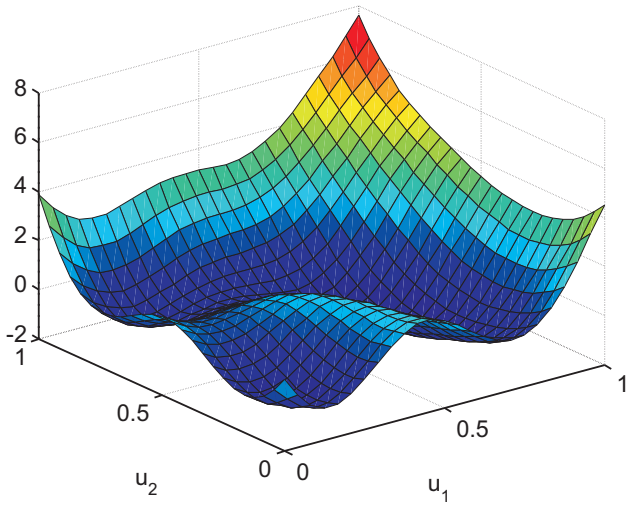

Fig. 6. The "radcos"–function with two input dimensions u_1 and u_2.

ten. Either the algorithms reach the error limit within the permitted number of local models or the maximum number of local models is used to gain the smallest possible NRMSE. The error values in Table I result from the given training data set. The quality of approximation is only slightly different

TABLE I
GLOBAL ERROR (NRMSE) OF THE TRAINING DATA SET IN PERCENT. THE BEST VALUE IN EACH COMPARISON IS HIGHLIGHTED.

error [%]	process	num. grad	anal. grad
2D	hyperbola	<5	<5
3D		<5	<5
4D		<5	<5
5D		<5	<5
6D		<5	<5
7D		<5	<5
2D	parabola	**15.15**	15.34
3D		24.84	**24.77**
4D		**45.30**	46.35
5D		59.93	**59.80**
6D		**65.53**	65.60
7D		**70.42**	70.62
2D	radcos	18.05	**17.30**
3D		**27.53**	28.43
4D		**37.9**	38.33
5D		42.57	**42.52**
6D		**44.76**	44.93
7D		**42.83**	43.18

between both methods. It depends on the example, which method delivers the best result. Regarding a separate data set, which is used for validation, similar results can be observed, as Table II shows. Likewise the training data, no arbitrative differences in the quality of the approximation can be seen. The effect of the usage of the analytical gradient on both error values can be neglected. Regarding the computing time shown in Fig. 7 and Table III, dramatical changes arise.

For all three examples for every dimension of the input

TABLE II
GLOBAL ERROR (NRMSE) OF THE TEST DATA SET IN PERCENT. THE BEST VALUE IN EACH COMPARISON IS HIGHLIGHTED.

error [%]	process	num. grad	anal. grad
2D	hyperbola	**5.38**	5.41
3D		4.93	**4.82**
4D		9.61	**9.60**
5D		**4.43**	4.54
6D		**4.89**	4.92
7D		**3.15**	3.18
2D	parabola	**14.94**	15.12
3D		24.91	**24.83**
4D		**46.55**	46.59
5D		61.02	**61.00**
6D		**68.49**	68.51
7D		74.01	**73.35**
2D	radcos	**11.08**	12.85
3D		**28.90**	29.69
4D		**38.97**	39.11
5D		44.12	**43.80**
6D		**46.71**	47.15
7D		46.90	**46.65**

space the analytical gradient has the lowest computational effort. If the new implemented analytical gradient is used, the

TABLE III
COMPUTION TIME* IN SECONDS. THE BEST VALUE IN EACH COMPARISON IS HIGHLIGHTED.

time*[s]	process	num. grad	anal. grad
2D	hyperbola	0.52	**0.45**
3D		1.00	**0.75**
4D		1.34	**0.6**
5D		2.62	**1.00**
6D		3.24	**1.00**
7D		4.42	**1.41**
2D	parabola	1.29	**0.82**
3D		2.51	**1.12**
4D		4.82	**1.38**
5D		7.05	**2.34**
6D		10.53	**2.64**
7D		19.79	**3.29**
2D	radcos	1.64	**1.08**
3D		3.33	**1.37**
4D		6.02	**1.70**
5D		6.65	**1.30**
6D		11.69	**2.75**
7D		15.29	**3.81**

*Windows 7 (64-Bit), Intel Core i7 M 620 @ 2,67 GHz, 4,00 GB RAM

computing time drops to 87% to 17% of the computing time with the numerical approach. Hence, the taken arrangements improve the performance of the algorithm considerably.

V. CONCLUSIONS

In this paper an improvement of the performance of the HILOMOT-algorithm [6] by modification of the intern nonlinear optimization is presented. Therefore the numerical calculation of the gradient is replaced by the analytical solution. First, the derivation of the loss function used in

ISBN: 978-988-19252-4-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

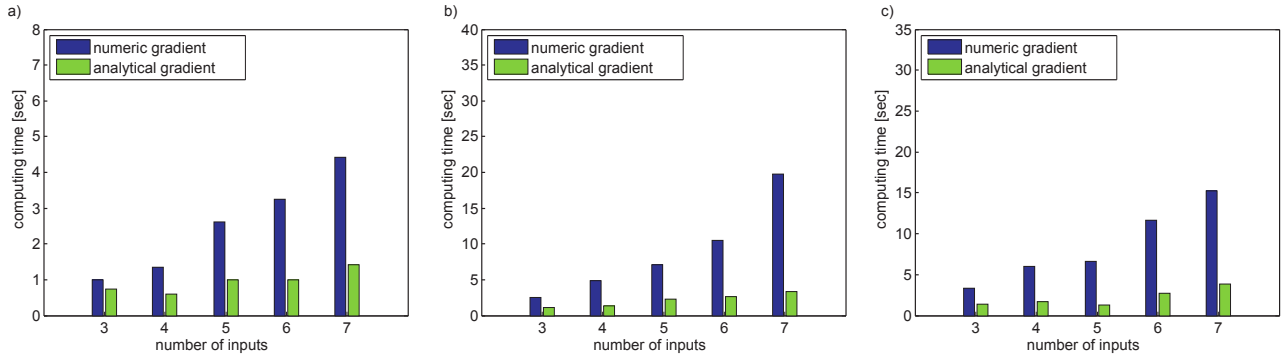

Fig. 7. Computing times of the three methods with 3 to 7 input dimensions for a) the hyperbola, for b) the parabola and for c) the "radcos"-function. The calculation was done by a computer using Windows 7 (64-Bit), Intel Core i7 M 620 @ 2,67 GHz with 4,00 GB-RAM.

HILOMOT, the NRMSE, with respect to the parameter of the sigmoid is done. The next modification is the implementation of an analytical gradient of the loss function. The analytical calculated gradient makes the finite difference technique superfluous, which requires especially for high-dimensional input spaces a high computational effort [5]. The empirical examination shows, that the new implemented algorithm fulfills the expectations regarding the increase of performance . The computing time is reduced significantly and the quality of the approximation remains constant.

REFERENCES

[1] R. Babuška and H.B. Verbruggen. An overview of fuzzy modeling for control. *Control Engineering Practice*, 4(11):1593–1606, 1996.
[2] L. Breiman and C.J. Stone J.H. Friedman R. Olshen R. *Classification and Regression Trees*. Chapman & Hall, New York, 1984.
[3] B. Hartmann and O. Nelles. Advantages of hierarchical versus flat model structures for high–dimensional mappings. In *Workshop Computational Intelligence*, Bommerholz, Germany, December 2009.
[4] T.A. Johansen, R. Shorten, and R. Murray-Smith. On the interpretation and identification of dynamic takagi-sugeno fuzzy models. *Fuzzy Systems, IEEE Transactions on*, 8(3):297–313, 2000.
[5] O. Nelles. *Nonlinear System Identification*. Springer, Berlin, Germany, 2001.
[6] O. Nelles. Axes-oblique partitioning strategies for local model networks. In *International Symposium on Intelligent Control (ISIC)*, Munich, Germany, October 2006.
[7] 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, Sept. 1996.
[8] J. Poland and A. Zell. Different criteria for active learning in neural networks: A comparative study. In *10. European Symposium on Artificial Neural Networks*, pages 119–124, 2002.
[9] L.E. Scales. *Introduction to Non-Linear Optimization*. Computer and Science Series. Macmillan, London, 1985.

ISBN: 978-988-19252-4-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)