Automated Order Determination Strategies for Nonlinear Dynamic Models

Julian Belz, Tim Oliver Heinz, Oliver Nelles
Department of Mechanical Engineering
University of Siegen
D-57068 Siegen, Germany
Email: http://www.mb.uni-siegen.de/mrt/mitarbeiter/

***Abstract*—A crucial part during the generation of nonlinear dynamic models is the determination of an appropriate model order. Five automated order determination strategies are compared. One model-based and four model-free approaches are investigated. We evaluated the performances of all methods with four artificial test processes and two noise levels. In an external dynamics approach, local model networks are trained with the determined (lagged) inputs and outputs that are found through the automated order determination strategies. An independent noise-free data set reveals the simulation quality of the estimated models. Most of the filter methods are unreliable since their performance varies strongly. Most robust is the wrapper method, which achieves good results in general. We show that in some cases even the model yielded through the incorporation of prior-knowledge is outperformed by some of the models resulting from the presented order determination methods.**

I. Introduction

For most modern techniques dealing with control related topics, e.g. model predictive control, high quality models are mandatory. In this paper we only consider experimental modeling (system identification), where the generation of the models heavily relies on measured data. The derivation of physical models as an alternative is often too complicated or at least too time consuming. Besides the measured data, choosing an appropriate dynamic model order is an important necessity for the generation of well-performing models. Often the determination of the dynamic order is done based on expert knowledge or a trial-and-error approach. Since accuracy demands as well as the complexity of the systems under investigation are more and more increasing, the traditional order determination methods reach their limits. Especially the increasing number of potential inputs (plus their delayed versions) requires more sophisticated, systematic investigation schemes to fulfill the accuracy demands. This paper compares several methods to automatically determine the dynamic order, that are potentially able to deal with the increased accuracy demands and the increasing system complexities.

As we are going to model nonlinear dynamic systems, the most frequently applied way for this purpose according to [1] is pursued. It is called external dynamics approach and is visualized in Fig. 1. Two parts can be distinguished: The external filter bank and the nonlinear static approximator. In this paper the filters are considered as simple time delays q^{-1} and for the nonlinear static approximator local model networks (LMN) are used. The order of the nonlinear dynamic model is determined by the number of delay blocks contained in the external filter bank. In this respect, order determination is equivalent to finding the most relevant inputs for the nonlinear static approximator and input selection methods can be applied to tackle this problem. In the following the inputs for the nonlinear static approximator are named net inputs, since they are the inputs to a LMN. It should be clear from now on, that both physical delayed inputs and delayed outputs are contained in the net inputs. Besides the model-based method presented here, there are several other possibilities, e.g. [2] and [3]. However, the used approach here offers special properties explained in detail in Sec. II-A. The aim of all methods that will be compared, is to find a necessary minimum of these net inputs, specifying the order of the nonlinear model. Reducing the number of net inputs to a necessary minimum yields several advantages, e.g. a higher density of data in the input space, fewer required parameters for the estimated model and an increased interpretability through a more concise description.

The methods for determination of the model order can generally be divided into two main groups: The model-based and the model-free methods. Model-based schemes can further be divided into wrapper and embedded methods according to [4]. Wrappers use a training algorithm as a black box and wrap the dynamic order determination around it. The evaluation criterion is directly related to the achieved model performance. In [5] the Akaike information criterion (AIC) is proposed,

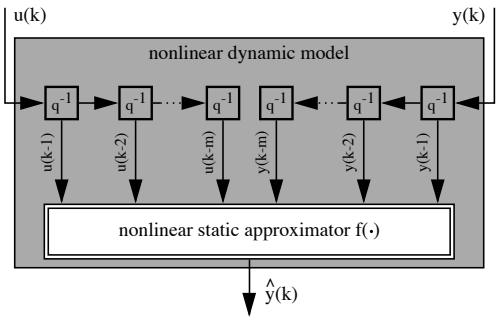

Fig. 1. External dynamics approach: nonlinear static approximator and external filter bank

but the root-mean squared error (RMSE) on the training or a validation data set can also be used. Embedded methods utilize model- or training-algorithm-specific properties to find good net input subsets, but do not have to rely on the achieved model performance. Model-free schemes are also called filter methods [4] and are often based on correlation, similarity or distance measures between the input and output variables [6].

Here we are going to compare four model-free and one model-based methods for dynamic order determination purposes. The following four mode-free approaches are considered: The Lipschitz Quotient, the False Nearest Neighbor (FNN), The Gamma Test and the Delta Test [7], [8], [9], [10]. A brief overview is given in Sect. II-B. For a more detailed summarization and thorough discussion of these approaches readers are referred to [11]. The model-based approach is proposed in [12] and is a mixture of a wrapper and an embedded method. It is able to exploit one special property of LMNs, i.e. the distinction between two types of (net) input variables: (i) scheduling or operating-point variables and (ii) local model or submodel variables. In a fuzzy system context type (i) variables describe the rule premise space and type (ii) variables describe the rule consequents space [1]. Note that both types of variables can be dynamic in nature but typically scheduling or operating variables are chosen statically or with very simple dynamics. This allows for much easier modeling and strongly weakens the curse of dimensionality [13]. Clearly, for these more advanced strategies a model-free approach is not sufficient.

II. Model Order Determination Methods

In contrast to a static model, a dynamic discrete-time NARX model depends on time delayed inputs and outputs. In this paper the focus lies on Single-Input Single-Output (SISO) systems. So let's assume some nonlinear dynamic function:

$$y(k) = f(\underline{\varphi}(k)) \,, \tag{1}$$

$$\begin{aligned} \underline{\varphi}(k) = [&y(k-n_1), y(k-n_2), \ldots, y(k-n_{no}), \ldots \\ &u(k-m_1), u(k-m_2), \ldots, u(k-m_{ni})] \,. \end{aligned} \tag{2}$$

Here $y(k)$ denotes the physical output and $u(k)$ is the physical input of the nonlinear process at time instant k. The symbols n_{no} and m_{ni} represent the possible maximum number of delays and therefore order for output and input respectively. The vector $\underline{\varphi}$ contains all delayed inputs and outputs or net inputs, respectively. An extension to Multiple-Input Multiple-Output (MIMO) systems is straightforward.

Order determination not only means the selection of n_i and n_o but also the selection of the net inputs. In the following some aspects are discussed, that are related to all order determination methods presented in this paper. With an increasing number of possible net inputs d, the number of potential subsets increases to $2^d - 1$. Due to the high computational costs, it is prohibitive to test all possible net input subsets. Therefore a lot of suboptimal search strategies have been developed, that try to find a reasonably good subset even if the best subset is not found [4]. The search procedure has a higher level strategy compared to the subset evaluation (Fig. 2).

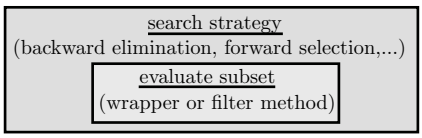

Fig. 2. Schematic representation of the linkage between search strategy and subset evaluation

Besides the argument of being feasible, the suboptimal search strategies are less prone to overfitting, especially when dealing with small sample sizes as mentioned in [6].

For all investigations in this paper, we used a backward elimination (also known as sequential pruning), that is slightly adjusted in order to ensure physically more reasonable results.

Backward elimination (BE) starts with all possible net inputs and prunes one of them in each iteration. The pruned net input is the one, that improves some evaluation criterion the most or decreases it the least. Usually the selection of the net input to be pruned considers all entities left in $\underline{\varphi}$. Through our slight adjustment, the BE is restricted in the choice of net inputs that can be pruned. It is only allowed to choose the least or most delayed version of each physical quantity contained in $\underline{\varphi}$. For example, if the minimum delay for a specific input is $u(k-1)$ and the maximum delay is $u(k-4)$, the pruning of $u(k-2)$ and $u(k-3)$ is prohibited. In that way we assure consecutive delays for each physical quantity contained in $\underline{\varphi}$ at any time. Of course each physical input or output is treated independently, such that there might be other minimum and/or maximum delays for each physical quantity. It results in less flexibility, but therefore should favor physically more reasonable results and less possibilities to overfit. We will call this search strategy *consecutive* BE.

A. Wrapper Method

For the model-based dynamic order determination local model networks (LMNs) [14] are used. As already mentioned in Sect. I, LMNs possess the possibility to distinguish between operating-point variables and variables describing the behavior of the local models. If local *linear* models are used, this distinction allows to divide all (net) inputs into linearly and nonlinearly influencing variables [14], [15]. In a fuzzy interpretation the rule premise space (IF part) is spanned by the nonlinearly influencing variables, the rule consequents space (THEN part) is spanned by the linearly influencing variables. The variables contained in the rule premises are denoted by $\underline{z}$, all variables in the rule consequents are denoted by $\underline{x}$. Therefore the variables in $\underline{z}$ define an operating point (or region) for which a (local) linear model is valid. The division of the $\underline{z}$-input space in several regions/operating points is called partitioning. Net inputs $\underline{\varphi}$ can now be assigned to the $\underline{x}$- and/or $\underline{z}$-input space. The output of the LMN can be calculated

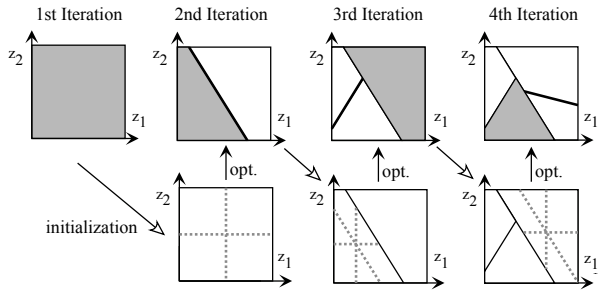

Fig. 3. Four iterations of the HILOMOT algorithm for a 2-dimensional $\underline{z}$-input space

as sum of M local models $\hat{y}_i$, weighted with their validity function Φ_i:

$$\hat{y} = \sum_{i=1}^{M} \hat{y}_i(\underline{x})\Phi_i(\underline{z}) \text{ with} \quad (3)$$
$$\underline{x} \subseteq \underline{\varphi} \text{ and } \underline{z} \subseteq \underline{\varphi}\,.$$

For the generation of LMNs the hierarchical local model tree (HILOMOT) algorithm [16] is used. It is able to cope with the separation between the $\underline{x}$- and $\underline{z}$-input space. Additionally, it is able to partition the $\underline{z}$-input space in an axes-oblique way, which makes this modeling approach very suitable for high-dimensional input spaces. All local models are chosen to be of affine type in this paper. The validity functions are generated by sigmoid splitting functions that are linked in a hierarchical, multiplicative way, see [16] for more details. The procedure of the HILOMOT algorithm can be explained with the help of Fig. 3. It is an incremental algorithm constructing a tree of local affine models. Starting with a global affine model, in each iteration an additional local affine model is generated. The local model with the worst local error measure (gray areas in Fig. 3) is split into two submodels, such that the spatial resolution is adjusted in an adaptive way. The linear parameters of the new submodels are estimated locally by a weighted least squares method. This is computationally extremely cheap and introduces a regularization effect which increases the robustness against overfitting, as stated in [1]. The axes-oblique partitioning is achieved by optimizing the current split direction and position in each iteration. Only the new split is optimized, all already existing splits are kept unchanged. The initial split direction for the optimization is either one of the orthogonal splits or the direction of the parent split (dotted lines in Fig. 3).

Figure 4 illustrates the procedure for the model order determination. A search strategy determines which of the net inputs contained in $\underline{\varphi}$ should be used as variables in the $\underline{x}$- and $\underline{z}$-input space. Then HILOMOT is used to train the LMN and its performance is evaluated by the model's simulation error on validation data. Note that the model complexity is determined with the help of a corrected version of Akaike's information criterion (AICc) [17] and not on the validation data. Therefore the validation data is only used once for each net input subset (and not several times for the model

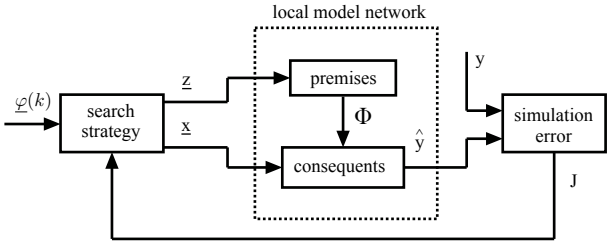

Fig. 4. Flow chart of the LMN based order determination method

complexity determination). The used search strategy is the consecutive BE as explained in Sect. II. Because of the possibility to assign each net input contained in $\underline{\varphi}$ to any of the two input spaces $\underline{x}$ and $\underline{z}$, different selection strategies can be defined. The so-called linked $\underline{x}$-$\underline{z}$ selection arises, if the net inputs in $\underline{x}$ and $\underline{z}$ are linked ($\underline{x} = \underline{z}$). In fact, such a selection strategy does not exploit the separability of the $\underline{x}$- and $\underline{z}$-input space and therefore can be pursued using any model structure. The separated $\underline{x}$-$\underline{z}$ selection exploits the separability-property fully, i.e. arbitrary subsets of the net inputs can be assigned to the $\underline{x}$-input space independently from the net inputs assigned to the $\underline{z}$-input space. And of course selections can be defined, where only one of the two input spaces is investigated while the inputs in the other input space are held fixed ($\underline{x}$ selection and $\underline{z}$ selection).

B. Filter Methods

While the wrapper methods depend on a (nonlinear) model of the process, filter methods are model-free. Without any model structure, only geometrical properties of the evaluated data set are available. The reviewed filter methods are based on the relationship between distances in the output to distances in the input space. To illustrate the differences, the methods are visualized in the Distance Space [11]. For this illustration, the Hammerstein system shown in Fig. 5 is used.

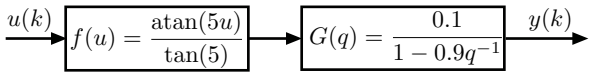

Fig. 5. Structure of Hammerstein system

The process is excited by an amplitude modulated pseudo random binary signal (APRBS) [1] to cover the whole input space. The resulting input and output signal is shown in Fig. 6.

1) Lipschitz Quotient: For a Lipschitz bounded function the following expression holds:

$$L \geq \frac{\left|y(\underline{\varphi}_i) - y(\underline{\varphi}_j)\right|}{\left|\underline{\varphi}_i - \underline{\varphi}_j\right|} = q_{i,j}\,. \quad (4)$$

Because of (4) the value of $q_{i,j}$ lies between zero and L. So the largest values of $q_{i,j}$ are close to L. The value L represents the maximum derivative of the sampled function. With this information an index is defined based on the p largest $q_{i,j}$,

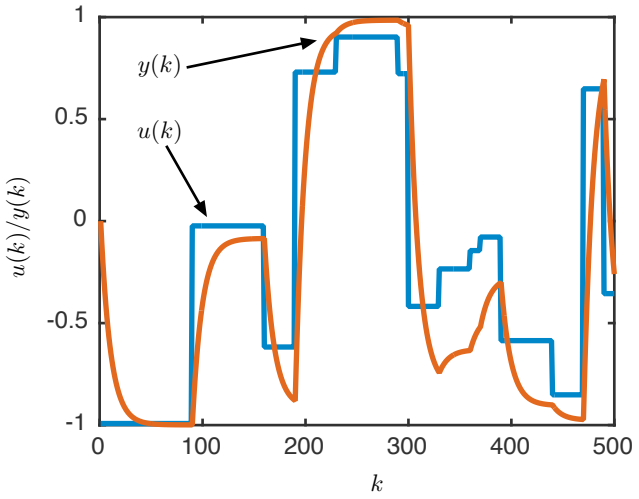

Fig. 6. Input and output signal of the nonlinear process with $N = 500$ samples

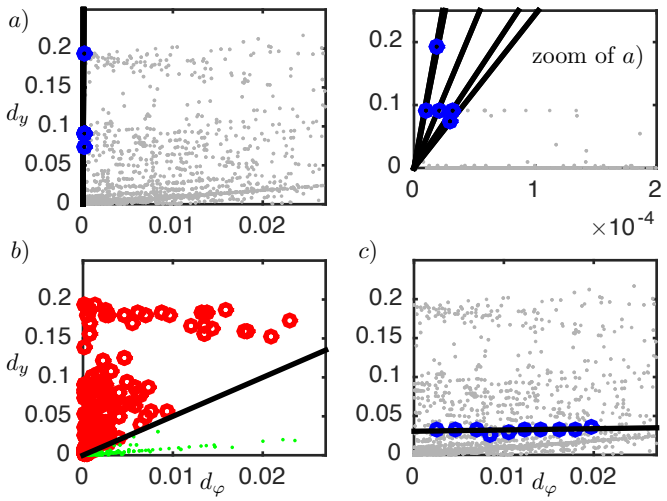

Fig. 7. Distance Space for an incorrect input space with a missing input. The subset consists only of the input $y(k-1)$.

which is an estimation of the bounding value L. With $p \approx 0.01N$ and n being the number of inputs contained in $\underline{\varphi}$.

$$q(\underline{\varphi}) = \sqrt{n}\left(\prod_{l=1}^{p} q_{i,j}(l)\right)^{\frac{1}{p}} \tag{5}$$

High values for $q(\underline{\varphi})$ are unusual, so this indicates a derivative which violates the bounding value L. These huge $q(\underline{\varphi})$ only occur at small changes in $\underline{\varphi}$ while the corresponding output y shows a relatively big discrepancy. This leads to the assumption that an important input of the system is missing.

Figure 7a and 8a demonstrate the Lipschitz Quotient in the Distance Space. The Lipschitz Quotient is represented by the points, which form the biggest slopes in conjunction with the origin of the coordinate system. For a subset with a missing input, the Lipschitz quotient $q(\underline{\varphi})$ is very big (Fig. 7a), compared to the correct input space (Fig. 8a).

2) False Nearest Neighbor: This approach classifies data point-pairs in true or false nearest neighbors (TNN/ FNN). The classification is done by the distance $d(\underline{\varphi}_i)$ of an input data point to its nearest neighbor and the corresponding output distance $d(y(\underline{\varphi}_i))$. The classification threshold R is chosen by the user and depends on the investigated problem.

$$\frac{d(\underline{\varphi}_i)}{d(y(\underline{\varphi}_i))} \leq R \tag{6}$$

If the expression in (6) is true the neighbors are true neighbors, otherwise they are false. The explanation is similar to the Lipschitz Quotient: If a reaction of the output has no corresponding action in the input, the fraction of the distances (in (6)) is much bigger than R. So this point-pair is a false nearest neighbor.

The classification of FNN is demonstrated in the Distance Space in Fig. 7b and Fig. 8b. The wrong subset contains many false nearest neighbors (red circles in Fig. 7b) while the correct subset in Fig. 8b has only true nearest neighbors (green dots).

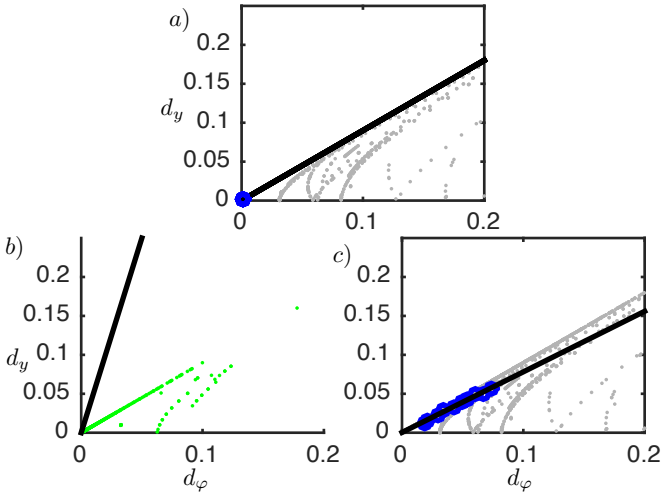

Fig. 8. Distance Space for the correct input space. The subset consist the necessary inputs $u(k-1)$ and $y(k-1)$.

3) Gamma Test: The Gamma Test uses the k-nearest-neighbors (kNN) in the input space to estimate the variance of the process, with $1 < k < p$ and $p \approx 10$. For each k the averaged squared distance between $\underline{\varphi}_i$ and its k-th NN $\underline{\varphi}_{i,k\text{NN}}$ is calculated:

$$\delta(k) = \frac{1}{N}\sum_{i=1}^{N}\left(\underline{\varphi}_i - \underline{\varphi}_{i,k\text{NN}}\right)^2. \tag{7}$$

For each distance between $\underline{\varphi}_i$ and $\underline{\varphi}_{i,k\text{NN}}$, the squared output distance is calculated and averaged to get $\gamma(k)$ (note that $y(\underline{\varphi}_{i,k\text{NN}})$ and $y(\underline{\varphi}_i)$ are not necessarily kNN in the output space, although $\underline{\varphi}_i$ and $\underline{\varphi}_{i,k\text{NN}}$ are kNN in the input space).

$$\gamma(k) = \frac{1}{2\cdot N}\sum_{i=1}^{N}\left(y(\underline{\varphi}_i) - y(\underline{\varphi}_{i,k\text{NN}})\right)^2 \tag{8}$$

The values of $\underline{\gamma}$ and $\underline{\delta}$ are used to fit a regression line (On the one hand, the vertical intercept of this line represents the variance of the process, on the other hand, the intercept can be interpreted as the squared distance of the $0th$ Nearest Neighbor, which is a good measure for the variance.).

While in this paper it is not the task to determine the variance of the process, the Gamma Test can be used to evaluate the subset. This is possible, because too small subsets act like the true subset with high variance (see Fig. 7 and Fig. 8). The scaled regression line, obtained by the δ and γ values are visualized in Fig. 7c and Fig. 8c. The estimated variance for the correct subset is close to zero (Fig. 8c), while the variance for the wrong subset is significantly higher (Fig. 7c).

4) Delta Test: The Delta Test (a simplification of the Gamma Test [9]) uses in contrast to the Gamma Test just the first-Nearest-Neighbor (NN) for this approach. Similar to FNN the distance in the output, is calculated by the nearest neighbor in the input space. In contrast to FNN the aggregated distance of the output is used to evaluate the chosen subset. Note that δ is the same value like $\gamma(1)$ out of the Gamma Test (8):

$$\delta = \frac{1}{2 \cdot N} \sum_{i=1}^{N} \left(y(\underline{\varphi}_i) - y(\underline{\varphi}_{i,\mathrm{NN}}) \right)^2 . \tag{9}$$

Similar to the Gamma Test, the estimated variance of the Delta Test is much smaller for the correct subset (Fig. 8c), compared to the wrong subset (Fig. 7c).

III. Processes

To test the model-based and model-free dynamic order determination methods explained in Sect. II, four synthetic processes are used. These are taken from [1] and incorporate a Hammerstein system, a Wiener system, a nonlinear differential equation and a dynamic nonlinearity, that is not separable into static and dynamic blocks. They cover different types of nonlinear behavior, such that strengths and weaknesses of the methods should be revealed. The sampling time for all systems is assumed to be $T_0 = 1$ s. The nonlinear difference equations are given below together with the recommended intervals of the input signals.

- Hammerstein system (HS):

$$\begin{aligned} y(k) =& 0.01876 \arctan[u(k-1)] + \cdots \\ & 0.01746 \arctan[u(k-2)] + \cdots \\ & 1.7826 y(k-1) - 0.8187 y(k-2) \ . \end{aligned} \tag{10}$$

  Input interval $u \in [-3, 3]$.
- Wiener system (WS):

$$\begin{aligned} y(k) = & \arctan[0.01867 u(k-1) + \cdots \\ & 0.01746 u(k-2) + 1.7826 \tan(y(k-1)) - \cdots \\ & 0.8187 \tan(y(k-2))] \ . \end{aligned} \tag{11}$$

  Input interval $u \in [-3, 3]$.
- nonlinear differential equation (NDE):

$$\begin{aligned} y(k) = & -0.07289[u(k-1) - 0.2 y^2(k-1)] + \cdots \\ & 0.09394[u(k-2) - 0.2 y^2(k-2)] + \cdots \\ & 1.68364 y(k-1) - 0.70469 y(k-2) \ . \end{aligned} \tag{12}$$

  Input interval $u \in [-1, 1]$.
- Dynamic nonlinearity (DN):

$$\begin{aligned} y(k) = & 0.133 u(k-1) - 0.0667 u(k-2) + \cdots \\ & 1.5 y(k-1) - 0.7 y(k-2) + \cdots \\ & u(k)[0.1 y(k-1) - 0.2 y(k-2)] \ . \end{aligned} \tag{13}$$

  Input interval $u \in [-1.5, 0.5]$.

The input ranges are chosen such that a sufficiently strong nonlinear behavior is created.

We created two data sets, one for the order determination and one for the evaluation of the achieved model performances (test data). An APRBS with 2560 samples serves as input for all processes. The simulated process outputs are disturbed by white Gaussian noise with with two different signal-to-noise ratios, which are 50 dB (low noise) and 30 dB (moderate noise). The test data is not disturbed by any noise. Input signals for the creation of the data sets are shown in Figs. 9 and 10. Note that in Fig. 9 only the first half of the order determination data set is shown, such that the characteristics of the signal can be recognized more easily. Here the input signals are scaled between zero and one, but for the generation of the process outputs the input intervals mentioned above were used.

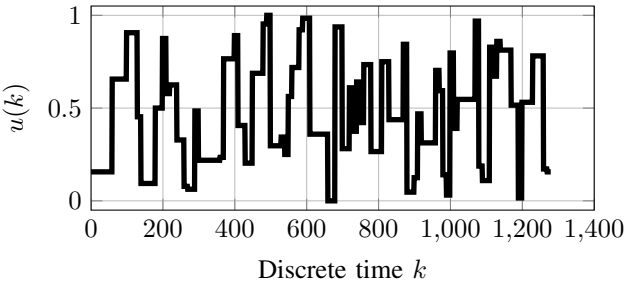

Fig. 9. First half of the order determination data set

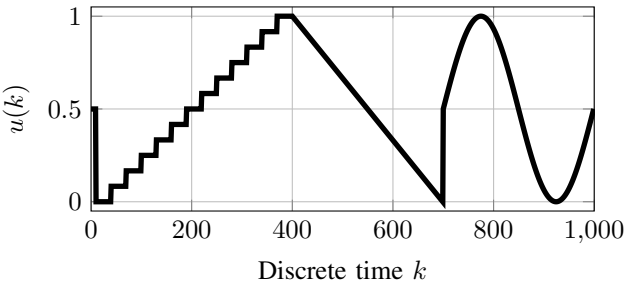

Fig. 10. Test input signal

IV. Results

All order determination methods are compared according to their performance on training and test data. Therefore LMNs with the net inputs chosen by the corresponding order determination strategy are trained. In addition the computation time required to obtain the results is compared. Furthermore we show the chosen net inputs selected by the different methods for two of the four test processes. First in Sect. IV-A and Sect. IV-B the selection of a specific net input subset for the model-based and the model-free methods are explained. Then in Sect. IV-C the results of all methods are compared.

A. Wrapper Results

To generate results that are comparable to the filter methods, the separability of LMNs is not exploited here. This means, that we used a linked $\underline{x}$-$\underline{z}$ selection as described in Sect. II-A. The model-based order determination, yields a curve of the evaluation criterion values over the number of net inputs. Here, we split the whole APRBS data set meant for the order determination into two halves. The first half (Fig. 9) serves as training data, that is exclusively used to determine all model parameters and to select an appropriate model complexity (i.e. the number of local models). The second half of the data set is used for the calculation of the evaluation criterion during the HILOMOT wrapper approach. It is named validation data set in the following, but only used for performance comparisons of different net input subsets, not for the determination of any model complexity. The RMSE value on this validation data quantifies the quality of a specific net input subset, and therefore guides the consecutive BE. For an automatic net input subset selection the one standard error (SE) rule according to [18] has been used. Therefore the SE is calculated for the model with the lowest RMSE value (RMSE_{min}) on validation data. Then the simplest model fulfilling the equation

$$\text{RMSE}_i \leq \text{RMSE}_{\text{min}} + \text{SE} \tag{14}$$

is chosen. Here *simplest* refers to the model with the least number of net inputs. RMSE_i denotes the achieved model performance using net input subset i. Because all curves of the RMSE values on validation data are similar for the investigated processes, only the progression of DN is shown in Fig. 11. The chosen number of net inputs is highlighted with a circle and the delayed inputs and outputs are mentioned directly above it. Typically, the evaluation criterion changes not significantly as long as the number of net inputs is not too low. The one SE rule helps in finding the necessary minimum of net inputs that should be used for the model generation.

B. Filter-Based Results

In contrast to the wrapper method, the evaluation criterion of the filter methods is no error value. Therefor the above-mentioned SE-rule cannot applied here. Rather the subset with the best rating is selected.

Figure 12 highlights the progression of the Delta Test approach for DN. By applying a backward elimination, the figure needs to be read from the right side and ends at the smallest subset on the left. Initially, the evaluation criterion is small due to the fact, that all necessary inputs are included. By repeatedly removing the worst input, the evaluation criterion changes only slightly until an important input is excluded. The large almost constant part of the curve makes it difficult to determine the best subset, since all evaluation criteria are comparable.

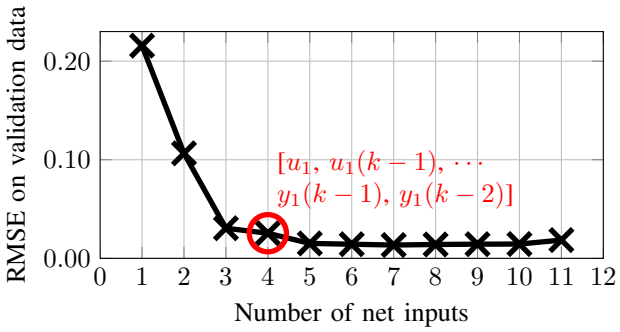

Fig. 11. RMSE on validation data vs. number of net inputs for the DN process

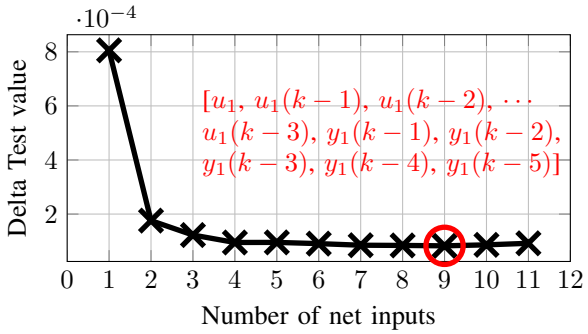

Fig. 12. Delta Test value vs. number of net inputs for the DN process and a logarithmic SNR of 50 dB

This progression of the evaluation criterion in Fig. 12 is typical for all filter methods and is almost independent of the chosen system. So the determination of the best subset with filter methods is a difficult task in general.

C. Comparison - Wrapper vs. Filter

In Fig. 13 the required computation time for the different methods is illustrated. Since the wrapper method is based on a nonlinear model training, the computational effort is considerably higher compared to all filter methods.

For a LMN the computational complexity depends on the number of estimated local models. For a low noise level, the dataset contains more information compared to a strongly disturbed dataset. Thus a training with less noise yields more complex models represented by more local models. In contrast to the wrapper method, the computational time for the filter methods are similar for both noise levels.

Fig. 13 illustrates a comparison between the computation time of the wrapper and the filter approaches. It represents

the low noise and thus worst-case scenario for the wrapper method.

The complexity of both methods can be analyzed by focusing on the evaluation of a single subset, because the BE procedure is identical for both approaches:

$$\text{HILOMOT complexity: } \mathcal{O}\left(2MSn^2N\right) \tag{15}$$

$$\text{Filter method complexity: } \mathcal{O}\left(N^2\right). \tag{16}$$

The complexity of the HILOMOT algorithm roughly depends on the number of samples N, the number of local models M, the number of net inputs n and the number of nonlinear split optimization iterations S. The complexity assessment considers the nested least squares estimation of the local affine models within the nonlinear split optimization.

The distance calculation of the filter methods depends only on the number of samples. It is quadratic because point-to-point distances are required. By increasing N, the discrepancy between wrapper and filter complexity decreases.

In order to compare the performance of the wrapper and the filter methods, the chosen subsets are used to train local model networks. The simulation RMSE on test data is visualized in Fig. 14 for a high noise level and Fig. 15 for a low noise level on the training data. For comparison reasons also the *true* subset, denoted as prior knowledge, was used to train a nonlinear model. The resulting test error is depicted together with the wrapper and filter methods.

Most of the filter methods are unreliable since their performances varies strongly. For example the Delta Test performs well for the DN but significantly worse on all remaining systems in the low noise case. The worse performance of the FNN approach is due to its sensitivity with respect to user-chosen threshold. Fine tuning would allow significant improvements. Surprisingly, the Lipschitz approach performs best many times. Note that the Gamma Test chooses a subset which led to an unstable model for WS, thus the error is not visualized in Fig. 14.

By comparing the two different noise levels in Fig. 15 and Fig. 14 the overall simulation test error increases with higher disturbances. While the Lipschitz approach performs best three times in the low noise case, the wrapper method is best three times for a higher noise level.

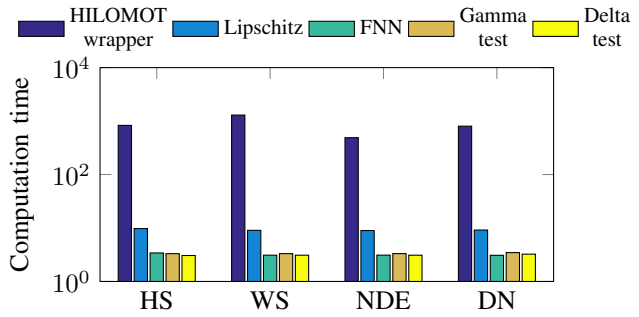

Fig. 13. Comparison of the required computation time of all order determination methods in seconds for logarithmic SNR of 50 dB

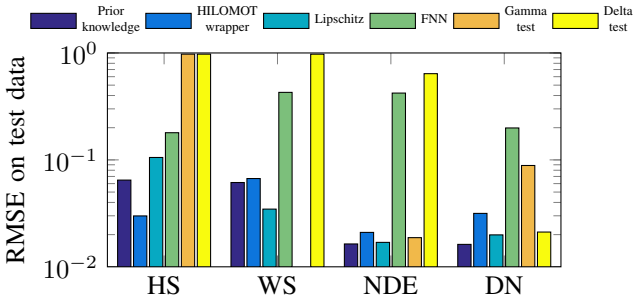

Fig. 14. Comparison of achieved model qualities on test data in case of a logarithmic SNR of 50 dB

It is noticeable, that the *true* subset often does not perform best. This is most likely because the best performing subsets contain more inputs than the true process (see Table I and Table II). So a more complex model with additional inputs is able to describe the process better, than to the *true* subset.

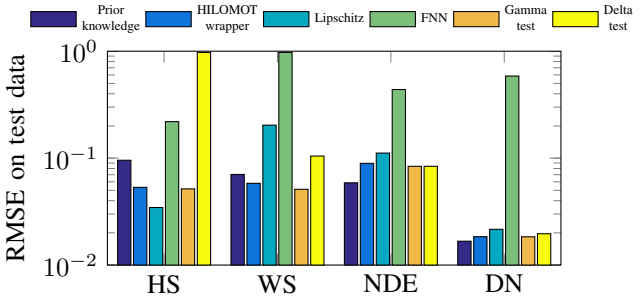

Fig. 15. Comparison of achieved model qualities on test data in case of a logarithmic SNR of 30 dB

While the overall performance varies strongly for different systems, only the (NDE) and the (WS) results for the low noise case will be considered in the following. For most approaches these two systems are the best and worst case.

In NARX configuration, the model quality is very sensitive with respect to the chosen inputs. A wrong model order and/or dead time could lead to bad model accuracy and even unstable models. For the proposed artificial nonlinear systems, it is possible to compare the chosen delays with the delays of the system difference equation. In Table I and Table II the correct net inputs are marked with a green cell color. The chosen delays of the approaches are denoted with different symbols. The poor performance of the FNN approach and the Delta test may be caused by the selection of delayed outputs only for both systems. Without any delayed input, the process can not be simulated by the model.

For the NDE system, the Lipschitz Quotient and Gamma Test select additional inputs besides the correct net inputs. The gained model complexity schieved through the Lipschitz Quotient and Gamma Test selection causes also a good representation of the nonlinear process. Thus the simulation test error is similar to the *true* subset.

TABLE I
CHOSEN NET INPUTS BY THE HILOMOT WRAPPER (♡), LIPSCHITZ (L), FNN (☹), GAMMA TEST (Γ) AND THE DELTA TEST (Δ) FOR THE NDE SYSTEM

	k	$k-1$	$k-2$	$k-3$	$k-4$	$k-5$
u	Γ	♡ L Γ	♡ L Γ	♡	♡	
y		L ☹ Γ Δ	♡ L ☹ Γ Δ	♡ L Γ	♡ L Γ	L Γ

TABLE II
CHOSEN NET INPUTS BY THE HILOMOT WRAPPER (♡), LIPSCHITZ (L), FNN (☹), GAMMA TEST (Γ) AND THE DELTA TEST (Δ) FOR THE WIENER SYSTEM

	k	$k-1$	$k-2$	$k-3$	$k-4$	$k-5$
u		L Γ	♡ L Γ	Γ		
y		♡ L ☹ Δ	♡ L ☹ Δ Γ	L Γ	L Γ	L Γ

Modeling the WS is a more challenging task. The wrapper method determines the correct subset with just a missing input delay. Similarly to the NDE system, the Lipschitz Quotient and Gamma Test determine a too complex model order. While the subset determined with the the Lipschitz Quotient performs best, the Gamma Test identifies a subset, which lead to an unstable model. This shows, that additional net inputs can harm or improve the model performance.

Most of the proposed filter methods are unreliable for an automatic model order determination. Only the Lipschitz quotient identifies acceptable subsets in low noise scenarios. For a realistic SNR of 30 dB the wrapper method outperforms the filter methods in most cases.

V. CONCLUSION AND OUTLOOK

Model order determination is an important task in nonlinear system identification, because wrong model order can influence the quality of the estimated model significantly. In this paper one model-based and four model-free approaches for model order determination were compared based on the achieved model qualities on four artificial test processes with two noise levels. By using a backward elimination the initial subsets were pruned until only one lagged net input remained and then the most appropriate subset was selected for modeling. Most of the filter methods are unreliable since their performance varies strongly. Most robust is the wrapper method which achieves good results in general.

One big disadvantage of the filter methods is the possibility of determination of unstable subsets, although the process is stable. This undesirable behavior is significantly less likely unlikely for the wrapper method.

A surprising result is the worse performance of the *true* subset compared to more complex subsets. The gained model quality of the *too* complex subsets is confusing and needs a thorough investigation in future work. Obviously, the additional net inputs were beneficial for the modeling task.

The subspaces $\underline{x}$ and $\underline{z}$ are linked in the presented wrapper method. Individual input spaces ($\underline{x} \neq \underline{z}$) can achieve improved results. The HS, NDE and DN can be separated in linear and nonlinear parts. For DN the inputs $u(k-1)$ and $u(k-2)$ are linear while the inputs $u(k)$, $y(k-1)$ and $y(k-2)$ are nonlinear. So a separation

$$\underline{x} = [u(k),\, u(k-1),\, u(k-2),\, y(k-1),\, y(k-2)] \quad (17)$$

$$\underline{z} = [u(k),\, y(k-1),\, y(k-2)] \quad (18)$$

would be promising. By applying a separated $\underline{x} - \underline{z}$ selection, the local models are functions of $\underline{x}$ while the input space partitioning is a function of $\underline{z}$. In future work, this will be investigated.

REFERENCES

[1] O. Nelles, *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer, 2000.

[2] H. Wei, S. Billings, and J. Liu, "Term and variable selection for non-linear system identification," *International Journal of Control*, vol. 77, no. 1, pp. 86–110, 2004.

[3] J. A. Solares and H.-L. Wei, "Nonlinear model structure detection and parameter estimation using a novel bagging method based on distance correlation metric," *Nonlinear Dynamics*, vol. 82, no. 1-2, pp. 201–215, 2015.

[4] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, "Feature extraction," *Foundations and applications*, 2006.

[5] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani, *The elements of statistical learning*. Springer, 2009, vol. 2, no. 1.

[6] H. Liu and H. Motoda, *Computational methods of feature selection*. Chapman & Hall, 2007.

[7] X. He and H. Asada, "A new method for identifying orders of input-output models for nonlinear dynamic systems," in *American Control Conference, 1993*. IEEE, 1993, pp. 2520–2523.

[8] C. Rhodes and M. Morari, "False-nearest-neighbors algorithm and noise-corrupted time series," *Physical Review E*, vol. 55, no. 5, p. 6162, 1997.

[9] A. J. Jones, "New tools in non-linear modelling and prediction," *Computational Management Science*, vol. 1, no. 2, pp. 109–149, 2004.

[10] F. Mateo and A. Lendasse, "A variable selection approach based on the delta test for extreme learning machine models," 2008.

[11] T. O. Heinz and O. Nelles, "Interpretation and analysis of input selection approaches in distance space," in *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 2015, pp. 367–374.

[12] J. Belz, D. Schwingshackl, J. Rehrl, O. Nelles, and M. Horn, "Order Determination and Input Selection with Local Model Networks," in *55th Conference on Descision and Control (CDC) [submitted]*, December 2016.

[13] J. Belz and O. Nelles, "Input selection using local model network trees," in *Proceedings of the 19th IFAC World Congress*, Capetown, South Africa, August 2014, pp. 4128–4133.

[14] R. Murray-Smith and T. Johansen, "Local learning in local model networks," in *Artificial Neural Networks, 1995., Fourth International Conference on*. IET, 1995, pp. 40–46.

[15] O. Nelles and R. Isermann, "Basis function networks for interpolation of local linear models," in *Decision and Control, 1996., Proceedings of the 35th IEEE Conference on*, vol. 1. IEEE, 1996, pp. 470–475.

[16] O. Nelles, "Axes-oblique partitioning strategies for local model networks," in *IEEE International Symposium on Intelligent Control*, 2006, pp. 2378–2383.

[17] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *International Symposium on Information Theory, 2 nd, Tsahkadsor, Armenian SSR*, 1973, pp. 267–281.

[18] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. Chapman & Hall/CRC, 1984.