Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Input Selection Using Local Model Network Trees

Julian Belz * **Oliver Nelles** *

* *University of Siegen, Department of Mechanical Engineering, D-57068 Siegen, Germany (e-mail: julian.belz@uni-siegen.de).*

Abstract: This paper presents an input selection wrapper approach using local model network trees. This model class allows the distinction in two input spaces - the rule premises input space and the rule consequents input space. Therefore the input selection can take place in both or just in one of these input spaces. As we will show, this leads to an improved model accuracy and an improved understanding of the dependencies between the inputs and the output. The introduced input selection algorithm is applied to one artificial data set and to the auto miles per gallon data set, see Frank and Asuncion [2010], to show the algorithm's abilities.

Keywords: Nonlinear System Identification; Neural Networks; Local Model Networks; Input Selection; Rule Premises; Rule Consequents.

1. INTRODUCTION

Mathematical descriptions of reality, respectively models, are very important in many areas, such as simulation, optimization, or feedback control. The number of potential input variables to achieve the modeling task typically is huge and a priori there is often no information which input variables are useful to model a certain kind of process. The term *useful* is chosen to make clear, that the best input variable subset regarding to model accuracy might not necessarily include all relevant input variables, according to Kohavi and John [1997]. Additional input variables usually cause a greater model flexibility and therefore increase the model's variance error. The most useful input variable subset corresponds to the best bias-variance trade-off, see Munson and Caruana [2009]. Besides the fact, that the selection of all input variables might not lead to the best bias-variance trade-off, one other fundamental reason for selecting subsets of input variables is the *curse of dimensionality*, as stated by Liu and Motoda [2008]. To mention just one problem related to the curse of dimensionality, the amount of necessary samples to cover the input space grows exponentially with the input dimensionality. This leads to more time consuming and therefore more expensive measurements. In summary, the reduction of the input dimensionality aims to

- improve the reliability and accuracy of the model,
- reduce the time for model construction and
- make the underlying process more concise and transparent.

The usage of local model networks for input selection tasks creates new possibilities, such as the separation between the rule premises and consequents. Although such a strategy can be taken for any local model network structure, the identification algorithm must be able to cope with this. LOLIMOT (LOcal LInear MOdel Tree) and HILOMOT (HIerarchical LOcal MOdel Tree) are prominent examples that are capable of this purpose, as stated by Nelles [2000] and Nelles [2006]. However, the popular algorithms based on product-space clustering according to Gustafson and Kessel [1978] as well as to Gath and Geva [1989] can only treat common input spaces. In the fuzzy interpretation the separation means that the rule premises (IF) can operate on (partly) other variables than the rule consequents (THEN). Therefore input selection can be performed on both input spaces. To investigate the two input spaces we use a wrapper approach based on the HILOMOT algorithm.

2. HIERARCHICAL LOCAL MODEL TREE

HILOMOT belongs to the the class of local model networks and is based on the ideas of *hinging hyperplane trees*, which are described by Breiman [1993], Ernst [1998] and Töpfer [2002]. As already mentioned, local model networks allow to distinguish between the input space of the validity functions $\Phi_i(\cdot)$ and the local models $\hat{y}_i(\cdot)$, where the index i corresponds to the i-th validity function and the i-th local model. The output $\hat{y}$ of a local model network can be calculated as the interpolation of M local model outputs:

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

with $\underline{x} = [x_1 \; x_2 \; \cdots \; x_{nx}]$ spanning the consequent input space and $\underline{z} = [z_1 \; z_2 \; \cdots \; z_{nz}]$ spanning the premise input space, refer to Nelles [2000]. The originally measured input variables can be assigned to the premise and/or consequent input space according to their nonlinear or linear influence on the model output (Nelles [2006]), see Fig. 1. The validity functions describe the regions where the local models are valid; they describe the contribution of each local model to the output. For a reasonable interpretation of local model networks it is mandatory that the validity functions form a *partition of unity*:

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

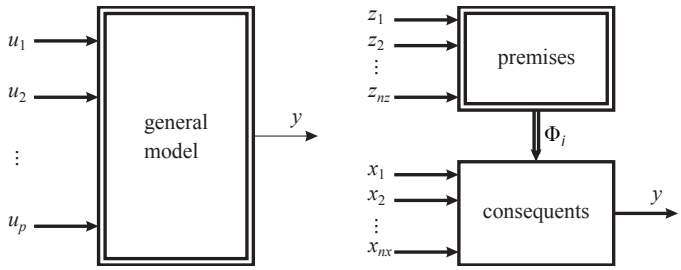

Fig. 1. For local model networks the inputs can be assigned to the premise and/or consequent input space.

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

HILOMOT uses sigmoid splitting functions, that are linked in a hierarchical, multiplicative way to determine the validity functions (Nelles [2006]). In every region defined by its validity function a local affine model is estimated. With the help of Fig. 2 the algorithm's procedure should be explained shortly. 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 is split into two submodels, such that the spatial resolution is adjusted in an adaptive way. The affine 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 by Nelles [2000]. A major advantage of HILOMOT is the possibility to perform axes-oblique splits, which makes this modeling approach very suitable for high-dimensional input spaces. Therefore the current split direction in each iteration is determined through a nonlinear optimization. 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.

One very important issue is the choice of the model complexity, which in case of HILOMOT is equivalent to the question: How many local models should be used for the final model? Especially for the input selection task a good bias-variance trade-off is mandatory. If the model is too complex overfitting arises, which means, that the model describes not only the process, but also the noise. In that case irrelevant inputs, that only contain noise, might appear important for the modeling of the process. If the model complexity is too low, the model is not able

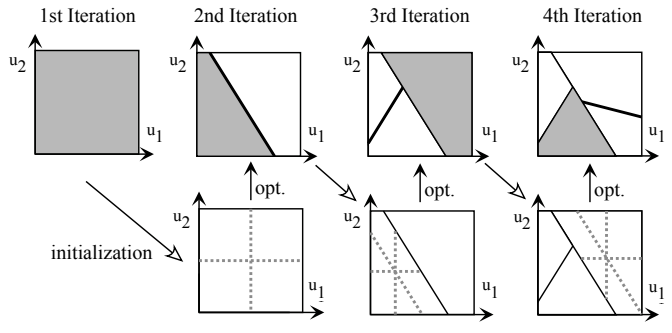

Fig. 2. Procedure of the HILOMOT algorithm in the first 4 iterations for a 2-dimensional input space ($p = 2$).

to describe the process sufficiently well and some inputs might be underrated. To find a good bias-variance trade-off the corrected Akaike Information Criterion (AIC_c) is utilized. The AIC_c estimates the expected, relative Kullback-Leibler information (which quantifies a "distance" between full reality and a model) based on the maximized log-likelihood function, see Burnham and Anderson [2004]:

$$\mathrm{AIC}_c = -2\ln\mathcal{L}(\hat{\underline{\theta}}|y) + 2n_{\mathrm{eff}} + \frac{2n_{\mathrm{eff}}(n_{\mathrm{eff}}+1)}{N - n_{\mathrm{eff}} - 1}\,. \qquad (3)$$

The number of data samples is denoted by N, whereas n_{eff} denotes the number of effective model parameters. The first term of equation 3 represents the log-likelihood function of the parameters $\hat{\underline{\theta}}$ given the process output y. In the assumed case of least squares estimation and normally distributed noise this term becomes, as stated by Burnham and Anderson [2004]:

$$\begin{aligned} -2\ln\mathcal{L}(\hat{\underline{\theta}}|y) &= N\ln\hat{\sigma}_n^2 \\ &= N\ln\left(\frac{1}{N}\sum_{i=1}^{N}(y(i) - \hat{y}(\underline{u}(i)))^2\right)\,, \end{aligned} \qquad (4)$$

with the measured process output $y(i)$ and the predicted model output $\hat{y}(i)$ at $\underline{u}(i)$ in the input space. More detailed information on Akaike's Information Criterion and its corrected version can be found in Burnham and Anderson [2002, 2004] as well as in Akaike [1973]. Additional information regarding the HILOMOT algorithm can be obtained from Nelles [2006].

3. INPUT SELECTION

As mentioned in the introduction, removing inputs may increase the reliability and accuracy of a model. This is possible, because some inputs might increase the model's variance error significantly (Munson and Caruana [2009]) and some inputs might be irrelevant or redundant (Liu and Motoda [2008]). Because even relevant inputs might decrease the predictive power of a model, as stated by Guyon and Elisseeff [2003], in the following the term *relevant* will be replaced by the term *useful*. Even if an input is useful in the sense of improving a model's accuracy, it has to be weighed up if the increase of predictive power overcomes the disadvantages of having more inputs, such as a higher computational demand and less interpretability.

To figure out which inputs are useful, three typical approaches exist: filter, wrapper and embedded methods, compare with Guyon [2006], Liu and Motoda [2008], Tan et al. [2006]. In embedded methods the generation of input subsets and their evaluation is incorporated in the training algorithm itself. Since the properties of embedded methods are very algorithm-specific, the following statements refer only to the filter and wrapper approaches. The main difference between filter and wrapper approaches is the evaluation criterion, according to Guyon [2006]. Filters use criteria not involving any learning machine. These criteria are often related to correlation estimations (Tan et al. [2006]) or similarity estimations (Guyon [2006]). In contrast wrappers use learning machines as a black box and wrap the input selection around that black box. So the evaluation criterion can be any criterion that is suited to measure the predictive power of a model, i.e.

cross-validation or as proposed in Karagiannopoulos et al. [2007] the correlation coefficient. Sindelar and Babuska [2004] suggest to use Akaike's information criterion (AIC), because it balances the model's error with the model's complexity. This is advantageous when comparing models with an unequal number of inputs and in order to find a good bias-variance trade-off.

An important issue of selecting useful inputs is the search strategy, that is used to explore the space of possible input combinations. An exhaustive search, where all possible input combinations are considered, takes usually a huge amount of time, since for p candidate inputs there are 2^p subsets to go through. Therefore a lot of *suboptimal* approaches have been developed, that try to find a reasonably good input subset even if the best subset is not achieved (Guyon [2006]). 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 Liu and Motoda [2008].

In our wrapper approach we use the HILOMOT algorithm with a corrected version of Akaike's information criterion (AIC_c) to determine useful input variable subsets. The AIC_c is used to ensure the validity of the criterion for low ratios of $\frac{N}{n_{eff}} < 40$ as recommended in Burnham and Anderson [2004]. Using the HILOMOT training algorithm is advantageous because there exist no crucial fiddle parameters and the algorithm is very robust. Furthermore the resulting model belongs to the class of local model networks, such that the input spaces for the rule premises and the rule consequents can be considered independently. Without specifying a special strategy to investigate the two input spaces, the procedure evaluating different input variable subsets is outlined in Fig. 3. In every iteration j the search strategy determines which of the original inputs $\underline{u}$ are included in the rule premises $\underline{z}$ and in the rule consequents $\underline{x}$. In the subsequent steps, a HILOMOT model is built and the output of the model is used to calculate the AIC_c as the criterion for the currently used combination of input variables. In this paper we use two search strategies, the so called *backward elimination* (BE) and an *exhaustive search* (ES). The backward elimination starts with all input variables. In each step of the search algorithm one of the variables is discarded from the input variable subset. Once an input is removed from the subset, it cannot be added afterwards. To decide which input variable should be removed at the current step, each of the remaining input variables is tentatively discarded from the input subset and a model is trained. The AIC_c value for each removed input variable is calculated and the input variable, that yields the best AIC_c value is picked for removal at the current step. After all input variables are removed, the model is empty and the search algorithm stops. The exhaustive search simply tries all possible combinations of inputs out and therefore is computationally very expensive. The latter approach is only feasible, when the number of potential inputs is relatively low. In the following, it is utilized to show the gap between the suboptimal simple backward elimination and what might be achieved through more sophisticated search strategies.
In this paper the input selection is carried out for two different input spaces. The first input space, we will call x-z-input space or for short just x-z-space. In this x-z-

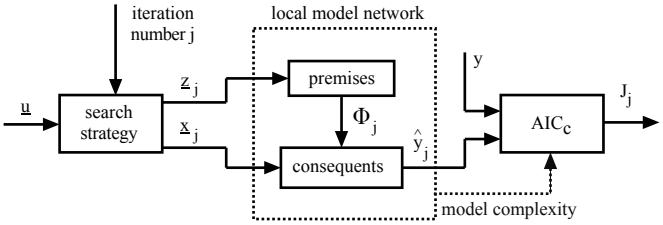

Fig. 3. Procedure to gain evaluation criteria for different input subsets. Every iteration j corresponds to an other input subset.

space the inputs, that are used in the rule premises are linked to the ones used for the rule consequents ($\underline{x} = \underline{z}$), see Fig. 4 (a). The second input space, we will call z-input space or for short z-space. Here all physical inputs are kept in the rule consequents ($\underline{x} = \underline{u}$), while only a subset of the physical inputs is considered in the rule premises, as outlined in Fig. 4 (b). The investigation of the

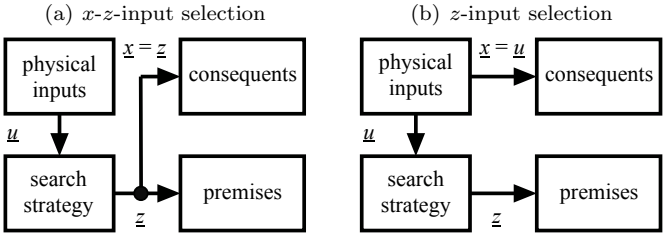

Fig. 4. Visualization of the x-z- and z-input selection. If the x-z-space is investigated, the two input spaces are linked, otherwise they are separated.

linked x- and z-space is possible with all kinds of modeling algorithms and from this point of view our approach can be seen as a competitor to existing wrapper approaches. In addition the HILOMOT algorithm extends the classical wrapper approach to investigations of the z-space, because it is able to cope with the separation between rule premises and consequents.

4. DEMONSTRATION EXAMPLES

In this section the input selection using HILOMOT will be tested on one artificial and one real-world data set. The artificial example is meant to demonstrate the abilities of the input selection with HILOMOT and should improve the understanding of what it means to deal with the two different input spaces, that are introduced at the end of Section 3. As real-world data set the auto miles per gallon (MPG) data set (Frank and Asuncion [2010]) is considered.

4.1 Artificial Example

The artificial example has three input variables (u_1 up to u_3), that contain information and one input variable (u_4), that only contains normally distributed noise with zero mean and a standard deviation of 0.05. For each useful input a separate function is defined and all functions are summed up. Variable u_4 is added to the sum of all functions:

$$y = \mathrm{f}(u_1) + \mathrm{f}(u_2) + \mathrm{f}(u_3) + u_4$$
$$y = \frac{0.2}{(0.2 + (1 - u_1))} + \mathrm{e}^{-\frac{(0.5-u_2)^2}{0.5^2}} + 0.8u_3 + u_4 \ . \qquad (5)$$

The first term in equation 5 is a hyperbola, that is nonlinear and monotonic. The second term contains a function, that is similar to a Gaussian function (nonlinear and non-monotonic), whereas the third term is just a linear function. All individual functions as well as the superposed noise are shown in Fig. 5 (a). To get a rough idea of the function's shape, the projection to the u_1-u_2 subspace is visualized in Fig. 5 (b). In the shown visualization the third and fourth input are set to zero. For this artificial example an input selection in the x-z-space and in the z-space is carried out as described at the end of Section 3. Because of the way, the artificial example is constructed, the usefulness of the inputs for the two different input spaces is known a priori. In the x-z-input space (rule consequents and rule premises linked), the first three inputs should be stated as useful, because these inputs contain information about the process behavior. In the z-input space (rule premises) only the first two inputs should be useful, because these inputs are the only ones with a nonlinear characteristic. The slope in the direction of input 3 is constant and, therefore, can be described by one linear model without further subdivision in the z-input space, as visualized in Fig. 6. To investigate the two input spaces, two different training data sets are utilized, that differ in the number of contained samples. In both cases the input samples are placed equidistantly on a grid. Five samples per axis are chosen for the smaller data set ($5^4 = 625$ samples) and nine samples per axis for the larger one ($9^4 = 6561$ samples). Because there are no interdependencies between the input variables, the backward elimination and the exhaustive search lead to identical results. To make sure, the specific realization of the added noise has no significant influence on the results, the investigations are performed with 100 different noise realizations. The variance due to the different noise realizations is very low, so only the result for one specific noise realization is shown in Fig. 7. The plot shows next to each point the input variable, that is discarded in the corresponding backward elimination step. In case of zero inputs, the model output is estimated to be the mean of all output values contained in the training data set. As can be seen in Fig. 7 (a) and (c) (left column) the results of the x-z-input investigation are qualitatively equal for the different sample sizes. The AIC_c values decrease until all inputs are chosen, that contain information about the artificial process. A similar behavior can be observed from the results of the z-input investigation, shown in Fig. 7 (b) and (d) (right column). After all input variables, that

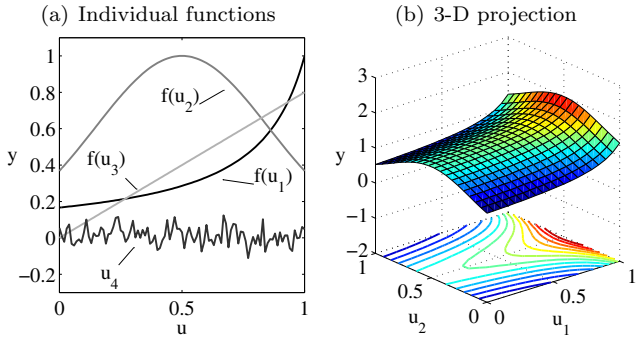

Fig. 5. Individual functions of the single inputs and a projection of the resulting function in the u_1-u_2-space.

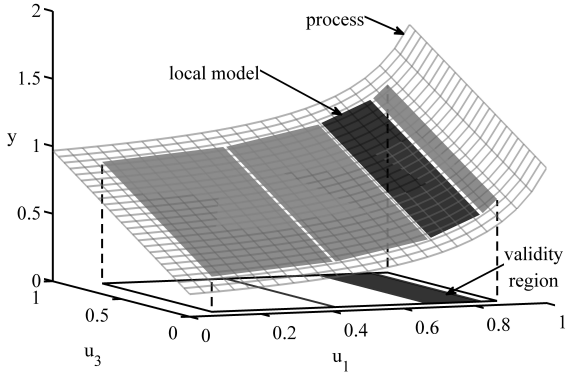

Fig. 6. Visualization of the process together with 4 local models and their validity regions in the u_1-u_3-space.

are useful in the z-input space are taken into account the AIC_c values stop decreasing as predicted and expected. Useless input variables in the z-input space are those, that only contain noise or influence the process output in an affine way, which can be described by just one affine model without further subdivisions of the z-input space. In case of 625 samples, the z-input investigation declares input 4 more useful than input 3, even though input 4 only contains noise. This is due to the small sample size and the superposed noise. In fact none of the two last input variables helps to increase the model's accuracy, if included in the z-input space. So it is more or less random, which of these two variables is stated as more useful. For the 100 different realizations of the noise, input 3 is stated as more useful in 63 cases. In Fig. 8 box plots of the required computation time are shown for the different search strategies and the different number of training data samples. At a first glance it is very surprising,

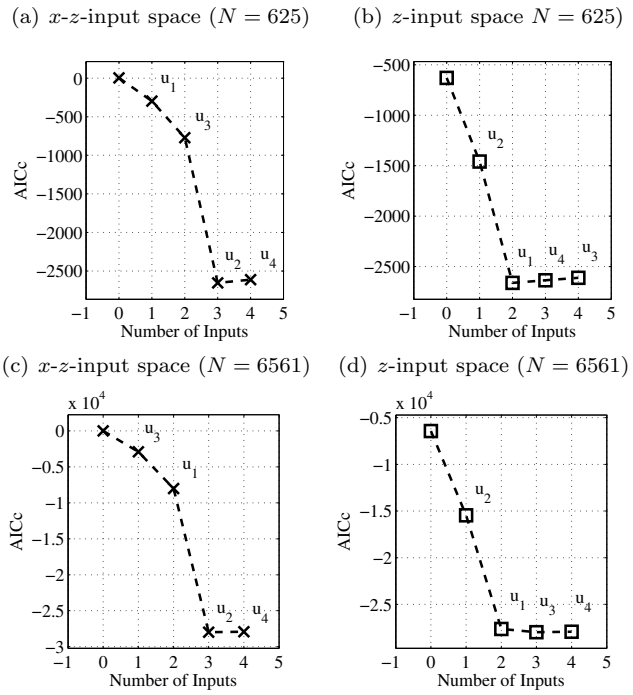

Fig. 7. Results of the x-z- and z-input investigation for the artificial data set and two different sample sizes N.

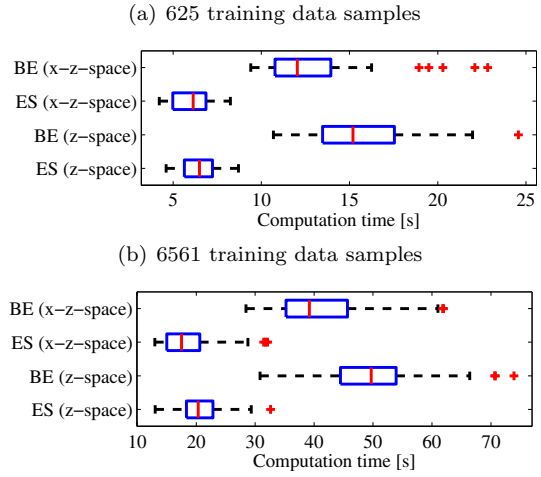

Fig. 8. Computation time for the different search strategies on a computer with eight CPUs operating at 2.4GHz.

that the exhaustive search requires less time than the backward elimination, because the computational effort of the exhaustive search is higher. But there is a simple explanation related to the algorithm's implementation and the possibility to calculate results for different input combinations in parallel. The calculations were performed on a computer, that has two quad core CPUs, such that eight different variable input combinations can be calculated at once. In case of the backward elimination the number of parallel calculations is restricted to the number of different input combinations left for a specific number of inputs. Let's say the algorithm is about to figure out, which input should be discarded in the second step and input u_4 is already discarded. There are only three possible input combinations left, such that only three calculations can be done in parallel. Before further calculations can be performed, the algorithm has to wait for the results of earlier iteration steps.

4.2 Real-World Example

In this subsection the HILOMOT wrapper approach is tested on the auto MPG data set (Frank and Asuncion [2010]). The data set consists of 392 samples and seven input variables, that will be named u_1 up to u_7 in the following. The information contained is the number of cylinders (u_1), the displacement (u_2), the horsepower (u_3), the car weight (u_4), the acceleration (u_5), the model year (u_6) and the origin (u_7). The output that should be predicted with the help of a model is the fuel consumption in miles per gallon (MPG). The data set is split into two groups. The first group (75% of the data samples) is used for the input selection. After the input selection is finished, models that are built with the best input combinations are tested on the second test data group. The test data is chosen in a deterministic way, with the goal to avoid extrapolation. At first an investigation of the x-z-space is carried out and the AIC_c curve together with the input selection path is shown in Fig. 9. The input selection path contains the information which input variables are taken into account, given the search strategy and the number of inputs. In case of the backward elimination, the input selection path has to be read from right to left. The results for the x-z-input investigation are very similar for both

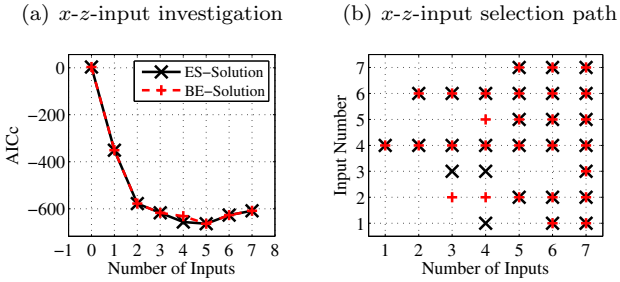

Fig. 9. Results of the x-z-input investigation of the auto MPG data set together with the corresponding input selection path (ES=x; BE=+).

search strategies and differ only for three and for four input variables. Both strategies have their optimum at five input variables, but the variable subset containing four input variables (ES-result) is very close to the best result. The mean squared test error (MSE) values are calculated for the subset with five variables, with four variables (the result from the ES-solution) as well as for all variables and are shown in Table 1. As can be seen, the input

Table 1. Test MSE for chosen input subsets in the x-z-space.

	ES (4 inputs)	ES & BE (5 inputs)	All inputs
MSE	5.978	5.977	7.679

combination containing five inputs yields the best result, closely followed by the subset with four input variables. The MSE value on test data is obviously worse, if all inputs are used. So the input selection with linked x- and z-inputs reveals a way to improve the model's accuracy.

For the following z-input investigation all physical inputs are included in the rule consequents. The results for the two search strategies are very similar, as can be seen in Fig. 10. The exhaustive search strategy as well as the backward elimination reach the minimum AIC_c in case of four input variables, but with different input subsets. Table 2 shows the results of the test MSE values. The best subset of inputs is determined by the exhaustive search strategy, but the backward elimination is very close. Both input

Table 2. Test MSE for chosen input subsets in the z-space.

	BE (4 inputs)	ES (4 inputs)	All inputs
MSE	5.266	5.171	7.679

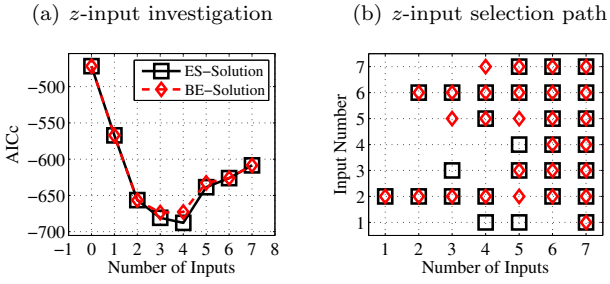

Fig. 10. Results of the z-input investigation of the auto MPG data set together with the corresponding input selection path (ES=square; BE=diamond).

subsets lead to a better model compared to the model with all inputs. In comparison with the results from the x-z-input space investigation (see Table 1), the results from the z-input investigation lead to better results. Together the results of both investigations can bring more insight in the interdependencies of some input variables with the output variable. In the input selection paths of both investigations the model year (u_6) is included in almost every subset of input combinations. So this input is very important for the modeling of the process, which sounds very reasonable as there are technical improvements over time, that lead to more efficient engines. In the z-input investigation the displacement (u_2) turns out to be very important, whereas in the case where the x- and z-inputs are linked, the car weight (u_4) is considered to be more useful. This can be interpreted as follows. The car weight is very important to model the fuel consumption, but in a more or less linear way. The influence of the displacement on the fuel consumption is not as important as the car weight, but has a more nonlinear characteristic. This insight can be gained through the distinction of the rule premises and the rule consequents and might be useful for further measurements or investigations of the process in general.

Finally, we will take a closer look at the computation time for the two search strategies, shown in Table 3. As expected, the exhaustive search requires more time than the backward elimination. The z-input investigation is more time consuming than the x-z-input investigation due to the fact, that the estimation of the local models is computationally more demanding. For every local model there are always seven parameters to estimate, whereas in case of the x-z-input investigation this number of parameters is less or equal to seven. With a growing input dimensionality the difference between the computation time of the exhaustive search and the backward elimination becomes greater due to the fact, that the number of possible combinations grow exponentially with the number of inputs. This exponential growing makes the exhaustive search infeasible for higher input dimensionalities. Nevertheless for a small number of inputs the exhaustive search is a realistic possibility in combination with the fast HILOMOT algorithm as the real world data set with seven inputs demonstrates.

Table 3. Required computation time for the auto MPG data set input selection.

	x-z-selection	z-selection
Backward elimination	24.66 s	25.73 s
Exhaustive search	65.65 s	71.06 s

5. CONCLUSIONS

This paper proposed a new advanced input selection framework that is based on local model networks. It distinguishes between linear (consequent space) and nonlinear (premise space) effects which leads to improved performance and better interpretability. The results motivate for further investigations regarding the influence of discrete inputs in the context of the presented input spaces.

REFERENCES

H. Akaike. Information theory and an extension of the maximum likelihood principle. In *International Symposium on Information Theory, 2 nd, Tsahkadsor, Armenian SSR*, pages 267–281, 1973.

L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *Information Theory, IEEE Transactions on*, 39(3):999–1013, 1993.

K.P. Burnham and D.R. Anderson. *Model selection and multi-model inference: a practical information-theoretic approach*. Springer, 2002.

K.P. Burnham and D.R. Anderson. Multimodel inference understanding AIC and BIC in model selection. *Sociological methods & research*, 33(2):261–304, 2004.

S. Ernst. Hinging hyperplane trees for approximation and identification. In *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, volume 2, pages 1266–1271. IEEE, 1998.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Isak Gath and Amir B. Geva. Unsupervised optimal fuzzy clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(7):773–780, 1989.

Donald E Gustafson and William C Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, volume 17, pages 761–766. IEEE, 1978.

I. Guyon. *Feature extraction: foundations and applications*, volume 207. Springer Verlag, 2006.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003. ISSN 1532-4435. URL `http://dl.acm.org/`.

M. Karagiannopoulos, D. Anyfantis, SB Kotsiantis, and PE Pintelas. Feature selection for regression problems. *Proceedings of HERCMA07*, 2007.

Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273 – 324, 1997. ISSN 0004-3702. doi: 10.1016/S0004-3702(97)00043-X. URL `http://www.sciencedirect.com/`.

H. Liu and H. Motoda. *Computational methods of feature selection*. Chapman & Hall, 2008.

M. Munson and R. Caruana. On feature selection, bias-variance, and bagging. *Machine Learning and Knowledge Discovery in Databases*, pages 144–159, 2009.

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

O. Nelles. Axes-oblique partitioning strategies for local model networks. In *International Symposium on Intelligent Control, 2006 IEEE*, pages 2378–2383. IEEE, 2006.

R. Sindelar and R. Babuska. Input selection for nonlinear regression models. *Fuzzy Systems, IEEE Transactions on*, 12(5):688–696, 2004.

P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

S. Töpfer. Approximation nichtlinearer Prozesse mit Hinging Hyperplane Baummodellen (Approximation of nonlinear processes with hinging hyperplane trees). *at-Automatisierungstechnik*, 50(4/2002):147, 2002.