Process State Observation Using Artificial Neural Networks and Symbolic Regression

Susanne Fischer

Intelligent Systems Research Group
Karlsruhe University of Applied Sciences
Karlsruhe, Germany
Email: susanne.fischer@hs-karlsruhe.de

***Abstract*—Process state observation is important for efficient automated online control in manufacturing. In this paper, we propose a new concept for online observation of the process state. First, we obtain state variables of physically-based numerical simulations of a process. After that, we reduce the state variables to a few characteristic features using artificial neural networks. As a result, we have a process state representation in a lower dimensional feature space. Using this representation, we apply symbolic regression to find a mathematical model that describes the state in the feature space. By applying these methods to a cup deep drawing process, we can describe 99% of the state variables' variation using only 7 features instead of 400. For these 7 features, we can find mathematical descriptions that represent a reduced process model which is used for process state observation.**

***Keywords*–Observation; control; reduction; regression; manufacturing.**

I. Introduction

Process control needs information about the current process state to adjust controllable process parameters. However, if the process state is not directly - or only with large effort - available during the process, a state observer is required to gain and observe process state information from available measurements (observables). For this reason, a process model is necessary to predict the immeasurable process state.

Given that the process model is used for process observation, it is evident that the process model has to be online-capable. For this purpose, black-box models or gray-box models can be applied. Black-box models characterize the process input-output relation. In addition to the input-output relation, gray-box models determine the resulting model structure.

One of the most used methods for implementing a reduced model is the proper orthogonal decomposition (POD), independently proposed by Kosambi [1], Karhunen [2], and Loève [3]. It is also known by Karhunen-Loève theorem (KLT) and principal component analysis (PCA). Rozza [4] covers a wide range of applications in engineering. Using POD, the reduced model $u_N(x,t)$ is represented as a linear combination of basis functions $\phi_j(x)$:

$$u_N(x,t) = \sum_{j=1}^{N} a_j(t)\phi_j(x), \qquad (1)$$

where $a_j(t)$ are time-dependent model coefficients and N is the reduced dimension. It is possible to apply POD to nonlinear problems, but the result leads only to a linear reduction [5].

Established methods for black-box modeling are Kriging and artificial neural networks (ANN). Kriging is based on the interpolation of the simulation results [6]. ANNs are trained to emulate the system by mapping the process input to the process output [7].

We intend to predict the process state and process dynamic, automated and online-capable. When implementing an online-capable model, one concern is the complexity of the process. Physical-based numerical process simulations predict the process state very accurately, but they are also computationally expensive and hence not feasible for online control. However, the resulting process state of the numerical simulation can be used to create a reduced process model.

We obtain process state variables for each time step and for different process parameters and use artificial neural networks to apply a nonlinear dimensionality reduction to reduce the obtained state variables to a few new state features. To achieve a gray-box model, we apply symbolic regression to describe the time evolution of the state in the reduced space. The determined process model is considered for process state observation by running the process model in parallel to the process.

In Section II, we introduce the several components of the state observation (Figure 1), including the dimensionality reduction algorithm. The result of this algorithm is compared

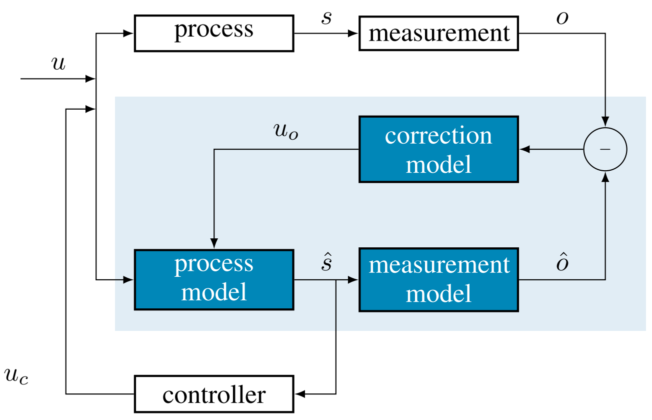

Figure 1. Components of the observer (blue) in parallel to the process.

to the results of the PCA and the nonlinear principal component analysis (NLPCA), introduced by [8]. The feasibility of this process state observation and dimensionality reduction is evaluated with data of a deep drawing finite element model in Section III and discussed in Section IV.

II. State observation

Standard observers for linear systems are the Luenberger observer [9] and the Kalman filter [10]. The concept for state observation proposed here is based on the Luenberger observer. We adopt the idea to run the observer in parallel to the process (Figure 1) and feed back the error between true observables o and predicted observables $\hat{o}$ to adjust the process model. An additional output is the predicted state $\hat{s}$ to feed the controller with the required information.

The observer components are: (1) the process model that predicts the state for the controller; (2) the measurement model that maps the predicted state to observables; and (3) the correction model. The inner construction of the individual components is explained in the following as well as the required dimensionality reduction.

A. Dimensionality Reduction

The goal of the dimensionality reduction is to reduce the number of state variables of physical-based numerical process simulations to a few characteristic features. The result is a representation of the state in feature space.

PCA is a widely used linear dimensionality reduction method. It starts from the premise that the maximal variance in the data corresponds to the highest information gain and hence small variances constitute negligible information or even noise. Thus, the feature space is described by the directions of the highest variances.

NLPCA, a nonlinear dimensionality reduction method, is based on an autoencoder — a bottleneck neural network where the input has the same meaning as the training target. Here, input and training target are the state variables. The activation of the bottleneck layer represents the extracted features. The NLPCA uses a hierarchical error function to achieve ordered features.

In [11], we described a nonlinear dimensionality reduction method based on artificial neural networks. This approach extracts ordered features that represents the process state in feature space. It has several individual bottleneck neural networks (BNN) that are arranged sequentially (seqBNN), with one bottleneck node each (Figure 2). The input X are the state variables that get reduced through the mapping layer to the bottleneck node. The activation of the bottleneck node represents the extracted feature. We have one network for each feature, so that we can examine each feature separately.

The training target of the first network are the state variables $Y_1 = X$. To construct the second feature, we keep the input and subtract the predicted output $\hat{Y}_1$ from the true target Y_1 and set it as the training target of the second network. Thus, the second feature is a reduction of the state variables in such a way that the feature reconstructs the variation of state variables that could not be defined by the first feature. This continues until we find n features that describe the state with only a negligible residual.

Thus, the network is trained to reduce the state variables to new features which in turn can reconstruct the state variables. The sum of the predicted output of each network corresponds to the reconstructed state variables. The state variables for the offline training are known through numerical simulation using ABAQUS (finite element analysis software).

B. Process Model

Symbolic regression [12][13], a field of genetic programming, can be used to identify process models. In contrast

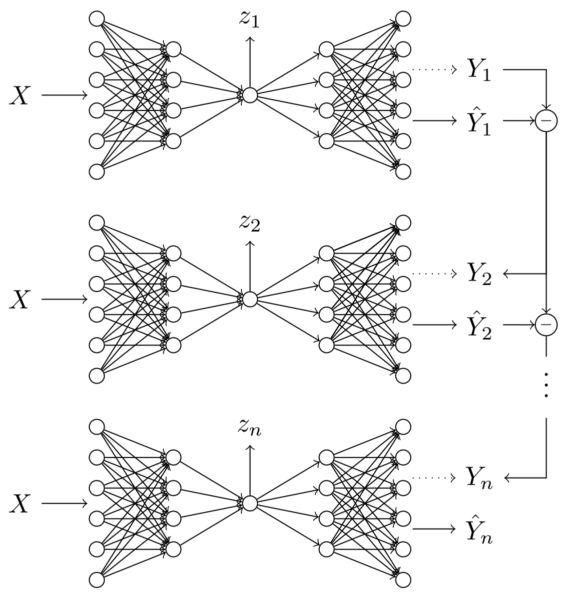

Figure 2. Architecture of the sequential BNN used for the dimensionality reduction.

to conventional regression methods, symbolic regression estimates not only the parameters, but, in addition, the structure of the required mathematical models. It has the advantage that expert knowledge about the structure of the model can be embedded and in turn the resulting model can be interpreted by experts.

Symbolic regression arranges the mathematical model in a binary tree (the so-called individuals) where constants, variables, and/or operators are children of operators (Figure 3). Using an entire population of individuals and genetic

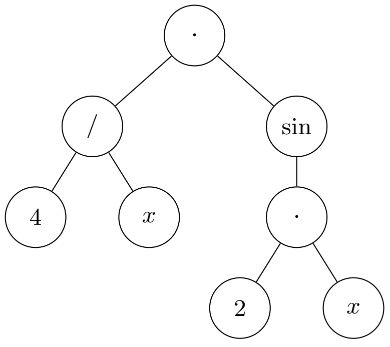

Figure 3. Symbolic regression individual for $\frac{4}{x} \cdot \sin(2x)$.

operations (such as mutation, crossover, and selection), the individuals are changed until an appropriate mathematical model is found that describes the given data points best.

In our case, we are trying to find a model that describes the trajectory of the state in the extracted feature space. The reduced state s depends on time t, the control parameter u_c, and noise ν: $\vec{s}(t, u_c, \nu)$. In this paper, we do not consider the influence of the noise. Consequently, the trajectory of the state $\vec{s}$ can be described by:

$$\vec{s}(t, u_c) = \sum_{i=1}^{n} z_i(t, u_c)\vec{e_i}, \tag{2}$$

where $z_i(t, u_c)$ is the i-th state feature depending on time t and process parameter u_c, $\vec{e}_i$ the unit vector, and n the dimension.

C. Measurement Model and Correction Model

The measurement model maps the extracted features to the predicted observables $\hat{o}$ (Figure 1). Subsequently, the predicted observables are compared to the true observables o.

Senn [14] shows that a three-layer artificial neural network (input, hidden, and output layer) is sufficient to predict state variables from observables with a negligible error. We assume that the inverse mapping, with the state variables as input and the observables as output of the neural network, provides similarly satisfying results.

The predicted reduced state variables of the process model and true observables, known from the numerical simulations, are used as the input and output, respectively. Thus, the numbers of input nodes and output nodes are predefined. To determine the number of nodes in the hidden layer, we use the following equation, based on [15]:

$$H = \frac{OS - O}{\alpha(I + O)}, \tag{3}$$

where I is the number of input neurons, O the number of output neurons, S the number of samples, and α a problem-dependent adjustable coefficient.

We applied a sigmoid activation function for the hidden layer to model the nonlinear relation between state and observables. The network is trained using the Levenberg-Marquardt backpropagation algorithm [16] to update the weights of the connection between the layers.

The input of the correction model is the difference between the true and predicted observables. The correction model converts this difference into information that the process model can use to update itself. Thereby, the observer can react to its own inaccuracies or noise.

III. Results

In this Section, we apply the presented method to a two-dimensional deep drawing model. The simulation of deep drawing is computed with ABAQUS. The dimensionality reduction is applied to the data that we extract from the simulations with different process parameters. We compare our dimensionality reduction method to the PCA, as a linear method, and NLPCA, as a nonlinear method. Using the results of the dimensionality reduction, we define a reduced process model using symbolic regression.

A. Application - Deep Drawing

As an example process, we used a two-dimensional finite element model of cup deep drawing. Deep drawing is a notable method regarding sheet forming. A metal sheet is clamped between a die and a blank holder (Figure 4). The force of the blank holder is adjustable. During the process the punch presses the metal sheet into an opening of the die.

The numerical simulation of the two-dimensional deep drawing model provides us with observables (such as displacements and reaction forces) and state variables (such as von Mises stress or strain). The simulation consists of 129 time steps and we perform 200 simulations, each with a different blank holder force. The blank holder force is time-invariant and varies between 70 and 100 kN. As a result, we have 9 observables and 400 state variables at 129 time steps for 200 different blank holder forces.

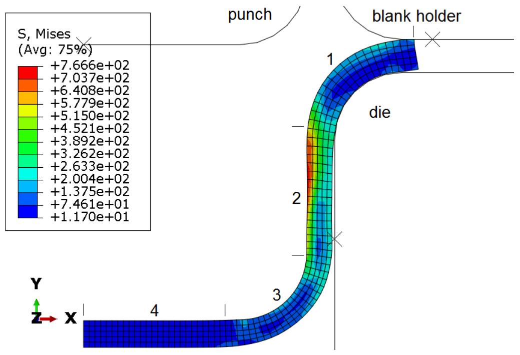

Figure 4. Two-dimensional deep drawing model: drawn metal sheet (v. Mises stress color-coded) clamped between blank holder and die.

We arrange this data in two matrices, one for the observables and one for the state variables. The size of each matrix X is described by $ST \times V$, where S is the number of samples (numbers of simulations with different blank holder force), T the number of time steps and V the number of observables or state variables, respectively.

We normalize the data of X using standard deviation:

$$X = \frac{X - \mathrm{mean}(X)}{\mathrm{std}(X)}, \tag{4}$$

where $\mathrm{mean}(X)$ is the arithmetic mean, and $\mathrm{std}(X)$ is the standard deviation.

B. Dimensionality Reduction

We reduced the state variables (von Mises stress) using our seqBNN approach in Section II-A. The reconstructed state variables are defined by

$$\hat{Y} = \hat{Y}_1 + \hat{Y}_2 + \ldots \hat{Y}_n. \tag{5}$$

These results are compared to the results of PCA and NLPCA. The reconstruction error, the mean square error (MSE) of the true state variables and the predicted state variables (both normalized by (4)), of the three methods is shown in Figure 5.

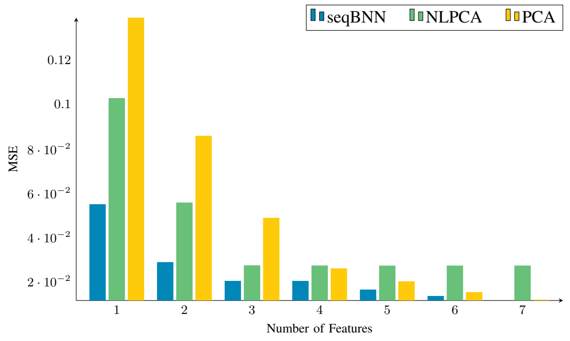

Figure 5. Comparison of reconstruction error (MSE) per feature of seqBNN, PCA and NLPCA.

The results of the methods based on artificial neural networks, seqBNN and NLPCA, depend on the configuration of the network. The network configuration for the seqBNN result in Figure 5 is:

- 8 nodes in mapping and demapping layer each
- 2000 training epochs

- 160 training samples with 129 time steps each
- 40 test samples with 129 time steps each

and for the NLPCA result:

- 16 nodes in mapping and demapping layer each
- 7 nodes in the bottleneck layer
- 2000 training epochs
- 160 training samples with 129 time steps each
- 40 test samples with 129 time steps each

For both methods different configurations, especially with regard to the number of nodes in mapping and demapping layer, yield different results for the higher features. Running different configurations has shown that there is no significant influence on the first two features, as long as the configuration is reasonable.

It is obvious that with increasing number of features the reconstruction error decreases. The error should converge toward zero as the number of features approaches the number of state variables. Furthermore, the diagram shows that feature 4 of the seqBNN could not explain relevant information that would reduce the reconstruction error significantly. The same is true for the feature 5, 6, and 7 of the NPLCA. By comparison, the seqBNN can reconstruct the data with only two features as good as the PCA with four features or the NLPCA with three features.

Figure 6 shows the coefficient of determination R^2:

$$R^2 = 1 - \frac{\text{SSE}}{\sum_{i=1}^{M} \left(Y_i - \text{mean}(Y)\right)^2}, \tag{6}$$

where SSE is the sum of the squared error between true and predicted values, and M the number of samples and time steps. R^2 measures how much of the total variation in the data is

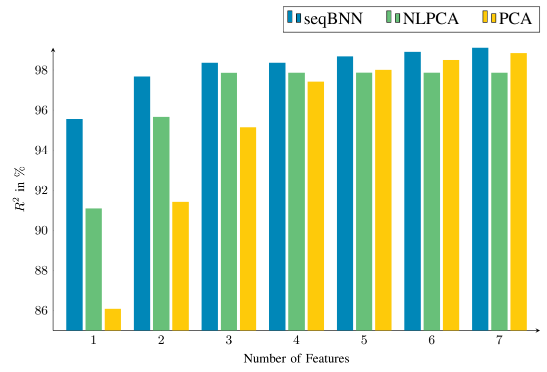

Figure 6. Comparison of R^2 per feature (in %) of the seqBNN, PCA and NLPCA.

described by the variation in the reconstructed state variables. With only one feature for seqBNN we can explain 95.55% of the true state variables variation (in the original domain - without preprocessing), and seven features can explain 99.11% of the example process.

Figure 7 shows the path of the reduced state within the feature space of only the first three features of the seqBNN with a MSE of 0.0201 and R^2 of 98.36%.

C. Process Model

Based on the extracted features, we intend to create a process model using the symbolic regression software Eureqa® [12]. The quality of the symbolic regression result is measured by the complexity of the mathematical model and the regression error. The complexity of each operation is defined by the user, depending on the prior knowledge, e. g., if it is almost certain that the result contains no exp-function, then the user can set the complexity of it to a higher number than the complexity of already known operations or exclude it entirely. Furthermore, it is common to set the complexity of basic arithmetic operations to a lesser value than, e. g., trigonometric operations. In our case, we assume no prior knowledge and run the symbolic regression with each operation, constant, and variable set to complexity 1.

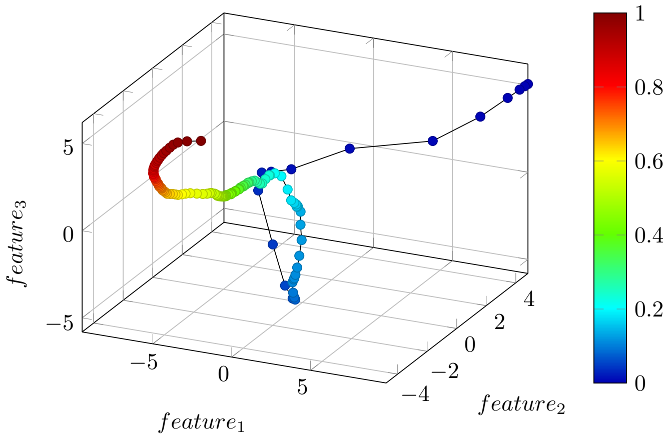

Figure 7. State in three-dimensional feature space. The time is color-coded.

For the following results the von Mises stress in each integration point of the finite element model is used as state variables. Figure 8 shows one result for the first feature (created in Section III-B with seqBNN). The resulting equation

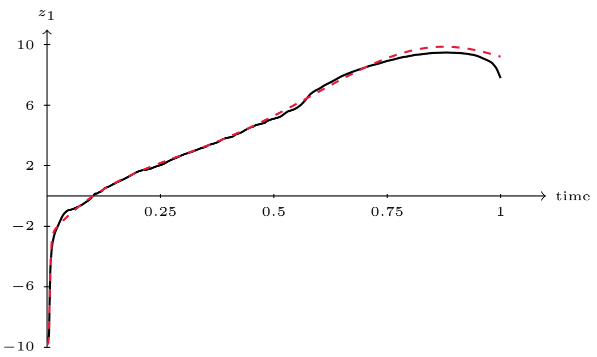

Figure 8. First feature over time (black line) of one specific process parameter u_c and $z_1(t, u_c)$ (red dotted line) using symbolic regression.

with a complexity of 37 and MSE of 0.0327 is:

$$\begin{aligned} z_1(t, u_c) = &-54.29t \cdot 8.60^t + 1.91 \cdot (8.60^t)^2 \\ &- \frac{12.80}{1.82 + (1823.35 \cdot 10^8)t^5} \\ &+ 257.45t^3 + 81.93t - 4.69. \end{aligned} \tag{7}$$

An other result with a lesser complexity of 21 but a higher error of 0.0979 is defined by

$$z_1(t, u_c) = -8.41 \cdot (4.87 \cdot 10^{-40})^{(132.16t^2)} - 3.68t^{10.53} + 14.04t - 1.48. \quad (8)$$

In these results, the process parameter u_c has no influence on z_1. We assume this is caused by the small range of the variation of u_c (see Section III-A). The obvious variation in the data is caused by the deformation over time. The linear-elastic part at the beginning of the process is obvious (Hooke's law) and so is the beginning of the plastic deformation.

If we consider the normal strain in x-direction in each integration point of the finite element model as state variables, then the extracted z_1 represents the highest variation in this quantity. Thus, we can assume that the derivative of z_1 shows similarities to the rate of change of the deformation. Hence, we are also looking for differential equations:

$$\frac{dz_i}{dt} = f(z_i(t, u_c), t, u_c). \quad (9)$$

One symbolic regression result for the derivative of the first feature of the strain values is explained by:

$$\frac{dz_1}{dt} = 7.53tz_1 + \frac{5.78 + 0.79z_1}{e^{t+z_1}} - 0.01te^{z_1}, \quad (10)$$

with a complexity of 24.

D. Measurement Model and Correction Model

The three-layer artificial neural network is trained with 140 samples and tested with 60 samples with 129 time steps each. We run 2000 training epochs with the extracted state features as input and observables as output.

We experimentally determined the coefficient α in (3), starting with a minimum of 1. Table I shows the MSE and R^2 for different numbers of nodes in the hidden layer. With 35 nodes in the hidden layer, we can explain 99.97% of the variation of the observables.

TABLE I. MSE AND R^2 OF THE TEST SET FOR DIFFERENT NUMBERS OF NODES IN THE HIDDEN LAYER.

	9	12	18	26	35
MSE	0.0064	0.0043	0.0014	0.0006	0.0004
R^2	0.9972	0.9973	0.9989	0.9996	0.9997

Afterwards, the predicted observables are compared to the true observables and the correction model updates the process model with new information. Note that building the process model, i. e., running the symbolic regression algorithm, is not online capable. Thus, for the presented online observation the correction of the process model is only possible using an additional summand.

IV. Conclusion and Future Work

In this paper, we introduced a new concept for observing state variables of a manufacturing process with the help of a process model, measurement model and correction model.

For this purpose, we reduced 400 state variables of physical-based process simulations to 7 new characteristic features that still describe 99% of the variation of the state variables. This nonlinear dimensionality reduction approach shows promising results for describing the process state in a low-dimensional space.

Before the network training, the NLPCA needs a fixed number of features that have to be extracted. This may result in too few or too many features in order to achieve a desired residual. With our method, however, it is sufficient to define the desired residual before the training and if that has not yet been reached, another feature will be extracted by an additionally network.

We assume that each feature represents a particular area (or part of it) of the metal sheet. This assumption is based on visual comparison of the extracted features with the von Mises stress curve for elements of a particular area, using the finite element model. For example, the stress curves of the elements in the area 1 in Figure 4 are similar to feature 1 and area 2 is close to feature 2. The proof of this assumption is subject of future work.

We proposed symbolic regression to find a process model in the extracted low-dimensional space. First results reveal that a function of time and process parameter can be found for each feature. We also showed first results for learning differential equations, but nevertheless expert interpretation is subject of future work.

The three-layer artificial neural network (measurement model) proved to be sufficient to map the reduced state variables to observables. The variation in the predicted observables describes 99% of the total variation of the true observables. Note that each step:

- dimensionality reduction
- learning of the process model
- regression: predicted state $\rightarrow$ predicted observables

introduces a minor error between predicted results and true results. It has to be ensured that the eliminated information is negligible or noise.

Future work will include the implementation of an optimal process chain controller using the presented process observation for each discrete process in the process chain.

Acknowledgements

The funding of this project within the Research Training Group 1483 "Process Chains in Manufacturing" by the German Research Foundation (DFG) is gratefully acknowledged.

References

[1] D. Kosambi, "Statistics in function space." Journal of Indian Mathematical Society, vol. 7, 1943, pp. 76–88.

[2] K. Karhunen, Über lineare Methoden in der Wahrscheinlichkeitsrechnung, ser. Annales Academiae scientiarum Fennicae: Mathematica - Physica. Universitat Helsinki, 1947, On linear methods in probability and statistics.

[3] M. Loève, "Fonctions alatoires de second ordre," in Processus stochastiques et mouvement Brownien, P. Lvy, Ed. Gauthier-Villars, 1948, Random functions of the second order.

[4] A. Quarteroni and G. Rozza, Eds., Reduced Order Methods for Modeling and Computational Reduction, ser. MS&A, Modeling, Simulation and Applications. Springer, 2013.

[5] G. Kerschen, J.-C. Golinval, A. F. Vakakis, and L. Bergman, "The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview," Nonlinear Dynamics, vol. 41, no. 1-3, 2005, pp. 147–169.

[6] M. Strano, "A technique for FEM optimization under reliability constraint of process variables in sheet metal forming," International Journal of Material Forming, vol. 1, no. 1, 2008, pp. 13–20.

[7] O. Nelles, Nonlinear System Identification, 1st ed. Springer-Verlag Berlin Heidelberg, 2001.

[8] M. Scholz and R. Vigário, "Nonlinear pca: a new hierarchical approach." in ESANN, 2002, pp. 439–444.

[9] D. G. Luenberger, “Observing the state of a linear system,” IEEE transactions on military electronics, vol. 8, no. 2, 1964, pp. 74–80.

[10] R. E. Kalman, “A new approach to linear filtering and prediction problems,” Journal of Fluids Engineering, vol. 82, no. 1, 1960, pp. 35–45.

[11] S. Fischer, O. Hensgen, M. Elshaabiny, and N. Link, “Generating low-dimensional nonlinear process representations by ordered features,” in 15th IFAC Symposium on Information Control in Manufacturing, 2015.

[12] M. Schmidt and H. Lipson, “Distilling free-form natural laws from experimental data,” Science, vol. 324, no. 5923, 2009, pp. 81–85.

[13] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, 1992.

[14] M. Senn and N. Link, “Hidden state observation for adaptive process controls,” in Proceedings of the Second International Conference on Adaptive and Self-adaptive Systems and Applications (ADAPTIVE 2010). IARIA, 2010, pp. 52–57.

[15] W. C. Carpenter and M. E. Hoffman, “Selecting the architecture of a class of back-propagation neural networks used as approximators,” Artificial Intelligence for Engineering, Design, Analysis and Manufacturing, vol. 11, 1 1997, pp. 33–44.

[16] M. Hagan and M. Menhaj, “Training feedforward networks with the marquardt algorithm,” IEEE Transactions on Neural Networks, vol. 5, no. 6, Nov 1994, pp. 989–993.