

# Knowledge Discovery in Engineering Dynamic System Analysis

Steffen Brückner<sup>\*</sup>, Stephan Rudolph  
Institute for Statics and Dynamics of Aerospace Structures,  
Universität Stuttgart, Germany

## ABSTRACT

Traditionally, the engineering modeling process is based on first principles, which usually yields large, complex and detailed models of a dynamic system. As an alternative, classical system identification procedures often produce simple (linear) models ignoring additional domain knowledge. In this context, numerical models, e.g. multi-body models or finite-element simulations based on first-principles are used to predict the system behavior. If only a small number of simulation outputs are needed, massive computational power is wasted on computing grid data which is of no further interest. In this paper, a different approach using engineering techniques, such as dimensional analysis, coupled with knowledge discovery methods, such as e.g. neural networks or  $k$ -nearest neighbor search, is used to predict the dynamic system behavior from only a few characteristic input parameters and the given initial or boundary conditions. The dynamic system is therefore modeled as a nonlinear static mapping whose parameters are estimated from experiment as well as from simulation. This static mapping allows for very fast prediction times compared to computationally intense numerical simulations. Additionally, some of the mapping methods allow the calculation of sensitivities, which in turn allow e.g. the ranking of the inputs according to their contribution to the output parameters. The presented approach to dynamic system analysis is first described in detail, then some of the used methods are described and the usefulness of the approach is demonstrated in the example of a non-linear spring-mass-damper system.

**Keywords:** Dimensional Analysis, Neural Networks,  $k$ -nearest-neighbor search, Modeling, Knowledge Discovery

## 1. INTRODUCTION

In dynamic system analysis, the simulation and analysis of nonlinear systems is a time and resource consuming process. In many applications detailed finite-element models with up to 1 million elements have to be built. The modification of these models is complicated and simulation times of up to one week restrict the usefulness of this approach in early design stages, where many systematic parameter variations of such simulations would be interesting to use. On the other hand, numerous simulations and experiments have already been done in these application domains.

A general dynamic system is depicted in figure 1. The current state of the system is given in the state vector  $x(t)$  which is usually not completely observable. The system is driven by the external input forces  $u(t)$  and given an initial condition vector  $x_0 = x(0)$  and the time-dependent input  $u(t)$  produces the output vector  $y(t)$ . For a linear time-invariant (LTI) dynamic system this behavior can be captured in a LTI state-space realization as shown in equation (1.1) where  $A$  is the system matrix which completely describes the internal (linear, time-invariant) dynamics of the system,  $B$  is the input matrix,  $C$  the output matrix and  $D$  the straight-way matrix.

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t) & ; & \quad x(0) = x_0 \\ y(t) &= Cx(t) + Du(t)\end{aligned}\tag{1.1}$$

---

<sup>\*</sup> pigroup@isd.uni-stuttgart.de, phone +49 711 685 3799, fax +49 711 685 3706, <http://www.isd.uni-stuttgart.de/pigroup/>, Universität Stuttgart, Institut für Statik und Dynamik der Luft- und Raumfahrtkonstruktionen, Pfaffenwaldring 27, 70569 Stuttgart, Germany

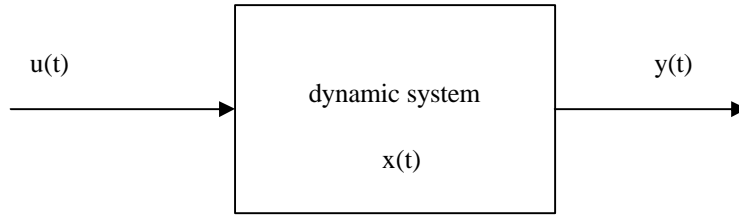


Fig. 1: Model of a dynamic system with inputs  $u(t)$ , states  $x(t)$ , and output  $y(t)$

Linear time-invariant dynamic systems (LTI systems) are well understood and a large number of analysis techniques in the time and the frequency domain exist<sup>2</sup>. The system behavior can be simulated using integration schemes, other techniques allow to characterization of these system using transfer functions or stability measures.

In the case of nonlinear, time-variant systems, the corresponding state-space description uses matrices with time- and space-dependent content as shown in equation (1.2). Numerical analysis techniques for these non-linear systems usually rely on an explicit integration scheme to simulate the system and generate the state trajectories, the output trajectory and their time derivatives.

$$\begin{aligned} \dot{x}(t) &= A(x, t)x + B(x, t)u \quad ; \quad x(0) = x_0 \\ y(t) &= C(x, t)x + D(x, t)u \end{aligned} \quad (1.2)$$

Using experiments and numerical simulations of a class of complex nonlinear systems, a database of these dynamic systems can be established as will be shown in the following section.

## 2. DATA COLLECTION

In this paper, the mapping from the description of a dynamic system and the initial conditions onto the dynamic response is modeled. In this context, the dynamic system itself, the initial conditions and the dynamic response are first modeled independently and the dynamic response is then estimated for unknown configurations or initial conditions using a nearest-neighbor search in known data sets.

The feature vector for a dynamic system can be given by the list of all relevant parameters as known from dimensional analysis<sup>3,4,5</sup>. The task of identifying the relevant parameters can at present time not be automated. However, techniques, such as correlation analysis, can help the engineer to form the relevance list, but profound domain knowledge is always required.

The features of the dynamic response of the system are given by a curve fit algorithm. Yet, the choice of base functions and fitting algorithms provide additional degrees of freedom for the choice of the feature vector. Domain independent curve fitting techniques, such as polynomial fits, splines or wavelet analysis can be employed but usually yield a high number of features that cannot be correlated to physical effects or a priori domain knowledge.

In this paper, another curve fitting algorithm is used. First, the general form of the functions is determined and a suitable base function for the acceleration, spanning the complete time range under consideration, is chosen. This base function is then integrated symbolically twice to yield the velocity and displacement functions. As much parameters as possible are then determined symbolically for these three symbolic functions using a priori domain knowledge, which is often given as invariant initial or boundary conditions or as heuristic knowledge. For each data set, these parameters are then determined from the data base and the remaining parameters are fit using conventional curve fitting algorithms, such as the least-squares method.

In many industrial applications simulations and experiments are conducted for a variety of reasons and most often, the data gained is used only once and then discarded. In the proposed system, the feature vectors for the dynamic system, initial conditions and the output can be determined for all experiments and simulations, and finally be stored in a database (see fig. 2). If retrieved from the database, this data can be used for the inference engine which is discussed in the following section.

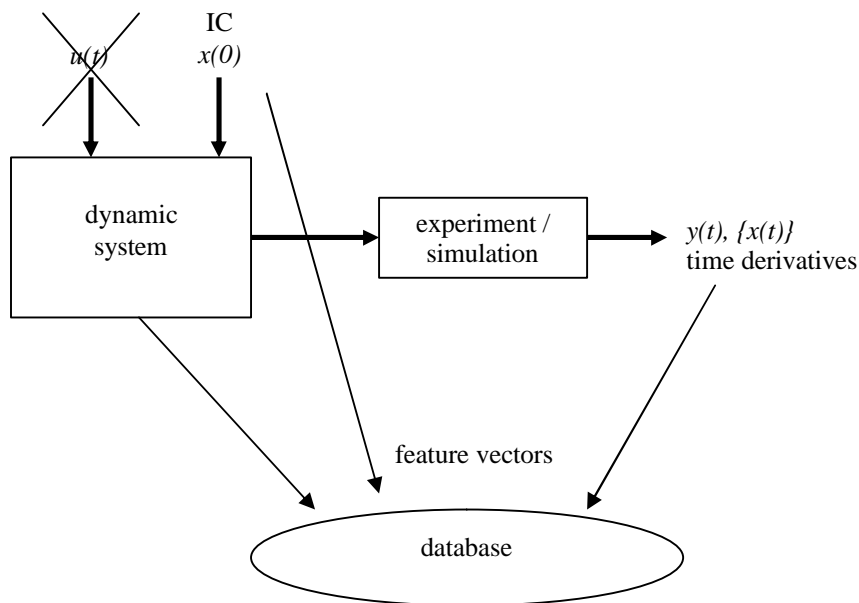


Fig. 2: The response of the dynamic system is given by experiments and simulations and the corresponding feature vectors are stored in a database. In the case shown, no external inputs are considered.

### 3. INFERENCE SYSTEM

To predict the system output for an unknown dynamic system with given initial conditions, an inference system as shown in figure 3 is used. This system relies on the information stored in the feature database for known experiments and simulations.

In this paper an inference engine based on the  $k$ -nearest neighbor classification algorithm<sup>1</sup> combined with a linear superposition of the classification results is shown. Similarly other mapping algorithms, such as neural networks, can be used to establish the prognosis of the dynamic system response from the system's feature vector.

The algorithm shown in this paper does not account for input signals  $u(t)$ , but these can easily be integrated in the system's feature vector using the same method shown for the characterization of the output signals.

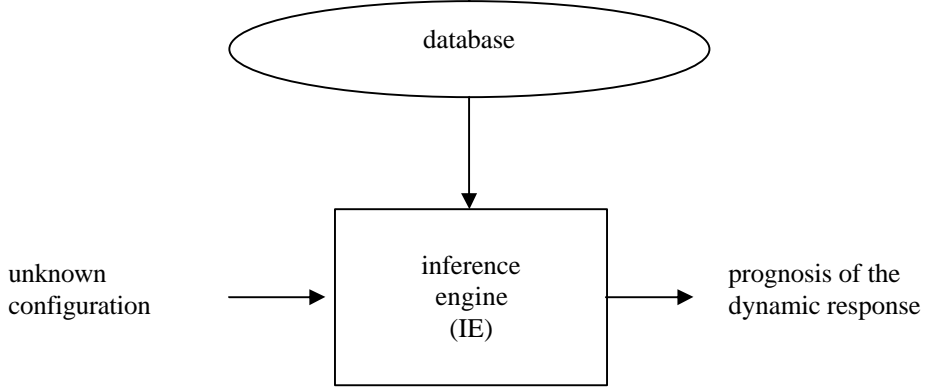


Fig. 3: Inference system for dynamic system behavior prognosis

The proposed inference engine is built using a  $k$  nearest neighbors search. For given input data  $Q$  (query) the input data is transformed into its feature vector representation. In the database the  $k$  nearest neighbors  $D_i$  ( $i=1, \dots, k$ ) in terms of these features are determined using a similarity measure and the distance from the query data  $d_i = \|Q - D_i\|$  is determined for each of these nearest neighbors using the same similarity metric. The stored output data  $R_i$  ( $i=1, \dots, k$ ) associated with the nearest neighbors  $D_i$  are then retrieved from the database and assembled according to the inverse distance  $1/d_i$  to the query.

In this paper, a special feature representation using dimensional analysis<sup>3,4,5</sup> is used for the dynamic system data  $D$  as well as the dynamic system output  $R$ . This method allows a physically inspired feature selection.

The similarity metric is twofold. Certain similarities are covered by the dimensionless representation of the features. Full physical similarity of the dynamic systems leads to identical dimensionless representations. These completely similar systems cannot be distinguished from each other in the dimensionless representation and are therefore undistinguishable for the inference engine. Another part of the similarity measure is used in the  $k$ -nearest neighbor search algorithm. Well-known similarity measures, such as the Euclidean distance, the City-block metric or Minkowski metrics can be used as well as weighted distance measures.

Euclidean distance	$d = \ D - Q\ _2 = \sqrt{(D - Q)(D - Q)^T} = \sqrt{\sum (D_i - Q_i)^2}$
City-block metric	$d = \sum  D_i - Q_i $
Minkowski metric	$d = \sqrt[p]{\sum  D_i - Q_i ^p}$

Tab. 1: Different similarity metrics for the  $k$  nearest neighbor algorithm

The estimated system output is composed of the features of the  $k$ -nearest neighbors as a linear superposition according to the inverse distance of their system features to the query system features. The linear superposition strictly only holds for linear systems, but using a dense population of systems in the database a linear approximation can be sufficiently accurate.

The properties of the  $k$ -nearest neighbor approximation are depicted in figure 4. On the left hand side the graph shows the shaded approximate area of interpolation and outside the area of extrapolation. The inference engine can easily detect extrapolation requests and produce a respective output. The right hand side of figure 4 shows the procedure of the  $k$ -nearest neighbor approximation algorithm. In this example, for a query “ $\oplus$ ” the  $k=4$  nearest neighbors are identified inside the dashed circle and the queried output is combined from the  $k=4$  nearest neighbors “X” according to their inverse distance to the query.

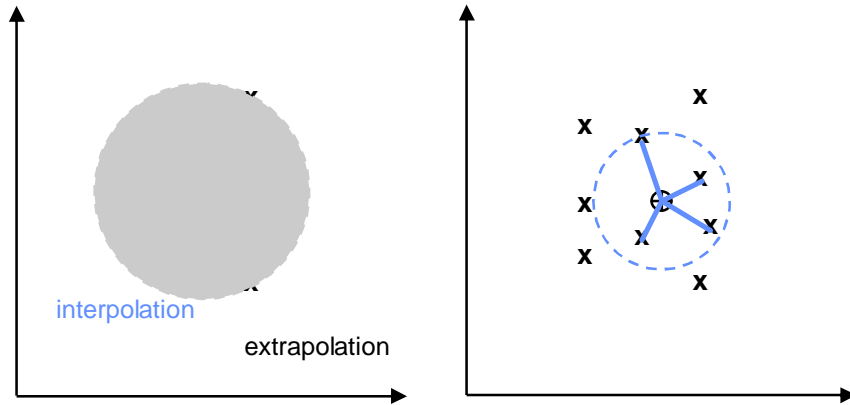


Fig. 4: Interpolation and extrapolation in database (left);  $k$ -nearest neighbor approximation ( $k=4$ ) (right)

#### 4. EXAMPLE

In the following, a simple one-dimensional pendulum (fig. 1) is considered. The equation of motion for this dynamic system is given by first principles and is shown in equation (4.1) along with the necessary initial conditions.

$$m\ddot{x}(t) + c\dot{x}(t) + kx = 0 ; x(0) = x_0 ; \dot{x}(0) = v_0 \quad (4.1)$$

This system can be transformed into its state-space representation as shown in equation (4.2). Since equation (4.1) is a second-order equation, two states and two initial conditions are necessary. It can be seen, that for this system, the output variables  $y$  correspond directly to the states  $x$ .

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -k/m & -c/m \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} ; x_0 = \begin{bmatrix} y_0 \\ v_0 \end{bmatrix} \\ \begin{bmatrix} y \\ \dot{y} \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \end{aligned} \quad (4.2)$$

Although this system is linear in its parameters mass  $m$ , damping coefficient  $c$ , and stiffness  $k$  it is nonlinear with respect to the initial conditions  $x_0 = [y(0) \ v(0)]^T$ .

The pendulum itself is characterized with the three parameters which can be taken directly from the equation of motion if these are known. Otherwise, the relevance list does not imply the knowledge of the underlying equations of motion.

mass	$m$	$[kg]$
damping coefficient	$c$	$[kg/s]$
spring stiffness	$k$	$[kg/s^2]$

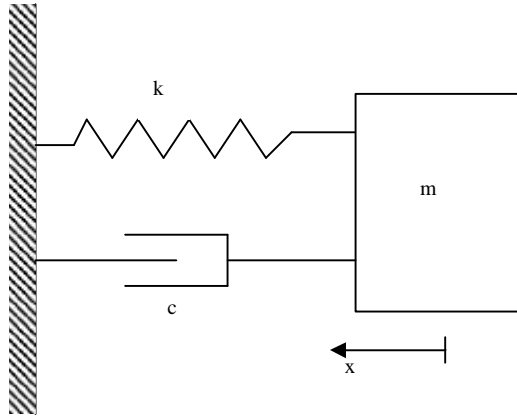


Fig. 5: Simple one-dimensional pendulum consisting of a mass  $m$ , a spring  $k$ , and a damper  $c$

Using dimensional analysis<sup>3,4,5</sup>, one single dimensionless group can be formed from these three parameters

$$s_{,1} = \frac{k m}{c^2} \quad (4.3)$$

For the data mining approach to dynamic system analysis, this is the important parameter of each experiment which has to be stored in the data base.

The dynamic response of the system is given in the time-series of  $x(t)$ ,  $v(t)$ ,  $a(t)$  with the initial conditions  $x(0)$  and  $v(0)$ . This forms the relevance list for the dynamic response

initial displacement	$x_0$	$[m]$
initial velocity	$v_0$	$[m/s]$
time	$t$	$[s]$
displacement	$x$	$[m]$
velocity	$v$	$[m/s]$
acceleration	$a$	$[m/s^2]$

and leads to four dimensionless groups for the dynamic response

$$\begin{aligned}
 A_{,1} &= \frac{v_0}{x_0} t \\
 A_{,2} &= \frac{x}{x_0} \\
 A_{,3} &= \frac{1}{x_0} v t \\
 A_{,4} &= \frac{1}{x_0} a t^2
 \end{aligned} \quad (4.4)$$

Again, these dimensionless parameters are stored in the database along with the initial conditions  $[x_0, v_0]$  to allow a reconstruction of the dimensional signal.

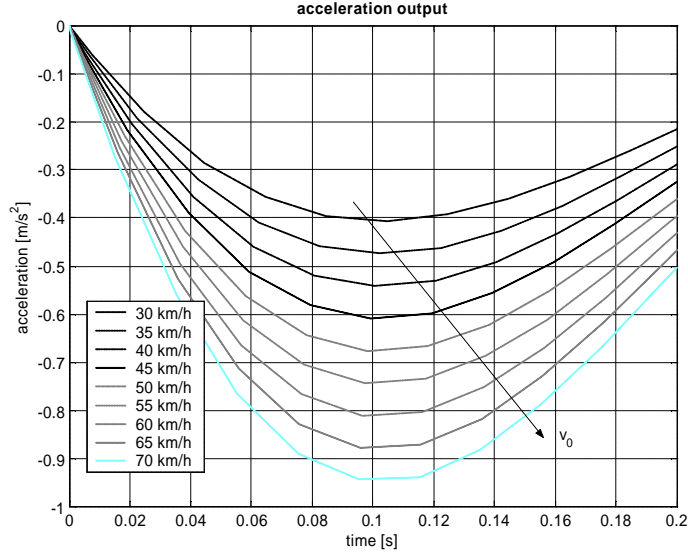


Fig. 6: Acceleration plots for different initial velocities. ( $m = 100\text{kg}$ ,  $c = 1000 \text{ kg/s}$ ,  $k = 15000 \text{ kg/s}^2$ )

Figure 7 shows the interpolation and extrapolation capabilities of the  $k$ -nearest neighbor based approximation algorithm. Despite the nonlinear dependence of the system output on the initial conditions, the mean-square error for the linear approximation is small. For the simulation in figure 7 initial velocities from  $30 \text{ km/h}$  up to  $70 \text{ km/h}$  as shown in figure 6 have been simulated. The data base known to the inference engine consisted only of the two system data points for  $v_0 = 40 \text{ km/h}$  and  $v_0 = 50 \text{ km/h}$ . It can be seen in figure 6, that the best performance, i.e. the lowest mean-square error, is achieved around the known data points with good performance in the interpolation between the known data and even slight extrapolation quality.

## 5. DISCUSSION

The prognosis of real-world non-linear dynamic system response is usually a very difficult task. The presented algorithm circumvents many difficult parameter choices in nonlinear models, at the expense to construct a useful “similarity measure” and the need for a sufficiently large data base of data gathered from simulation and experiment.

The  $k$ -nearest neighbor algorithm offers a transparent way of blending various different system responses into each other, thus constructing a prognosis of a system response.

The fact that the linear system prognosis of the nonlinear response gets better if the density of data points in the data-base increases is hereby a nice feature of the approach.

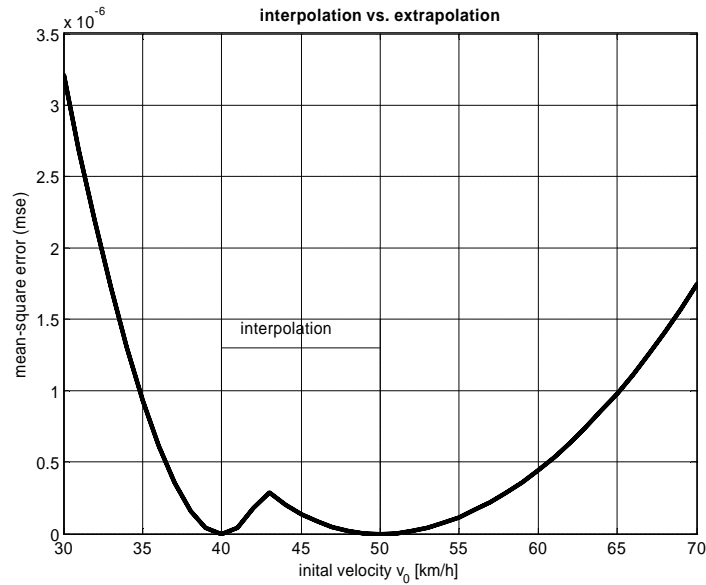


Fig. 7: Comparison of interpolation vs. extrapolation properties of the  $k$ -nearest neighbor approximation for the nonlinear dependence of the simple pendulum on the initial velocity  $v_0$ . Shown is the mean-square error in the acceleration signal of a simulation versus the prognosis of the  $k$ -nearest neighbor approximation with two data points at  $v_0 = 40 \text{ km/h}$  and  $v_0 = 50 \text{ km/h}$  for a velocity range of  $30 \text{ km/h} < v < 70 \text{ km/h}$ . The non-symmetric curve results from the nonlinear dependence of the system output on the initial velocity  $v_0$ .

## 6. SUMMARY

It has been shown that for a sufficiently dense database a  $k$ -nearest neighbor approximation can estimate the system output even for nonlinear system behavior. The data base can be filled with experimental as well as simulation data. The use of dimensional analysis on engineering system allows the definition of additional similarity measures which automatically account for physical similarity between different systems.

The advantage of the  $k$ -nearest neighbor approach over other data-driven methods, such as e.g. neural networks, lies in the additional output of the chosen nearest neighbors and their distance from the query point. The algorithm is capable of identifying new designs (i.e. extrapolation queries) and produces the according output. For new designs within the “known-world” the algorithm produces a list of the most similar designs known and predicts the output of the new design using known data.

## REFERENCES

1. M. Ester and J. Sander, *Knowledge Discovery in Databases*, Springer, Berlin, 2000
2. K. Schittkowski, *Numerical Data Fitting in Dynamic Systems*, Kluwer, 2002
3. E. Buckingham, *On physically similar systems; illustration of the use of dimensional equations*, Phys. Rev. 4, 1914
4. E. Buckingham, *The principle of similitude*, Nature, 96:396-397, 1915
5. H. Görtler, *Dimensionsanalyse*, Springer, 1974