

---

# A NON-INTRUSIVE NONLINEAR MODEL REDUCTION METHOD FOR STRUCTURAL DYNAMICAL PROBLEMS BASED ON MACHINE LEARNING

---

**Jonas Kneifl**

Institute of Engineering and Computational Mechanics  
University of Stuttgart  
Stuttgart, Germany  
jonas.kneifl@itm.uni-stuttgart.de

**Dennis Grunert**

Institute of Engineering and Computational Mechanics  
University of Stuttgart  
Stuttgart, Germany  
dennis.grunert@itm.uni-stuttgart.de

**Jörg Fehr**

Institute of Engineering and Computational Mechanics  
University of Stuttgart  
Stuttgart, Germany  
joerg.fehr@itm.uni-stuttgart.de

December 9, 2020

## ABSTRACT

Model order reduction (MOR) has become one of the most widely used tools to create efficient surrogate models for time-critical applications. For nonlinear models, however, linear MOR approaches are only practicable to a limited extent. Nonlinear approaches, on the contrary, often require intrusive manipulations of the used simulation code.

Hence, non-intrusive MOR approaches using classic model order reduction along with machine learning (ML) algorithms can provide remedy. Such approaches have drawn a lot of attention in the recent years. They rely on the idea to learn the dynamics not in a high-dimensional but in a reduced space, i.e., they predict the discrete sequence of reduced system states successively. Open questions are the suitability of such methods in the field of structural dynamics and the best choice of the used ML algorithm. Both are addressed in this paper in addition to the integration of the methodology into a modular and flexible framework that can effortlessly be adapted to various requirements.

By applying the methodology to a dynamic mechanical system, accurate surrogate models are received, which can speed up the simulation time significantly, while still providing high-quality state approximations.

**Keywords** Data-Based Model Reduction, Machine Learning, Surrogate Model, Nonlinear Behavior, Black Box Model, Structural Dynamics

## 1 Introduction

Computer-aided engineering is a pillar of complex technical system design and an essential tool in many industries. Accordingly, product development heavily relies on modeling and simulation results. Models are used to derive control strategies and to evaluate the behavior of technical prototypes quickly. For time-critical applications, however, the conduction of high-fidelity simulations is not always feasible. In real-time applications, predictions must be made within concise time intervals as it is often the case for applications of control theory. In contrast, scanning of high-dimensional parameter spaces for adaptive multi-disciplinary optimization or a robust and resilient system design requires many simulation runs. Often, not only individual components of a model but the expensive system as a whole

is considered. There is no time to perform such a sheer number of simulations in today's very short development cycles.

All this leads to a high demand for efficient but accurate surrogate models. An often-applied technique to mitigate the computational bottleneck is model order reduction (MOR). Its basic concept is to project high-dimensional dynamical systems, like the nonlinear ordinary differential equations (ODE) system of a finite element (FE) model, into a much smaller suitable subspace. In the course of this, the subspace must meet the requirement to reconstruct the original system on its basis with as little loss as possible. For this, a computationally expensive preprocessing, for example evaluating the high-fidelity model at several points in the parameter space, is conducted during an offline stage. Subsequently, the solutions can be approximated during a low-cost online stage by the reduced system for arbitrary parameter sets.

Nevertheless, there is no guarantee that standard MOR approaches can approximate all systems to a satisfactory degree. Strongly nonlinear behavior, as it is frequent in explicit structural dynamics, represents a great challenge among others. Therefore, mature linear model reduction techniques [1] are only practicable to a limited extent. On the contrary, nonlinear methods often require intrusive manipulations of the used solver code [2] or at least knowledge of the system's governing equations. To avoid such problems, non-intrusive model order reduction (NIMOR) methods can be used. They require little or no access to the used simulation code, are straightforward to implement, and can handle black-box models. Accordingly, they are acknowledged as frequently investigated topic [3], e.g., in the field of parametrized MOR [4].

In this work a non-intrusive data-based approach using machine learning (ML) techniques is examined. Machine learning, which is closely related to artificial intelligence, is currently experiencing a level of interest never seen before. The increased computing power enabled a renaissance in data science and gave new momentum to the idea of data-driven solutions. Consequently, the idea to build surrogate models based on data has become more and more popular during the past decade. It remains exciting, which algorithm suits best for which task field. In some modeling methods, ML is used to estimate model parameters [5] or applied to multiscale models [6], while others use data for model adaptation of digital twins [7].

Another data-based procedure to capture the dynamics of a system is to learn a mapping from suitable input variables to the decisive system states. The attempt to achieve this in the full space often fails due to the high-fidelity system's sheer dimensionality, e.g., FE models can have more than  $10^7$  degrees of freedom. Learning and predicting each individual system state is neither efficient nor reasonable in terms of resources. Therefore, a viable approach is to consider the system description in a low-dimensional reduced subspace, so that only a few quantities, the reduced basis' coefficients, must be learned. This results in highly efficient approximations, which are still able to cover the complex dynamics of the original large-scale system by back projection into the high-dimensional space.

Such methods have been applied in the field of fluid dynamics, e.g., to examples like the Burgers equation [8], which is rather academic in its model size and shape complexity. In other papers the temporal sequence of the reduced states is approximated by radial basis functions [9], k-nearest neighbor methods [10] or neural networks [11]. For an overview of further applications, Gao et al. [12] provide an overview of further applications.

The methodology presented in this paper follows a related thought. In the subspace, the coefficients of the reduced basis are iteratively predicted based on an initial state. This corresponds to the aim of approximating a mapping, which maps the current reduced system state onto its following successor. In contrast to the presented research, one major feature of this work is to transfer the methodology to problems arising from the field of structural dynamics. While structural dynamics and fluid dynamics both deal with hyperbolic partial differential equations (PDEs), they differ in the applied discretization methods. Structural dynamics is one of the most important tools in the industrial development of technical systems. Among other topics it is concerned with the investigation of vehicle impact scenarios to increase passenger safety. Thereby, the most significant challenges are nonlinearities resulting from material behavior or nonlinear boundary conditions such as contact. In this paper, such a crash scenario is addressed and the suitability of the data-based NIMOR approach is verified. More specifically, the crash behavior of a racing kart, which can be seen as a simple representation of a lightweight vehicle and thus represents a very exciting and practice-oriented research topic. For this racing kart several surrogate models are created and examined.

In this context, one essential point is the practical applicability and flexibility with which surrogate models can be created. Therefore, another feature of this paper is a framework which is independent of special software solutions and can be applied to a wide range of problems. A critical aspect of this is the possibility of choosing any desired regression algorithm. Each one provides advantages and disadvantages and must be selected depending on the corresponding requirements. This is achieved by a modular structure in which the different regression algorithms can be exchanged and compared quickly. As a result, individual solutions can be tailored precisely to particular problems. To provide the interested reader with a guide for a selection of machine learning algorithms, four of the most promising

ones are implemented. They are compared regarding their required amount of data, simulation speedup, as well as approximation quality using the racing kart model. The algorithms include linear regression, a  $k$ -nearest neighbor algorithm, a neural network, and a Gaussian process. In addition to the algorithms presented here, there exists an enormous number of variations or completely different algorithms with their own benefits, which the reader is strongly encouraged to test.

In order to explain the applied approach, the used workflow and the required theoretical background is presented in Section 2. Hereafter, the methodology is applied to a structural mechanics problem, i.e., a crash of the racing kart frame, in Section 3. Afterward, the results are discussed in Section 4. The paper finishes with a conclusion and an outlook in Section 5.

## 2 Methodology and Process Chain

One possibility to model systems arising from structural dynamics is to use the so-called finite element method (FEM) [13]. During this procedure, a continuous model is discretized into many smallest elements. Unfortunately, many of these finite elements are necessary to represent complex systems leading to a high computing effort. Model order reduction, however, can be applied to limit the resulting computational costs. It is concerned with reducing the computational complexity without losing the essential features of the original complex model.

Nevertheless, in many engineering challenges the exact equations used by software solutions to calculate the underlying dynamics remain hidden and inaccessible. Therefore, the remaining details, like the in- and output of a system, must be sufficient as a basis for analyses, modeling, and simulation. This is of particular interest against the background of commercial simulation and FEM software, where the producers are careful not to reveal their source code. With the left information, the user cannot apply most of the effective nonlinear reduction methods. In such cases, non-intrusive nonlinear MOR methods, where the system's behavior is not drawn from explicit knowledge about the internal equations but learned from data, can present a solution. In the following, a non-intrusive data-driven MOR approach is used to find suitable surrogate models for the dynamics of structural-mechanical systems. Roughly described, this is done by reducing the system's dimension with classical model order reduction first and learning the temporal state evolution in the reduced space via machine learning algorithms afterward.

The methodology is integrated into a modular framework, shown in Fig. 1. Four major steps describe the framework. First, in a computational expensive preprocessing step, high-fidelity FEM simulations are conducted to gather simulation data and compute a reduction matrix. For this purpose, several jobs are first parameterized in Matlab and sent to the FE software's calculation queue. Subsequently, the obtained simulation results are extracted and processed, whereupon a reduced subspace is determined using the extracted data. These first steps are indicated as blue area in Fig. 1.

In the following training phase, highlighted in red, regression algorithms learn the system's unknown behavior based on the generated data. With this knowledge, the nonlinear reduced model can be evaluated in the prediction phase (green) and finally be compared against the full model, indicated by yellow color. These individual steps are explained in detail in the following subsections.

### 2.1 Preprocessing

In order to cover the systems behavior, several high-fidelity simulations with different parameter sets are conducted. Therefore,  $\kappa$  different parameter sets  $\mathbf{p}_i, i = 1, \dots, \kappa$  are sampled to cover a wide range of the parameter space. The offline simulation results, i.e., the FE simulation results, serve as a presumed reference and contain information about the nodal and element quantities of the model. The main interest lies on the nodal quantities. Each of the  $n_{\text{nodes}}$  nodes possesses three translational and three rotational degrees of freedom (DOFs) resulting in six DOFs per node and  $N = 6 \cdot n_{\text{nodes}}$  in common. Accordingly, the state trajectory of the  $n$ -th node during the  $i$ -th simulation run is described by its displacements  $\mathbf{q}_{n,i}(t) \in \mathbb{R}^6$ . Each individual simulation result  $\mathbf{S}_i$  contains the state trajectories for the whole range of nodes and is stored in the so-called snapshot matrix

$$\mathbf{S} = [\mathbf{S}_1(t) \quad \mathbf{S}_2(t) \quad \dots \quad \mathbf{S}_\kappa(t)] \in \mathbb{R}^{N \times \kappa \eta} \quad \text{with} \quad \mathbf{S}_i(t) = \begin{bmatrix} \mathbf{q}_{1,i}(t_1) & \dots & \mathbf{q}_{1,i}(t_\eta) \\ \mathbf{q}_{2,i}(t_1) & \dots & \mathbf{q}_{2,i}(t_\eta) \\ \vdots & & \vdots \\ \mathbf{q}_{n_{\text{nodes}},i}(t_1) & \dots & \mathbf{q}_{n_{\text{nodes}},i}(t_\eta) \end{bmatrix}, \quad (1)$$

where  $\eta$  is the amount of time steps one simulation contains.

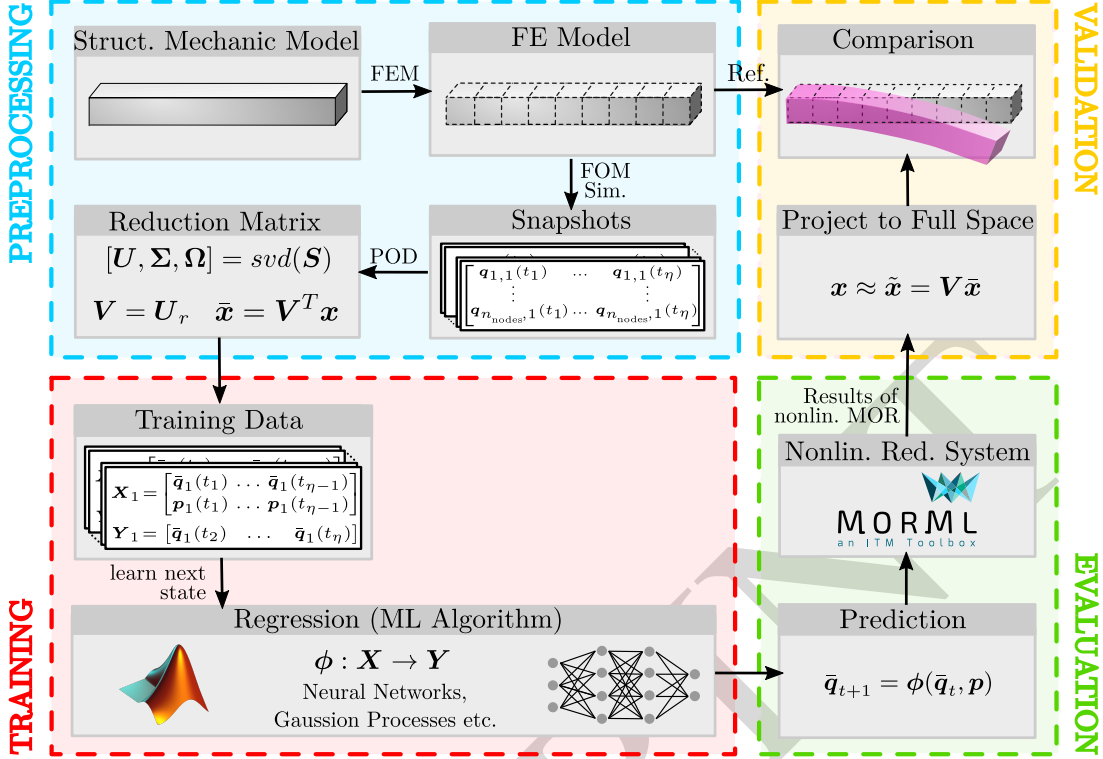


Figure 1: Modular framework to conduct a non-intrusive nonlinear model order reduction by learning a temporal state mapping in a reduced setting. For this, a combination of Proper Orthogonal Decomposition and machine learning algorithms for regression is utilized.

**Model Reduction** With the help of this snapshot matrix, a projection matrix can be found. The key concept of reduction is projection, whereas a projection  $\mathbf{P}$  is defined by the two spaces spanned by the orthogonal matrices  $\mathbf{V}$  and  $\mathbf{W}$ . In general, a distinction is made between Galerkin projections, i.e.,  $\mathbf{V} = \mathbf{W}$ , and Petrov-Galerkin projection where  $\mathbf{V} \neq \mathbf{W}$  [1]. For the sake of clarity only Galerkin projection is considered in the following. The reduced vector space is then defined by the span of  $\mathbf{V}$  and quantities represented in it are marked with a bar throughout this paper, e.g.,  $\bar{q}$ . With the projector's help, the approximation of an original vector can be drawn from the reduced one following

$$\mathbf{q} \approx \tilde{\mathbf{q}} = \mathbf{P}\mathbf{q} = \mathbf{V}\mathbf{V}^T\mathbf{q} = \mathbf{V}\bar{\mathbf{q}}. \quad (2)$$

There exist several approaches to find the necessary reduction matrix. One popular is the so-called proper orthogonal decomposition (POD), which can be used for the reduction and approximation of nonlinear dynamical systems [14]. This is done by representing their state trajectories  $\mathbf{S}_i(t) \in \mathbb{R}^N$  in a lower dimensional space  $\mathbb{R}^r$ , where  $r < N$  holds. It should be noted, however, that the reduced system is prone to errors for trajectories that have not been measured.

The goal of POD is to find a set of orthonormal bases  $\mathbf{u} \in \mathbb{R}^N$  that are able to represent the snapshot matrix, i.e.,  $\mathbf{S} = \mathbf{U}\mathbf{\Gamma}$  with  $\mathbf{s}_i = \sum_{j=1}^N \gamma_{ji}\mathbf{u}_j$ ,  $i = 0, \dots, \kappa\eta$ ,  $\mathbf{s}_i$  and  $\mathbf{u}_i$  representing the  $i$ -th column of  $\mathbf{S}$  respectively  $\mathbf{U}$  and  $\gamma_{ji}$  is an entry of the matrix  $\mathbf{\Gamma}$ . Such bases can be found using the singular value decomposition (SVD)

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{\Omega}, \quad (3)$$

where the coefficients are specified by  $\mathbf{\Gamma} = \mathbf{\Sigma}\mathbf{\Omega}$  with the orthogonal matrix  $\mathbf{\Omega}$ . The columns  $\mathbf{U}$  are known as left singular vectors and those from  $\mathbf{\Omega}$  are accordingly called right singular vectors. The former represent the system's empirical eigenfunctions, which are called POD modes hereafter. Furthermore, the system's singular values are given by the positive diagonal entries of  $\mathbf{\Sigma}$ , which are stored in descending order. The extent of these singular values reflects the contribution of the associated POD mode to the overall approximation. Hence, if they decrease fast, it is possible to approximate the original system with only the  $r$  most dominant POD modes. By simply extracting the first  $r$  columns of  $\mathbf{U}$ , a reduction matrix for a Galerkin-projection

$$\mathbf{V} = \mathbf{U}_r \in \mathbb{R}^{N \times r}. \quad (4)$$

is found. According to Berkooz et al.[15], this results in an optimal linear low-rank approximation with respect to the Frobenius and  $L2$  norm.

## 2.2 Training

The second phase of the process chain consists of the so-called training and includes the processing of the data besides the actual training itself. The aim is to find a nonlinear function  $\Phi : \mathbf{X} \rightarrow \mathbf{Y}$  which maps the current system state, i.e., the nodal displacements  $\mathbf{q}(t_i)$ , onto the following ones  $\mathbf{q}(t_{i+1})$ . In order to increase the prediction quality, an extended parameter set  $\mathbf{p}(t) \in \mathbb{R}^{n_p}$  is attached to the input. It may contain, for example, the simulation parameters selected for the full FEM simulation in which case  $\mathbf{p}(t_i) = \mathbf{p}$  becomes time-independent. Instead of learning this mapping in the high-dimensional original space, it is learned in its reduced representation to minimize the number of variables that must be predicted. Thus, the searched function

$$\bar{\mathbf{q}}(t_{i+1}) = \phi(\bar{\mathbf{q}}(t_i), \mathbf{p}) \quad (5)$$

is supposed to map the reduced representation of a system state  $\bar{\mathbf{q}} = \mathbf{V}^T \mathbf{q} \in \mathbb{R}^r$  on its successor. Based on this, the training data can be composed. The input  $\mathbf{X}$  consists of the states during the first time steps  $t_1, \dots, t_{\eta-1}$  along with the extended parameter sets  $\mathbf{p}(t)$  and the respective output  $\mathbf{Y}$  contains the time-shifted states from  $t_2, \dots, t_\eta$ . According to this description, the data set is obtained as

$$\mathbf{D} = \left( \underbrace{\begin{bmatrix} \bar{\mathbf{q}}_i(t_1) \dots \bar{\mathbf{q}}_i(t_{\eta-1}) \\ \mathbf{p}_i(t_1) \dots \mathbf{p}_i(t_{\eta-1}) \end{bmatrix}}_{\mathbf{X}_i \in \mathbb{R}^{(r+n_p) \times (\eta-1)}}, \underbrace{\begin{bmatrix} \bar{\mathbf{q}}_i(t_2) \dots \bar{\mathbf{q}}_i(t_\eta) \end{bmatrix}}_{\mathbf{Y}_i \in \mathbb{R}^{r \times (\eta-1)}} \right)_{i=1, \dots, \kappa}. \quad (6)$$

Powerful tools to find the underlying relationship of the given data, i.e., find  $\phi(\bar{\mathbf{q}}(t_i), \mathbf{p})$ , are regression algorithms from the field of machine learning.

**Machine Learning** Machine learning enjoys great popularity and provides a bouquet of possibilities and algorithms to gather knowledge out of data. Suppose the previously introduced data underlies some statistical model. Then the task of machine learning can be described as approximation of the original distribution  $\phi(\bar{\mathbf{q}}, \mathbf{p})$  by fitting a function  $\hat{\phi}(\bar{\mathbf{q}}, \mathbf{p})$  to the given data set  $\mathbf{D}$ . There exist numerous different ML algorithms and in certain applications some approaches will be superior to others and vice versa. Therefore, they are integrated modular into the framework making them interchangeable and comparable.

A popular regression approach is to constrain the data to a given class of functions, such as polynomial ones. In the learning process, a finite number of possible parameters is optimized to learn the underlying data in the best possible way with respect to a specific metric. Algorithms following this procedure are called parametric. A serious disadvantage of this approach is that even if there is no knowledge about the data structure, a predefined structure is imposed. Instead of defining a function type from the outset, it can be advantageous to consider all functions that allow a possible representation of the given data. Without a predefined restriction, they can have as many parameters as required. This represents a contradiction to parametric procedures, where the number of parameters is known in advance. Hence, such procedures are called non-parametric, i.e., they have an infinite number of parameters [16]. In this paper two representatives for both types are validated for the intended use. Since the functionality of these algorithms cannot be discussed extensively in this paper, the interested reader is referred to the given references. The used parametric approaches consist of linear regression (lin. regr.) [16] and generalized regression neural networks (GRNN) [17], while the non-parametric ones are  $k$ -nearest neighbor methods ( $k$ -NN) [18] and Gaussian Processes (GPR) [19].

## 2.3 Evaluation and Validation

Once an approximation of the nonlinear mapping (5) is learned, the dynamics of the system can be approximated starting at some initial state  $\bar{\mathbf{q}}(t_1)$ . Iteratively, all following successor states are estimated yielding the dynamics from  $t_2$  to  $t_\eta$ . To compare them with the reference results obtained by the high-fidelity model, they must be projected back into the full space following  $\tilde{\mathbf{q}}(t) = \mathbf{V} \bar{\mathbf{q}}(t)$ .

The steps described in the last subsections cover the complete methodology. In the following, its suitability for the creation of surrogate models is shown using an example from structural dynamics.

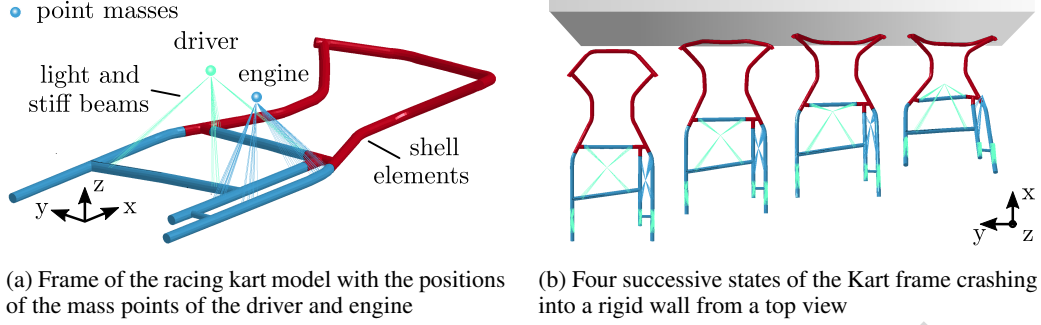


Figure 2: The isolated frame of a racing kart (a) and a visualization of a crash simulation (b).

### 3 Implementation

The scenario, on which the presented methodology’s performance is evaluated, is a crash simulation of a lightweight racing kart [20]. Its model description and the relevant FE software files are provided under Kneifl and Fehr[21]. For the high-fidelity simulations the commercial FEM tool LS-DYNA [22] is utilized, which is mainly applied in crash simulations and high impact dynamics. In this context, it should be noted that any other simulation/FEM software could be used as well since only the simulation data is required for the following steps.

By using the kart model, not only nonlinear material behavior but also nonlinear contact behavior can be investigated. Since the dynamic behavior of a racing kart highly depends on the structural characteristics of its frame, the latter is suitable for studies of model reduction. Thus, the frame shown in Fig. 2a is detached from the remaining bodies, like its rear axle or rotational joints in the front knuckles. As displayed, the weight of the driver and engine are represented by point masses, which are connected to the frame by light and stiff beams.

Crash simulations are carried out as load cases for the kart as visualized in Fig. 2b. During these simulations, the kart frame hits a rigid wall. Up to the moment of impact the kart maintains its initial speed  $v_{init}$ , which is varied in the interval  $[5 \frac{m}{s}, 40 \frac{m}{s}]$  throughout different simulation runs. Applying the POD introduced in Section 2.1 yields the reduction matrix  $\mathbf{V}$  and the singular values  $\sigma$ . Videos of the corresponding ten most dominant POD modes are made available as digital supplementary material [21] along with the simulation results generated by the full model.

From these results, the training data set can be composed. The extended parameter  $\mathbf{p}$  used for the kart simply corresponds to its initial speed  $v_{init}$  yielding the data set

$$\mathbf{D} = \left( \begin{array}{c} [\bar{\mathbf{q}}_i(t_1) \quad \dots \quad \bar{\mathbf{q}}_i(t_{\eta-1})] \\ [v_{init} \quad \dots \quad v_{init}] \end{array} \right)^{i=1, \dots, \kappa}, \quad [\bar{\mathbf{q}}_i(t_2) \quad \dots \quad \bar{\mathbf{q}}_i(t_\eta)] \quad (7)$$

Following the description of Section 2.2, the underlying dynamics are learned from this data providing nonlinear surrogate models, which are validated in the section hereafter. The regression algorithms are incorporated using the Matlab implementation `newgrnn` for the generalized regression neural network and `fitrgp` for the Gaussian process, while own implementations are used for linear regression and  $k$ -NN.

### 4 Simulation Results

Several regression algorithms are introduced in Subsection 2.2, all possessing their own benefits and drawbacks. By creating several surrogate models on their basis, the algorithms’ abilities in the used methodology are made comparable. Different suitable hyper parameters are tested for each algorithm and the most promising ones are presented in the following. This means a number of  $k = 10$  neighbors is chosen for the  $k$ -NN algorithm, squared basis functions are used for the linear regression and the `ardmatern52` kernel function is chosen for the Gaussian process.

For the studies, up to 17 full simulations of the kart model, serving as training data, and three further full simulations, which serve as validation data, are conducted. Appropriate measures of error must be defined to evaluate the results. The first of which is the error of the node displacements averaged over all nodes

$$d_{\text{mean}}(t_j) = \frac{1}{n_{\text{nodes}}} \sum_{i=1}^{n_{\text{nodes}}} d(\mathbf{q}_i(t_j), \tilde{\mathbf{q}}_i(t_j)), \quad (8)$$

Table 1: System settings

Category	Value
Operating System	Debian 10 Buster
CPU	AMD Ryzen 3960X
RAM	256 GB
Language	Matlab R2019b

Table 2: Performance

	$\Delta t_{train}$ [s]	$\Delta t_{sim}^{red}$ [s]	$S_{\uparrow}$	$d_{mean}^{[t_1, t_\eta]}$ [cm]	$d_{max}^{[t_1, t_\eta]}$ [cm]
$k$ -NN	-	0.27	11519	0.92	7.99
lin. regr.	7.87	0.07	19693	1.33	10.34
GRNN	4.97	1.21	604	0.91	6.38
GPR	868.99	0.79	1663	0.39	2.46

where  $d(\mathbf{q}_i, \tilde{\mathbf{q}}_i)$  is the euclidean distance between the node displacements obtained by the full model and those ones obtained by the reduced model. The overall mean distance  $d_{mean}^{[t_1, t_\eta]} = \frac{1}{\eta} \sum_{j=1}^{\eta} d_{mean}(t_j)$  is received by averaging these values over the time. For all investigations only the interesting time period, in which highly dynamic behavior occurs, is considered. Thus, the overall mean distance  $d_{mean}^{[t_1, t_\eta]}$  allows a reliable estimation of the overall performance. Another error measure is the maximum node distance during all time steps  $d_{max}^{[t_1, t_\eta]} = \max_j \max_i d(\mathbf{q}_i(t_j), \tilde{\mathbf{q}}_i(t_j))$ . The last quality criterion taken into account is the simulation speedup factor

$$S_{\uparrow} = \frac{\Delta t_{sim}^{full}}{\Delta t_{sim}^{red}}, \quad (9)$$

i.e. the time required for a full simulation  $\Delta t_{sim}^{full} \approx 582$  s using four cores divided by the online simulation time of the reduced model  $\Delta t_{sim}^{red}$ . All subsequent results are generated on a local workstation with the properties listed in Table 1. Moreover, they are averaged for three test simulations to guarantee meaningful conclusions. The values for the speedup factor are then calculated from the averaged simulation times.

A first outlook on the performance using a suitable amount of training data (9 snapshots) and a medium time step size of the reduced model of  $\Delta t^{red} = 5 \cdot 10^{-4}$  s is given in Table 2. It is particularly noticeable that Matlab's implementation of Gaussian processes requires a comparatively high training time  $\Delta t_{train}$  and that  $k$ -NN and linear regression yield a high speedup. Further discussions about the results follow hereafter.

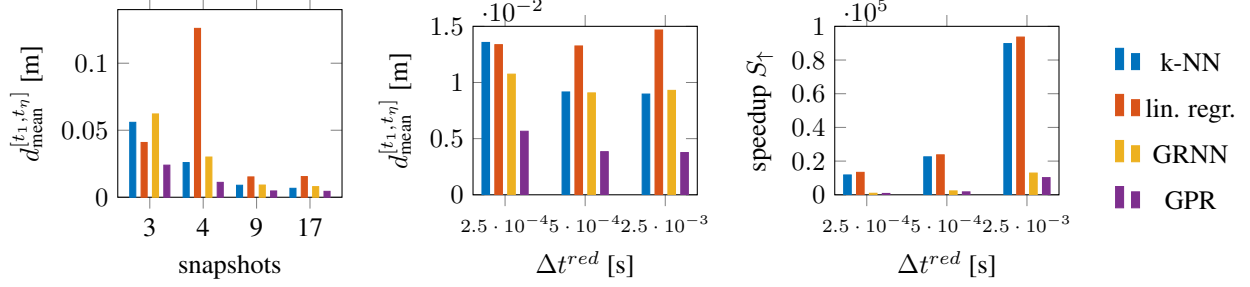
#### 4.1 Algorithm Comparison

In the field of model reduction, a distinction can be made between two kinds of applications. On the one hand those for which the preprocessing phase is time and cost critical and, on the other hand, those for which it is only a matter of saving resources during the online phase. For the former it is of certain importance to conduct as few full simulations as possible. Accordingly, the question how much data is required until further information does not yield significant improvement in the algorithm's prediction quality is of certain importance. For this investigation, several simulations with surrogates using a different number of snapshots, i.e., full simulation results as training data, are conducted. In this context, linear regression yields the most unsteady results and provides a significant outlier using four snapshots as displayed in Fig. 3a. Gaussian processes, on the contrary, yield consistently satisfying results and can convince even using few data sets. However, even if the algorithms do not perform equally well with an additional amount of accessible offline data, it applies to all that there is no relevant improvement if more than nine snapshots are used. Hence, this is the amount used for further examinations.

At this point, it should be positively emphasized that in contrast to classical ODE solvers, the used methodology is not bound to any minimal step size to obtain stable results. Thus, depending on the individual requirements, the step size of the surrogate model  $\Delta t^{red}$  can be arbitrarily increased to further speed up the simulation without a remarkable influence on the approximation quality as displayed in Fig. 3c and 3b. It needs to be clarified that the step size of the high-fidelity model remains unchanged in this comparison since it is bound to small step sizes due to the numerical solvers. Nevertheless, the surrogate's step size is chosen to be at the medium value of  $\Delta t^{red} = 5 \cdot 10^{-4}$  s for further results.

A comparison between the individual algorithms themselves is given in Fig. 4. Considering the temporal evolution of the mean distance for the individual algorithms (4a), it is striking that the Gaussian process outperforms the others. While the  $k$ -nearest neighbors algorithm and the generalized neural network perform very similar, linear regression comes off worst in this category. Regarding the simulation speedup, the circumstances turn as shown in Fig. 4b. The highest acceleration is reached by linear regression, while GPR and GRNN are the slowest. Furthermore,  $k$ -NN is placed between the two peaks and yields a solid simulation time acceleration.

For the final discussion, only the two most promising algorithms are presented. On the one hand, GPR because it provides the best approximation quality, and on the other hand  $k$ -NN, which offers the best trade-off between per-



(a) mean distance error for different amount of used snapshots (b) mean distance error for different time step sizes of the surrogate model (c) simulation speedup for different time step sizes of the surrogate model

Figure 3: Error quantities between the reduced and high-fidelity model obtained by several simulations using a different amount of snapshots for training (a) and for different time step sizes (b, c).

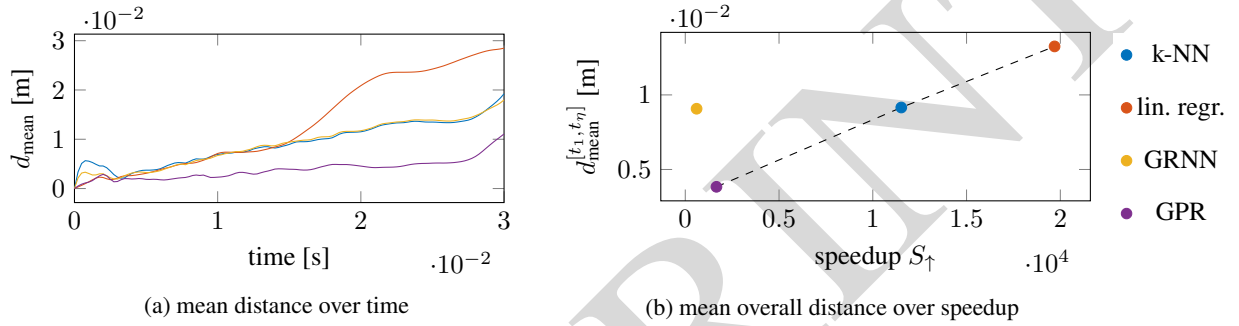


Figure 4: Mean distance  $d_{mean}$  with respect to the time  $t$  (a) and overall mean distance  $d_{mean}^{[t_1, t_n]}$  with respect to speedup  $S_{\uparrow}$  (b) using different regression algorithms, nine full simulations as training data, and a time step size of  $\Delta t^{red} = 5 \cdot 10^{-4}$  s.

formance and speed. Both accelerate the simulation time considerably and approximate the node displacements very well. To supply the reader with additional and more tangible results, a visualization of the simulations at three points in time is given in Figure 5 with accompanying videos to be found in Kneifl and Fehr[21]. There, the high-fidelity results are compared with the low-cost approximations revealing that the kart’s qualitative dynamics are captured in a good manner by both algorithms. While  $k$ -NN overestimates them, GPR slightly underestimates them but is closest to the reference. Therefore, a generally valid recommendation for the best choice of algorithm cannot be given. For tasks where computational acceleration is prioritized  $k$ -NN is clearly the favorable choice since it is not only faster in online simulations but also does not require any training time. However, if approximation quality is prioritized Gaussian processes reveal their potential.

Only approximating positional information of a system may in some cases be insufficient. Hence, it can be beneficial to include velocity and acceleration data into the surrogate model to complete the description of the system. For this purpose, the data set (6) can simply be appended by  $\dot{\mathbf{q}}$  and  $\ddot{\mathbf{q}}$ . This, in turns, changes the mapping (5) to learn the dynamics into

$$\begin{bmatrix} \bar{\mathbf{q}}(t_{i+1}) & \dot{\bar{\mathbf{q}}}(t_{i+1}) & \ddot{\bar{\mathbf{q}}}(t_{i+1}) \end{bmatrix}^T = \phi(\bar{\mathbf{q}}(t_i), \dot{\bar{\mathbf{q}}}(t_i), \ddot{\bar{\mathbf{q}}}(t_i), \mathbf{p}). \quad (10)$$



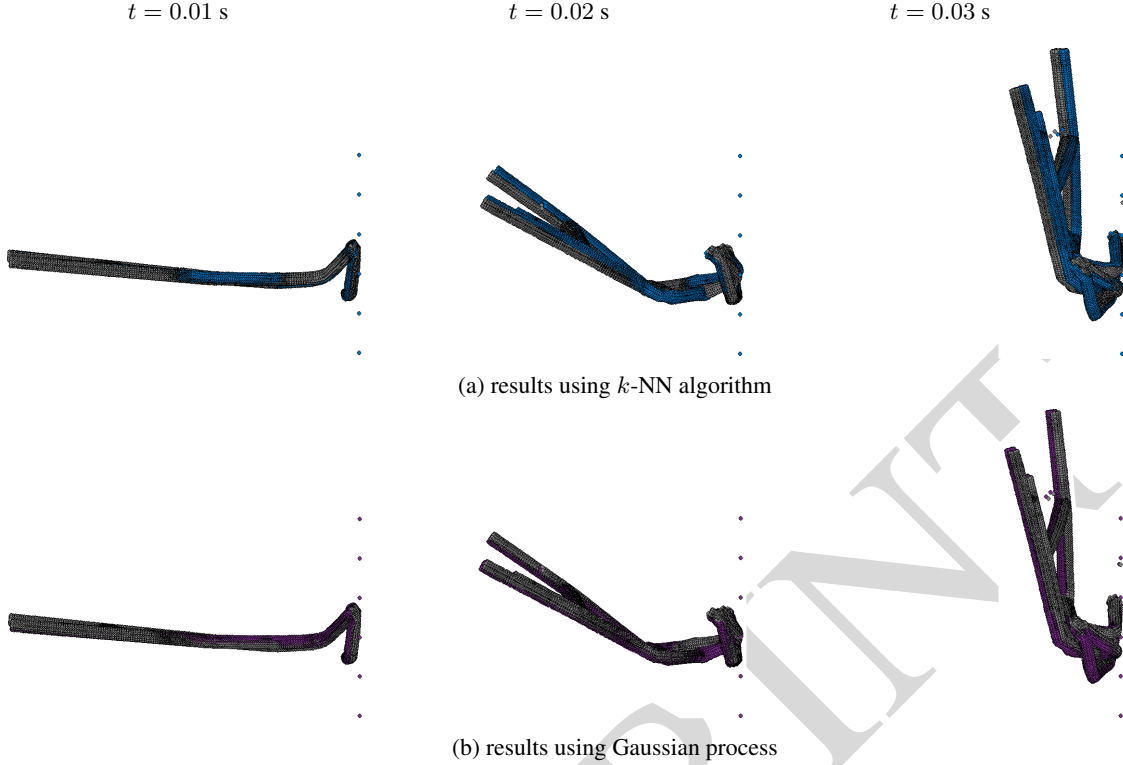


Figure 5: Visualization of different states of kart crash simulations. The FEM simulations results are drawn in gray, whereas the approximations using  $k$ -NN (a) and GPR (b) are represented in the corresponding color.

## 5 Conclusion and Future Work

In this paper a nonlinear non-intrusive model reduction approach for structural dynamic models was implemented and evaluated. Based on a combination of classical projection approaches and regression algorithms from machine learning, the temporal evolution of reduced system states was learned. This methodology yields the benefit of being data-driven. Thus, it is easy to implement for arbitrary approximation problems and capable of capturing nonlinearities. While similar approaches have been applied to fluid dynamic problems, this paper transferred the methodology to structural dynamics. In addition, it integrated the approach into a modular framework enabling fast modifications of the used algorithm, hyperparameters, and learned quantities.

For this purpose, the approach itself and required prior knowledge was presented. Subsequently, it was applied to capture the full dynamics of a finite element crash simulation of a racing kart model. It turned out that the approach has great potential to approximate the underlying system behavior with high accuracy. A decisive criterion for success is the choice of the used machine learning algorithm. The investigations in this work, however, already revealed that there is no universally valid choice but that a different selection must be made depending on the implemented system and desired requirements. Depending on the preference, an acceleration of the simulation time up to the 10000-fold could be achieved.

While the presented approach relies entirely on unknown systems, there are many problem statements where knowledge about the system is at least partially known. In future research, it is one point of interest to include such information into the surrogate models by integrating them into the training phase.

## Acknowledgement

The work presented was carried out at the Institute of Engineering and Computational Mechanics and the Cluster of Excellence Simulation Technology. It was funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2075 – 390740016, Project PN7-6.

## References

- [1] Antoulas A. *Approximation of Large-Scale Dynamical Systems*. Philadelphia: SIAM . 2005
- [2] Fehr J, Holzwarth P, Eberhard P. Interface and Model Reduction for Efficient Explicit Simulations – a Case Study with Nonlinear Vehicle Crash Models. *Mathematical and Computer Modelling of Dynamical Systems* 2016; 22(4): 380–396. doi: 10.1080/13873954.2016.1198385
- [3] Yu J, Yan C, Guo M. Non-intrusive reduced-order modeling for fluid problems: A brief review. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 2019; 233(16): 5896–5912. doi: 10.1177/0954410019890721
- [4] Kostorz WJ, Muggeridge AH, Jackson MD. An Efficient and Robust Method for Parameterized Nonintrusive Reduced-order Modeling. *International Journal for Numerical Methods in Engineering* 2020. doi: 10.1002/nme.6461
- [5] Pawlus W, Karimi HR, Robbersmyr KG. Data-based Modeling of Vehicle Collisions by Nonlinear Autoregressive Model and Feedforward Neural Network. *Information Sciences* 2013; 235: 65–79. doi: 10.1016/j.ins.2012.03.013
- [6] Wirtz D, Karajan N, Haasdonk B. Surrogate Modeling of Multiscale Models Using Kernel Methods. *International Journal for Numerical Methods in Engineering* 2015; 101(1): 1-28. doi: 10.1002/nme.4767
- [7] Kapteyn M, Knezevic D, Huynh D, Tran M, Willcox K. Data-Driven Physics-Based Digital Twins via a Library of Component-Based Reduced-Order Models. *International Journal for Numerical Methods in Engineering* 2020. doi: 10.1002/nme.6423
- [8] Hesthaven JS, Ubbiali S. Non-intrusive Reduced Order Modeling of Nonlinear Problems Using Neural Networks. *Journal of Computational Physics* 2018; 363: 55–78. doi: 10.1016/j.jcp.2018.02.037
- [9] Xiao D, Fang F, Pain C, Hu G. Non-Intrusive Reduced-Order Modelling of the Navier–Stokes Equations Based on RBF Interpolation. *International Journal for Numerical Methods in Fluids* 2015; 79(11): 580–595. doi: 10.1002/flid.4066
- [10] Swischuk R, Mainini L, Peherstorfer B, Willcox K. Projection-Based Model Reduction: Formulations for Physics-Based Machine Learning. *Computers & Fluids* 2019; 179: 704–717. doi: 10.1016/j.compfluid.2018.07.021
- [11] Wang Q, Hesthaven JS, Ray D. Non-intrusive Reduced Order Modeling of Unsteady Flows Using Artificial Neural Networks with Application to a Combustion Problem. *Journal of computational physics* 2019; 384: 289–307. doi: 10.1016/j.jcp.2019.01.031
- [12] Gao H, Wang JX, Zahr MJ. Non-Intrusive Model Reduction of Large-Scale, Nonlinear Dynamical Systems Using Deep Learning. *Physica D: Nonlinear Phenomena* 2020; 412: 132614. doi: 10.1016/j.physd.2020.132614
- [13] Belytschko T, Liu WK, Moran B. *Nonlinear Finite Elements for Continua and Structures*. Chichester: John Wiley & Sons . 2000.
- [14] Antoulas AC. An Overview of Approximation Methods for Large-Scale Dynamical Systems. *Annual reviews in Control* 2005; 29(2): 181–190. doi: 10.1016/j.arcontrol.2005.08.002
- [15] Berkooz G, Holmes P, Lumley J. The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows. *Annual Review of Fluid Mechanics* 2003; 25: 539-575. doi: 10.1146/annurev.fl.25.010193.002543
- [16] Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media . 2009
- [17] Specht D. A General Regression Neural Network. *IEEE transactions on neural networks* 1991; 2(6): 568–576. doi: 10.1109/72.97934
- [18] Bishop CM. *Pattern Recognition and Machine Learning*. Springer . 2006.
- [19] Rasmussen CE. Gaussian Processes in Machine Learning. In: Springer. ; 2003: 63–71
- [20] Shiiba T, Fehr J, Eberhard P. Flexible Multibody Simulation of Automotive Systems with Non-modal Model Reduction Techniques. *Vehicle System Dynamics* 2012; 50(12): 1905-1922. doi: 10.1080/00423114.2012.700403
- [21] Kneifl J, Fehr J. Deformation of a Structural Frame of a Racing Kart Colliding against a Rigid Wall [Data set]. 2020 doi: 10.18419/darus-1150
- [22] Hallquist JO, others . LS-DYNA Theory Manual. *Livermore software Technology corporation* 2006; 3: 25–31.