Towards Simulation-Data Science – a Case Study on Material Failures

Holger Trittenbach Martin Gauch Klemens Böhm Katrin Schulz

Karlsruhe Institute of Technology (KIT)
{holger.trittenbach, klemens.boehm, katrin.schulz}@kit.edu
martin.gauch@student.kit.edu

***Abstract*—Simulations let scientists study properties of complex systems. At first sight, data mining is a good choice when evaluating large numbers of simulations. But it is currently unclear whether there are general principles that might guide the deployment of respective methods to simulation data. In other words, is it worthwhile to target at "simulation-data science" as a distinct subdiscipline of data science? To identify a respective research agenda and to structure the research questions, we conduct a case study from the domain of materials science. One insight that simulation data may be different from other data regarding its structure and quality, which entails focal points different from the ones of conventional data-analysis projects. It also turns out that interpretability and usability are important notions in our context as well. More attention is needed to gather the various meanings of these terms to align them with the needs and priorities of domain scientists. Finally, we propose extensions to our case study which we deem necessary to generalize our insights towards the guidelines envisioned for "simulation-data science".**

***Index Terms*—Simulation Data, Data Science, Materials Informatics, Material Failures**

I. Introduction

For many scientific disciplines, data science plays a significant role. Methods and tools to analyze scientific data have become much more sophisticated and versatile in the last years. In a current research thread, we focus on data generated by simulations. Next to experiments, simulations are a prevalent method in science. With experiments, scientists typically gather data in a controlled environment. Insights come from analyses of observational data using, say, statistical or data mining methods. On the other hand, simulations let engineers abstract from the real-world system and investigate interesting parts of it in isolation.

While both methods are fundamental, they also have certain limitations when looked at separately. For experiments, a substantial, well-documented challenge is to gather a sufficiently large body of observational data [1]. It is difficult to attain a representative sample if the physical variables are numerous. Simulations in turn can generate arbitrary volumes of data, only at the costs of the computational effort. Researchers can systematically examine interesting areas in the configuration space of a system. However, it often is difficult to find a parameterization so that the simulation model accurately represents a given real-world scenario. Next, the usefulness of simulation results depends on the quality of the simulation model, i.e., the level of abstraction from the real world and the extent of simplifications, which may be deliberate.

A current trend is the increasing integration of data-science methods into the process of scientific discovery [1]. There are examples where data-science models have been used as a surrogate for simulations [2], [3]. However, it is still unclear whether there are general principles that might guide the deployment of data-mining methods to simulation data. It is an open question whether it is worthwhile to target at "simulation-data science" as a distinct subdiscipline of data science, independent of the domain.

This article is a first step in this direction. Its core is a case study where we, a team of materials scientists and computer scientists, have studied the deployment of data-science methods on simulation data. We see our study as an intermediate step, to identify questions that are difficult to answer with current methods, and to see which ones regarding the connection between simulations and data science are specific to the application, and which ones are generic. Based on these questions, our objective is identifying and structuring research issues we deem relevant to advance "simulation-data science".

In materials sciences, work that involves data science has focused on the "materials genome", i.e., the identification of materials descriptors and the prediction of properties for novel materials [4]–[6]. Some approaches even target at replacing experiments and simulations with predictions purely based on training data [7]. However, bringing together simulations of materials behavior and data-science methods has received considerably less attention. In our study, we use data from simulations of cracks and the assessment of their effects on the behavior of materials. Scientists currently tend to simulate crack propagation through materials and structures with methods like XFEM [8], [9], phase field modeling [10], [11], or isogeometric analysis [12]. We have identified several questions that are difficult to answer with these existing approaches. These questions will serve as concrete objects of study to make some strides towards our overall objective.

Q1 *Estimation of Current State:* How can the state of the material at time t_k be inferred from sensor measurements observed earlier in time, i.e., at $t_0, \ldots, t_{k-1}$, as well as at t_k?

Q2 *Estimation of Time to Failure (TTF):* How much further load can be applied after time t_k until the material enters a critical state?

Q3 *Probability of Failure:* How likely is a material failure before $t_k + \Delta t$?

In general, simulations generate data for different system variables. For materials, these variables may be stress tensors measurements at different positions on the material. Here, scientists are eager to understand which variables reliably indicate material failures at an early state of deformation. So there are further relevant questions which concern the sensitivity of the answers to Q1 – Q3 with respect to the measurements available.

(Q4) *Variable Importance:* Which variables influence the estimation accuracy most?

(Q5) *Timing of Prediction:* How much does the estimation accuracy improve when the TTF decreases, and more measurements are available?

To address Q1 – Q3, we strive for a mathematical formulation of these questions, e.g., as a regression or classification problem. This has been challenging for our materials scientists, as they had to articulate information needs explicitly. The simulation data must have a certain structure, and the simulation model must be of sufficient quality for a fruitful deployment of data-science approaches. Next, scientists are interested in insights regarding the simulation as a whole. An example is the selection of parameter values one should devote attention to. This may be because they relate to inconclusiveness or cause unexpected results. – Our study has taught us that these premises lead to three essential challenges:

1. Method Selection: A necessary step is to select an appropriate data-mining method from the multitude of existing approaches. However, for insights regarding the simulations as a whole, selecting an approach purely based on performance may not be sufficient. Instead, one might be interested in finding models with certain characteristics such as interpretability and usability. Both characteristics do receive attention in the context of model selection, but their meaning varies considerably, and they often remain unexplained [13]. For simulations, it is not clear how one should formalize and quantify these characteristics. Next, the extent of domain-specificity that is necessary here is unclear.

2. Semantics: A second challenge is that simulations may yield time-series data with special characteristics which cannot be fed one-to-one into conventional analysis algorithms. In our study, the data is a multivariate time series of tensor measurements. The time dimension is not meaningful, because it is an artificial unit intrinsic to the simulation. To illustrate further, the resolution of the data generated can be adaptive, e.g., the simulation generates more fine-granular output if it enters a critical state. This requires adequate prepocessing methods which deviate from the ones for other settings like business data or experimental data.

3. Data Quality: A third challenge is to deal with the quality of the simulation model, which may be low and at the same time not known. Clearly, data generated by a model contains some error when comparing it to the real system. If so, the data might not help to uncover relationships relevant to the analysis or even be misleading. When scientists want to improve the simulation model as a whole, they should know to which extent the data-mining method or simulation-data quality is causal for the low accuracy. Next, despite low costs compared to experiments, it generally is not possible to run simulations for all parameter variations. Scientists must consciously decide which simulations to run. This can result in imbalanced data sets if a certain phenomenon is only observable in a small region of the parameter space.

As a first step in our case study, we map the time-series measurements to a common scale to address Challenge *Semantics*. We then apply feature engineering to facilitate the use of standard statistical methods. To address Challenge *Method Selection*, we have come up with notions of interpretability and usability that turned out to be useful for our materials scientists. For interpretability, materials scientists value models that allow to identify relationships between input data and predictions. For instance, neural networks support interpretability, as they allow to readily single out variables that are good predictors of material failure. Next, we discovered that a distinction between spatial and temporal variable importance contributes to better interpretability. For usability, two aspects have a strong influence in our case study: the effort required to preprocess data for a specific data-mining model, and the algorithm runtime. A naive application of standard performance measures, on the other hand, has not turned out to be overly useful to select models. Instead, our materials scientists have found it more inspiring to compare these metrics for simulations that have been grouped by specific parameter settings. A finding from this group-wise comparison is that, contrary to our expectation, the estimation accuracy (Q5) does not seem to be higher with decreasing TTF.

To address the quality of simulation data (Challenge *Data Quality*), we propose to compare predictions on data generated by the simulation with predictions on the simulation parameters. This comparison has turned out to be particularly useful to trace back data-quality issues to the simulation model.

While the results of our case study have already been insightful for our materials scientists, we deem our conclusions on how a research agenda towards simulation-data science could look like a core contribution of this current article. An important takeaway is that we now can articulate this agenda fairly clearly, see Section V.

II. Related Work

An important current trend in science is to combine domain-specific theory and data science systematically [1]. On the one hand, data science can be useful to improve theoretical models, e.g., through estimation of parameters of physical models. On the other hand, domain knowledge can provide

useful constraints to restrict outputs of the the data-science model. This trend is also visible in materials sciences, where data-science methods become increasingly popular. Several studies have attributed a high potential to data science to advance traditional materials-science methods [7], [14]–[18]. A very prominent application of data science that these studies have discussed is materials design. More specifically, the benefits expected from the deployment of data-science methods concentrate on the Processing-Structure-Property-Performance (PSPP) relationship [19]. The PSPP relationship means that the performance of a material depends on its properties, which in turn depends on the structure, and the structure is the result of specific processing steps. The exact relationships usually are not fully known, and discovering them requires much effort, in order to conduct experiments or develop simulations. Here, data science can be useful in two ways: in a forward direction, e.g., to predict the properties from a material from a given structure, or for the inverse problem, e.g., select the structure which optimizes a specific material property [19].

One may further differentiate between computational materials science and materials informatics [20]. Computational materials science relies on physical models and numerical simulations, while methods from materials informatics are purely data-based, i.e., the application of data-science methods to the materials domain. Computational materials use a multiscale approach. This means that, depending on the problem at hand, models are developed at different spatial or temporal model granularities. To this end, Hill et al. mentions two ways how data science could improve the connection between different model granularities: by using simulation results from high granularity models to predict parameters for models at finer granularity, or by building data science models on top of the simulation outputs [20]. However, the description of both approaches is on a high level, and the authors do not give specific examples. For materials design, there are examples of data science used with different model granularities [7]. The authors argue that the choice of model granularity also depends on the accuracy that one expects from applying data-science methods, but do not elaborate on how to connect models from different granularities. Müller et al. gives a broad overview of data-science applications in the material sciences [15]. Most applications described are on the design of materials, such as the crystal structure predictions or the development and discovery of density functionals. The authors also give examples from other application areas, such as machining and the material behavior under heat treatment and deformation processes.

Data science methods have also been used in combination with simulations in various engineering disciplines. The focus has been to optimize simulation parameters, or to fully replace a compute-intensive simulation model with an approximation, also called metamodeling [2], [3]. The primary target of metamodeling is a reduction of computational runtime. Other approaches have been proposed for the extraction of decision rules to ease the understanding of the simulation [21], [22].

However, the data-science problems discussed in these studies so far have always been a static one, i.e., the training data is a fixed set of attributes or features. These features are, for instance, descriptive properties of materials, results of an experiment, or simulation parameters. An example of a static approach is a comparison of several regression models that take the chemical composition and mechanical properties as an input and predict the fatigue strength of the material [23]. In contrast, a dynamic problem relies on process data gathered over the course of an experiment or simulation, i.e., the resulting data is a series of measurements. Examples for dynamic approaches with experimental data are remaining useful life prediction [24] and electronic nose [25]. To our knowledge, dynamic data from simulations, i.e., data which is gathered at intermediate simulation steps, has not been a focus of research.

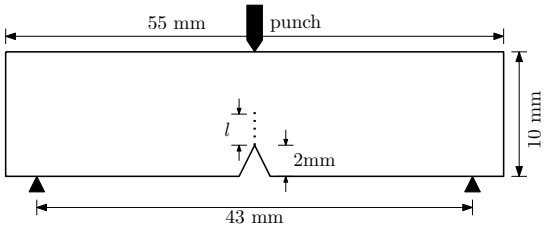


Fig. 1. Three-point bending test.

III. Methods

A. Simulation Model

Our specific scenario is a simple simulation model of a three-point bending specimen, see Figure 1. The goal of the simulation is to investigate how the material behaves when a punch exerts an increasing mechanical stress on its surface. More specifically, the specimen has a notch at the opposite side of the punch, and one wants to find out if and how a crack will open under stress.

We model the specimen with the notch and a predefined crack in the symmetry plane with the finite element method (FEM). This means that the specimen is represented as a mesh of interconnected nodes, which becomes finer around the crack tip. During a simulation, the punch applies increasing deformation onto the structure. This results in an increasing stress in the material. The crack opens when the stress increases beyond a failure threshold, which we set to $y_{failure} = 0.9$. The position of the punch is described relative to its initial one and is called boundary condition displacement (bcd).

As a matter of form, we summarize the simulation parameters in the following paragraph. However, they are not essential for the further understanding of this article.

1) Simulation Parameters: In a real system, the geometry and the material parameters are subject to variation. The simulation varies the system geometry by altering the notch angle ($\alpha \in [15°; 105°]$) and the initial crack length ($l \in [0; 0.2\text{mm}]$). The Young's modulus E = 210 GPa ($\pm 10\%$), Poisson's ratio $\nu = 0.3$, and a prescribed isotropic plastic yield function with the yield stress $\sigma_y = 460$ MPa ($\pm 10\%$) define the elasto-plastic material behavior of the specimen. In the simulation,

TABLE I
ILLUSTRATION OF THE DATA STRUCTURE.

Sim	t	bcd	x_1	x_2	...	x_m	y
s_1	t_0	0	0	0.1	...	0.01	0
s_1	t_1	0.002	4.3	0.1	...	0.11	0.01
s_1	t_2	0.004	7.1	0.2	...	0.22	0.013
$\vdots$							
s_1	t_{K_1}	0.24	22.3	0.6	...	0.66	0.91
$\vdots$							
s_n	t_{K_n}	0.19	4.2	0.42	...	0.48	0.88

we assume a plane strain state. Furthermore, the normal and the shear failure stresses vary with $\sigma_f = 550$ MPa ($\pm 10\%$) and $\tau_f = 0.75 \cdot \sigma_f$. The simulation uses a stress-based *failure criterion* for the interface elements along the symmetry axis, which is defined as:

$$y_{failure} = \sqrt{\left(\frac{\sigma_n}{\sigma_f}\right)^2 + \left(\frac{\tau_n}{\tau_f}\right)^2} \tag{1}$$

The failure criterion characterizes the state of the crack during the loading, dependent on the normal stress σ_n and the shear stress τ_n at the crack tip. It will determine further crack opening if the corresponding nodal value exceeds the critical value $y_{failure} = 0.9$.

2) Simulation Output: During a simulation, the bcd increases stepwise. This applies an external loading in the symmetry axis on the opposite side of the notch. The stress evolves until the material fails, i.e., the full time series is observed. We also make some assumptions about sensor measurements accessible in a real-world application. We assume that measurements are only possible on the material surface, and that the actual stress state in the vicinity of the crack tip is unknown. Therefore, we only consider a selection of output variables at the lower surface of the specimen except for the notch surface. The measurements collected are the displacements, the stresses, the maximum stress value, and its position along the surface, in each simulation step. The data generated by the simulation is a multivariate time series of sensor measurements. Our data set consists of 66,550 simulations.

B. Problem Formalization

The data set D is a set of n simulations $S = \{s_1, s_2, \ldots, s_n\}$. For each simulation s_i, $i \in \{1, \ldots, n\}$, point in time t_k, $k \in \{0, \ldots, K_i\}$ and sensor $j \in \{1, \ldots, m\}$, we observe a measurement $x^i_{t_k,j}$. Table I illustrates the structure of the data. One can subset the observations either by sensor or by time. The subset by time is a vector $x^i_{t_k,\cdot} = (x^i_{t_k,1}, \ldots, x^i_{t_k,m})$ of measurements at time t_k. The subset by sensor is a vector $x^i_{\cdot,j} = (x^i_{t_0,j}, x^i_{t_1,j}, \ldots, x^i_{t_K,j})$ of measurements of Sensor j. The time series from t_0 to t_k is $x^i_{[t_0,t_k],j} = (x^i_{t_0,j}, x^i_{t_1,j}, \ldots, x^i_{t_k,j})$. The failure criterion at time t_k for i is $y^i_{t_k}$. The failure criterion over all simulations at time t_k is $y_{t_k} = (y^1_{t_k}, \ldots, y^n_{t_k})$.

The simulations use an adaptive time stepping method. This means that the temporal resolution of measurements increases if the state changes of the material are large. Consequently, time stamps of different simulations do not necessarily correspond to the bcd (cf. Challenge Semantics), e.g., $bcd^1_{t1} = 0.002 \neq bcd^2_{t1} = 0.003$ in Table I. However, the quantity relevant for our analysis is bcd because it corresponds to the load. So it is more meaningful to index the values by bcd. In the following, we may write bcd_k instead of bcd_{t_k} for brevity. The adaptive time stepping has two implications on the data. First, two subsequent measurements in general are not equidistant in terms of their bcd. Second, the number of observations until the simulation reaches bcd_k, i.e., the length of Vector x^i, differs between simulations. We will return to this in Section III-C.

Q1: One can formulate Q1 as a regression problem. In general, a regression problem is of the form

$$y = f(x) + e \tag{2}$$

where y is the response variable, in our case the failure criterion, x the sensor values observed, and e a random error. In the following, we write bcd_k instead of bcd_{t_k} for brevity. $\hat{f}$ denotes the estimate of the prediction function and $\hat{y}$ the estimate of the failure criterion.

Because the data from each simulation is a time series, the specific regression problem for Q1 is

$$\hat{y}^i_{bcd_k} = \hat{f}(x^i_{bcd_0,\cdot}, x^i_{bcd_1,\cdot}, \ldots, x^i_{bcd_k,\cdot}) \tag{3}$$

Example 3.1: For Q1, a simulation s_1 generates values until $bcd = 0.004$, and the objective is to estimate the failure criterion at this state. Formally, the model estimates $y_{failure}$ right after the punch has moved from its initial position $bcd_0 = 0$ to $bcd_k = 0.004$:

$$\hat{y}^1_{0.004} = \hat{f}(x^1_{0,\cdot}, x^1_{0.002,\cdot}, x^1_{0.004,\cdot})$$

Q2: One can also estimate the failure criterion if the punch advances further to $bcd_k + \Delta bcd$. For example, after observing the sensor values until $bcd_k = 0.004$, one can estimate $y^i_{0.01}$. This gives way to a formalization of Q2. First, we define the critical bcd:

$$bcd^i_{critical} := min\{bdc \colon y^i_{bcd} \geq y_{critical}\} \tag{4}$$

In our case, we set $y_{critical} = 0.9$ (cf. Section III-A1). Q2 targets at an estimate of the difference between the bcd at the time of failure and the bcd at time of prediction, i.e., $\Delta bcd = bcd^i_{critical} - bcd_k$. Because bcd_k is known, we can express Q2 as:

$$\hat{bcd}^i_{critical} = \hat{f}(x^i_{bcd_0,\cdot}, x^i_{bcd_1,\cdot}, \ldots, x^i_{bcd_k,\cdot}) \tag{5}$$

Q3: Q3 aims for an estimate of the probability of failure when additional load is applied, i.e., when the punch advances by Δbcd. First, we define a random variable for the maximum value of the failure criterion between bcd_k and $bcd_k + \Delta bcd$.

$$Y^{max}_{[bcd_k, bcd_k + \Delta bcd]} := max\{Y_v \mid bcd_k \leq v \leq bcd_k + \Delta bcd\}$$

Next, we define X_{bcd_k} as the random variable of the sequence

of measurements until bcd_k. Then the failure probability $P_{Y^{i,max}}$ is defined as:

$$P(Y^{max}_{[bcd_k, bcd_k+\Delta bcd]} \geq 0.9 | X_{bcd_k} = x^i_{[bcd_0, bcd_k],.})$$

Alternatively, one can also interpret Q3 as a classification problem. In this case, there are two possible outcomes: Either the material fails or does not fail until $bcd_k + \Delta bcd$. We can define a random variable $Z \in \{0, 1\}$ which is binary with the probability mass function

$$P(Z^i) = \begin{cases} P_{Y^{i,max}} & \text{if } Z^i = 1 \\ 1 - P_{Y^{i,max}} & \text{if } Z^i = 0 \end{cases} \tag{6}$$

The data mining task is to predict the class label (0 or 1) of the simulation.

C. Preprocessing

For statistical models, a naive way to work with time series data would be to use each measurement until bcd_k as an independent variable. However, the time dependency between observations would be lost. Thus, we extract features from the time series on the interval $[bcd_0, bcd_k]$ for each sensor. We generate a feature on k observed values of the time series by a function of type $\phi\colon \mathbb{R}^k \to \mathbb{R}$, i.e., a function that maps the k observations of a sensor to a value. The set of feature generating functions is $\Phi = \{\phi_1, \phi_2, \ldots, \phi_d\}$. Hence, the transformation of the multivariate time series for one simulation into the feature space is of type $\Phi\colon \mathbb{R}^{m \times k} \to \mathbb{R}^{m \times d}$. $\Phi(x^i_{.,.})$ is the representation of a simulation in the feature space. For better readability, we use the shorthand $\mathbf{x}^i = \Phi(x^{s_i}_{[bcd_0, bcd_k],.})$. After transforming the data into the feature space, the prediction model changes to

$$\begin{aligned} \hat{y}^i_{bcd_k} &= \hat{f}\big(\Phi(x^i_{[bcd_0, bcd_k],.})\big) \\ &= \hat{f}\big(\phi_1(x^i_{[bcd_0, bcd_k],1}), \phi_1(x^i_{[bcd_0, bcd_k],2}), \\ &\ldots, \phi_1(x^i_{[bcd_0, bcd_k],m}), \phi_2(x^i_{[bcd_0, bcd_k],1}), \\ &\ldots, \phi_d(x^i_{[bcd_0, bcd_k],m})\big) \end{aligned}$$

with

$$\begin{aligned} \phi_l(x^i_{[bcd_0, bcd_k],j}) = &\phi_l(x^i_{bcd_0,j}, x^i_{bcd_1,j}, \ldots, x^i_{bcd_k,j}), \\ &l \in \{1, \ldots, d\}, j \in \{1, \ldots, m\} \end{aligned}$$

In our study, the features are maximum, minimum, mean, average and maximum slope, and the area under the curve. Applying these transformations to each of the 607 sensor measurements results in 3642 features.

For artificial neural networks, manual feature construction is not necessary, because neural networks learn low-level feature representations automatically. However, the sample rate for all simulations should be equal for a meaningful analysis. This is not the case here, because the the bcd axis becomes finer when the material enters a critical state. To homogenize the simulation steps, we select the same bcd intervals for all simulations and interpolate the values such that $bcd^i_k = bcd^j_k \; \forall i, j$ by using the next larger value available.

D. Data Mining Methods

We now list the data-mining methods used in our case study. See [26], [27] for more information.

1) Multiple Linear Regression: Linear regression models assume that the failure criterion is a linear combination of the surface values measured.

$$\hat{y}^i_{bcd_k} = {\mathbf{x}^i}^\top \hat{\beta} \tag{7}$$

with the estimated coefficient vector $\hat{\beta}$.

2) Decision Trees: A decision tree segments the feature space into M non-overlapping regions $\{R_1, R_2, ...R_M\}$. The prediction for all data objects that fall into segment R_m is c_m. For a regression task with the quadratic loss function, c_m is the average of the response variables of the data objects in R_m.

$$\hat{y}^i_{bcd_k} = \sum_{m=1}^{M} \hat{c}_m \; I[\mathbf{x}^i \in R_m] \tag{8a}$$

$$\hat{c}_m = \frac{1}{|R_m|} \sum_{i=1}^{n} [y^i_{bcd_k} | \mathbf{x}^i \in R_m] \tag{8b}$$

To avoid overfitting, decision trees are typically regularized, i.e., they have restrictions such as a minimum size of the segments or a maximum depth of the tree.

3) Gradient Boosted Decision Trees: Instead of just learning one decision tree, boosting learns J trees sequentially. The final prediction is the sum of the predictions of the individual trees

$$\hat{y}^i_{bcd_k} = \sum_{j=1}^{J} \hat{f}_j(\mathbf{x}^i) \tag{9}$$

where f_j is the prediction function of the j-th tree. We use so-called *gradient boosted regression trees (GBRT)* [28], [29], an iterative approach. The first tree segments the observation space only coarsely, and each subsequent tree aims to compensate the error of the current model. GBRT typically are regularized as well.

4) Logistic Regression: For the classification task in Equation 6, the model must predict a probability. With logistic regression, the response variable is transformed such that the log-odds are a linear function of the observations:

$$log\left(\frac{P(Z^i = 1 | X = \mathbf{x}^i)}{P(Z^i = 0 | X = \mathbf{x}^i)}\right) = {\mathbf{x}^i}^\top \hat{\beta}$$

By doing so, the prediction is bound to the interval $[0, 1]$. The transformation is similar when using GBRT for classification tasks.

5) Recurrent Neural Networks: Recurrent neural networks (RNN) are artificial neural networks specialized for sequential data, with directed cycles between units, e.g., between hidden units. In our case, the input layer of the network consists of 607 units, one for each sensor. The output layer consists of a single unit, which outputs $\hat{y}_{failure}$, $\hat{bcd}_{critical}$ or P^i_Z. The state of a recurrent hidden unit h at t_k depends on the state

of the hidden unit at t_{k-1}, the sensor values observed at t_k, and model parameters θ.

$$h^i_{t_k} = f(h^i_{t_{k-1}}, x^i_{t_k,\cdot}; \theta) \tag{10}$$

We index by t to emphasize that RNNs work on sequences of input data, independently of the distance between observations. As explained, two simulations can have different numbers of observations until the simulation reaches bcd_k. Although RNNs can handle sequences of different length, we interpolate to make simulations directly comparable.

We use two different recurrent network types in our study. The first architecture is Simple RNN, which consists of self-recurrent units in the hidden layer. The second architecture is Long Short-Term Memory (LSTM) networks, which can remember inputs over a longer time. So the LSTMs can learn dependencies of two distant simulation steps.

E. Model Parametrization

For Linear/Logistic Regression and Decision Trees, we reduce the number of features to 50 with Mutual Information as the selection criterion. For XGB, we select hyperparameters, e.g., max-depth, by random-search with cross-validation. The selection of a suitable topology of neural networks is more difficult, and it is not feasible to search exhaustively. We compare several networks of different complexity, i.e., number of hidden layers and nodes. We opt for three hidden layers with 607, 50 and 50 units for the Simple RNN. For LSTM networks, we use two hidden layers, each with 50 units. We have also tried subsampling and weighted loss functions in case of imbalanced classes. However, this has not improved the results.

F. Error Metrics

We partition the set of simulations into two subsets S_{train} with 80% and S_{test} with 20% of the data. We use standard error metrics to assess prediction accuracy on S_{test}. For regression, we use the Mean Absolute Percentage Error (MAPE). For classification, we use the F1-Score.

IV. Study Results

We first present the accuracy of the prediction models for the regression and classification tasks. The assessment of model accuracy taken by itself has not been overly insightful, but it is the basis to compare different groups of simulations. To this end, we discuss insightful findings on early- and late-breaking simulations, as well as on the timing of predictions. Then we discuss interpretability and usability in Sections IV-B and IV-C. For reproducibility, we have made our data and implementation publicly available[1].

[1]https://www.ipd.kit.edu/simds/readme

TABLE II
MAPE values, lowest errors per bcd_k in bold.

k	Model	$y_{failure}$	$bcd_{critical}$
0.005	Mean	16.60	89.53
	Median	16.89	25.33
	Linear Regression	**5.50**	43.39
	Regression Tree	5.53	**15.13**
	XGBoost	5.51	15.30
	Simple RNN	5.57	15.69
	LSTM	5.53	15.73
0.02	Mean	13.28	114.97
	Median	13.43	39.15
	Linear Regression	5.33	61.00
	Regression Tree	5.01	19.90
	XGBoost	4.71	18.93
	Simple RNN	4.60	19.96
	LSTM	**4.55**	**18.91**
0.04	Mean	6.20	18.06
	Median	6.19	17.92
	Linear Regression	5.00	15.24
	Regression Tree	5.47	16.50
	XGBoost	5.00	15.20
	Simple RNN	**4.88**	15.11
	LSTM	4.95	**15.01**

TABLE III
F1-score for different settings.

k	Logistic Regression	XGB	RNN	N	Positive Class
0.005	0.98	0.99	0.99	13302	82%
0.010	0.92	0.92	0.92	13302	82%
0.015	0.72	0.91	0.92	13209	82%
0.020	0.46	0.90	0.90	6395	63%

A. Model Accuracy

Table II lists the Mean Absolute Percentage Error (MAPE) for prediction of $y_{failure}$ Q1 and $bcd_{critical}$ Q2 for different displacements, i.e., after the punch advanced by 0.005, 0.02 and 0.04 towards the specimen.

In most cases, the average prediction error with standard statistical models is higher than for the neural networks. The largest difference occurs when transiting from a simple linear regression to slightly more complex regression trees for $bcd_{critical}$ at $bcd_{0.005}$ and $bcd_{0.02}$. One can observe that the error grows larger with increasing k for some predictions, in particular for $bcd_{critical}$ from $bcd_{0.005}$ to $bcd_{0.02}$. This might seem unintuitive because increasing bcd means that the model has more information on the simulation available. One explanation is that many simulations break relatively early, i.e., even before the punch reaches an offset of 0.02. We had removed these early-breaking simulations from the test set because predicting $y_{failure}$ or $bcd_{critical}$ for a broken specimen is meaningless. A larger error after these simulations are removed indicates that the early-breaking ones might be easier to predict.

For Q3, we classify simulations into early and late breaking ones. The positive class are simulations with $bcd_{critical} \leq 0.04$, and N is the total number of simulations in the test

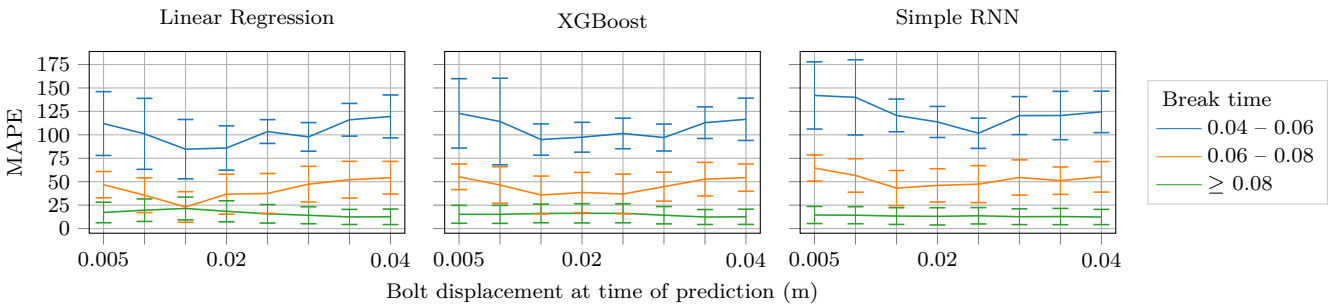


Fig. 2. Prediction error at different bcd. Simulations are grouped by their $bcd_{critical}$.

set. Table III lists the classification accuracy for different models at varying bcd_k. Until $bcd_{0.01}$, all approaches show the same performance. For predictions at $bcd_{0.02}$, class imbalance decreases, and the more complex approaches outperform logistic regression. We hypothesize that this is a result of more complex variable interactions in the remaining simulations which the logistic regression fails to recognize.

1) Timing of Predictions: To further investigate late breaking simulations, we reduce the test set to only contain simulations with $bcd_{critical} \geq 0.04$. Figure 2 depicts the MAPE for the prediction of $bcd_{critical}$. Here, the bcd at time of prediction is varied from 0.005 to 0.04. The simulations in the test set are partitioned further into three groups by their actual $bcd_{critical}$: $[0.04, 0.06]$, $[0.06, 0.08]$, and $[0.08, \infty)$. We observe that the average prediction error is smaller for later break points. This means that, if the prediction takes place at, say, $bcd_{0.03}$, simulations with $bcd_{critical} \in [0.04, 0.06]$ are harder to predict than ones with $bcd_{critical} > 0.06$. However, the prediction error within each group does not change significantly with increasing bcd in any group. So, contrary to our expectation, the answer to (Q5) is that estimation accuracy does not seem to be higher with decreasing TTF.

B. Interpretability

The meaning of the term "interpretability" varies in the literature [13]. In this current study, we focus on the following meaning which has turned out to be particularly appealing to our materials scientists: A data science model is interpretable if it allows to identify insightful simulation runs and parameter settings. We have identified two categories of simulations our materials experts have found interesting. The first one are interesting spatial segments and temporal ranges of the simulations (Section IV-B1). For this category, a possible action could be to increase or decrease the spatial or temporal resolution in some part of the simulation model for further simulation runs. The second category is the identification of settings where data quality might be low (Section IV-B2). Comparing prediction quality for different groups of simulations, e.g., by parameter settings, or after different simulation steps can point to simulations in this category.

1) Variable Importance: Here, variable importance is the relative contribution of a variable to the model prediction [30]. In our study, our domain experts have found a distinction between spatial and temporal variable importance quite useful. "Spatial" refers to sensor locations which the prediction models deem important for a specific bcd. The temporal assessment is to identify the most important simulation steps among all sensor locations. For standard statistical models, a sensitivity analysis can quantify variable importance. However, a direct analysis of the temporal sensitivity is not possible with features that summarize the measurements over several simulation steps. So we use recurrent neural networks instead which allow to assess both spatial and temporal importance.

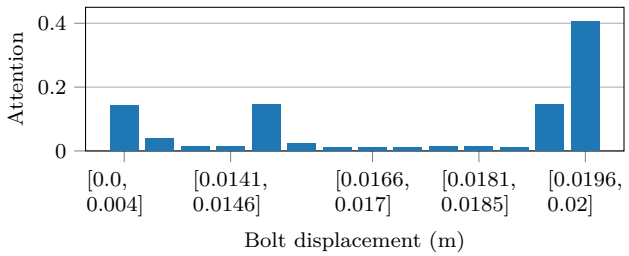
To analyze spatial importance, we use a perturbation method as described in [30]. Perturbation means that one adds noise to sensors one-by-one and measures the increase in prediction error compared to the model trained on unperturbed data. Because of the high number of variables, we choose to visualize variable importance on a schematic of the specimen with the mesh of nodes. Figure 3b shows the right half of the experiment setup (cf. Figure 1). Our material scientists have found this visualization particularly helpful to identify areas on the specimen surface that influence the simulation result most.

To assess the temporal importance, we train an LSTM network with an attention mechanism [27]. This mechanism lets the neural network score the importance of a time step, i.e., the attention it pays to a time step, on the final prediction. Our scientists have liked the visualization in Figure 3a which shows the attention for different bcd intervals. Three periods gain attention: the beginning of the simulation, when the punch is around $bcd_{0.015}$, and the most recent observations.

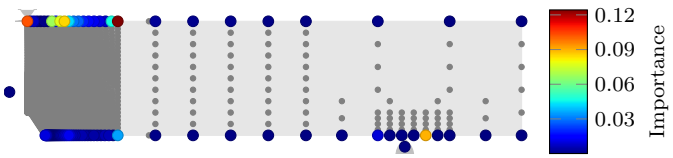
A takeaway is that visualizations that map spatial and temporal variable importance to the simulation setup help to guide the domain experts through the results. Inspecting the simulation model guided by variable importance, for instance to reduce the simulation model complexity, is future work of our materials scientists.

2) Data Quality: Although our regression models after fitting perform reasonably well on the simulation data, some simulations seem to be more difficult to predict than others (cf. Section IV-A). Finding an explanation for this is important. Poor predictions could either be due to an error in the prediction model trained or to the quality of the simulation data (cf. Challenge Data Quality).

We use the following approach to identify parts of the data



(a) Attention grouped by *bcd* intervals.



(b) Profile of half of the specimen with importance of sensors.

Fig. 3. Spatial and temporal importance of simulated sensor measurements.

space where data quality might be an issue. We consider two approaches to answer Q2: The *data-based* approach, where we train a prediction model on the simulation output presented in Section III-B, and a *parameter-based* approach, where we train a prediction model on the simulation parameters. Suppose that the prediction accuracy for the parameter-based approach is good. Intuitively, if there is a causal relationship between the simulation parameters and $bcd_{critical}$, the simulation data should also bear this information. On the other hand, if the predictions of the data-based approach are worse than the ones of the parameter-based approach, this might indicate a lack of data quality. There certainly is no guarantee that data quality is the cause of bad predictions. For example, the prediction model itself might just not be suitable for the data. However, our conjecture seams reasonable if several prediction models fail to learn the relationship. Furthermore, if parameter-based predictions are already bad, this might point to a more general problem with the simulation model. Indeed, the materials scientists in our team have found this comparison helpful in order to assess the simulation model. We illustrate how the comparison between the data-based and parameter-based approach can be helpful to identify potential shortcomings in the simulation model with an example.

Example 4.1: The plot in Figure 4a highlights simulations with low prediction accuracy for the data-based approach. The predictions are from an XGB model at an actual $bcd_{critical}$ between 0.002 and 0.004. Figure 4b shows the parameter-based predictions from a feed-forward neural network. These predictions also are worse than average for the highlighted simulations. Inspecting the highlighted cases brings up that simulations have the same values for all but two parameters, and the actual $bcd_{critical}$ differs significantly. However, the data generated by the simulation model is identical in all attributes until $bcd_{0.02}$. So any data-mining model cannot distinguish between these simulations. Further investigation has revealed that this is a consequence of simplifications in the simulation model. For future simulation runs, materials scientists should adapt the simulation model for this part of the data space.

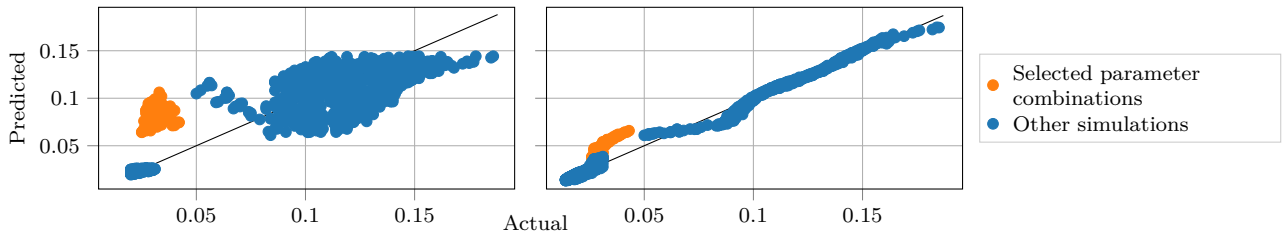
C. Usability

In our study, we have come to the conclusion that it mainly depends on three points whether a method is usable. The first two are the effort necessary for data preprocessing and the runtime of model training. A third aspect has been that finding a good parametrization of models may be difficult in some cases. But this is not specific for simulation data, and we do not elaborate on it in the following.

1) Preprocessing: If preprocessing is non-standard or domain-specific, it entails intellectual effort and implementation work which bog down usability. In our case, the difficulty of preprocessing mainly depends on the specific semantics (Challenge Semantics), which are different than for, say, experimental or business data where instance-wise cleaning of noise or of user-input errors tends to be more common. We in turn use interpolation to make sequences directly comparable. However, this step also distorts the original values and must be well considered. For standard statistical methods, we have derived features like average and maximum slope for the simulation data at hand (cf. Section III-C). Clearly, not all features are suitable for all variables. For a sinusoidal curve for instance, the average might not be as meaningful as, say, the amplitude or frequency. So although feature engineering is a standard step in data mining, it can be challenging here. This is because the product of the number of simulation variables and the one of features can grow very large. Consequently, feature engineering for each variable individually may not be feasible.

2) Runtime: Low runtimes of model training is important here, because the workflow of creating simulations often is iterative. Scientists start with a basic simulation model that might contain unreasonable simplifications or even design flaws. After some simulations, they revisit the model, fix and rerun it. In fact, our materials scientists would have liked more iterations within this current study to improve the models. If data science is part oft the workflow, model training needs to be fast. In our study, these runtimes differ by orders of magnitude. Linear/Logistic Regression and simple Regression/Decision Trees are trained within seconds. For XGBoost, runtimes in our setup have varied between several minutes to a few hours, because of an expensive grid search for model hyperparameters. For RNNs, runtimes can be up to several hours, depending on the architecture and the length of the input sequences. If one even decides to increase the temporal resolution of the simulation, the input sequences grow larger, and runtimes might become prohibitive.

A takeaway from the assessment of usability is that the time



(a) Prediction on data (b) Prediction on parameters

Fig. 4. Prediction vs. actual $bcd_{critical}$ at a bolt displacement of $bcd_{0.02}$ at the time of prediction.

from simulation data to model prediction is crucial. In addition, it is likely that simulation models evolve quickly based on the first results from the data-science models. Furthermore, findings from Section IV-A indicate that model accuracy has turned out to be of less importance in a simulation setting. In consequence, usability might benefit from trading model accuracy for a reduction of time from simulation to prediction, for instance by starting with only one rather simple data-science model.

V. Towards Simulation-Data Science

A core finding from our case study is that we have identified generalizations to our approach which we deem necessary to make some strides towards the guidelines envisioned for "simulation-data science". These generalizations depend on the nature of the modifications made to the simulation setup. We present them as extensions to our study with increasing level of generality.

1. Extended Data Analysis: The first level is that the simulation data available stays as is, only its analysis may be different. Here, it may be insightful to deploy further data-mining approaches, to study other aspects of interpretability and usability. For example, the grid-like nature of the simulation model looks like a good fit for convolutional neural networks. Such topologies could help to identify local motifs, i.e., groups of correlated measurements that are important for certain predictions [31]. Others have experimented with different approaches, like the identification of relationships of simulation variables [32], or clustering of simulations [33]. However, they do not address interpretability or usability as systematically as envisioned here.

2. Advanced Materials Simulations: The second level is to use more advanced simulation approaches to study similar problems of material failures. This requires modifying the simulation model, by varying characteristics such as geometrical details, loading scenarios, strain rate, or by adding material parameters that we have consciously left aside so far for the sake of simplicity. Measurements at the internal nodes of the specimen could also augment the data. Including these nodes in the simulation output might give a more complete view on the specimen. At the same time, this might also increase the dimensionality of the data by orders of magnitude. This in turn may call for methods which can handle high-dimensional data inherently, or to include dimensionality-reduction techniques. With all these modifications, while the general structure of the data remains, the domain-specific research questions might change as well. For cyclic loads for instance, Q2 might be less relevant. On the other hand, predicting the load cycle when the material will most likely fail should be more interesting. Finally, more complex scenarios like cyclic loads might have different data characteristics, like sinusoidal curves, and require a more sophisticated feature engineering than the one here (cf. Section IV-C).

3. Different Simulation Types: The third level is to abstract from material failures and to consider other types of simulations. For example, materials science has certain particularities. One is that it follows a multi-scale approach. This means that researchers may use different modeling approaches of different space and time scales [34]. Our study has been on the macroscopic length scale (centimeter to meter range). This gives way to less complex interactions of model variables than, say, in a micro- or nanometer scale. In general, there are different ways to categorize computer simulations [35], [36]. Examples are continuous vs. discrete time scales or agent-based vs. equation-based modeling. Future work includes selecting different examples from such taxonomies systematically. It may then be possible to evaluate whether the challenges and methods from our study are general, and whether our current understanding of interpretability and usability remains to be meaningful for other simulations.

VI. Conclusions

This article has been a first step to identify and structure open research questions for the deployment of data-science methods to simulation data. Its core has been a case study on material failures. We have proposed extensions to our study which we deem necessary to advance our insights towards general guidelines envisioned for "simulation-data science".

References

[1] A. Karpatne, G. Atluri, J. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, "Theory-guided data science: A new paradigm for scientific discovery from data," *TKDE*, vol. 29, no. 10, pp. 2318–2331, 2017.

[2] G. G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *J Mech Des*, vol. 129, no. 4, pp. 370–380, 2007.

[3] T. W. Simpson, J. D. Poplinski, P. N. Koch, and J. K. Allen, "Metamodels for computer-based engineering design: Survey and recommendations," *Eng Comput*, vol. 17, no. 2, pp. 129–150, 2001.

[4] L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler, "Big data of materials science: critical role of the descriptor," *Phys Rev Lett*, vol. 114, no. 10, p. 105503, 2015.

[5] Y. Liu, T. Zhao, W. Ju, and S. Shi, "Materials discovery and design using machine learning," *J Materiomics*, vol. 3, no. 3, pp. 159–177, 2017.

[6] K. Rajan, "Materials informatics: The materials "gene" and big data," *Annu Rev Mater Res*, vol. 45, no. 1, pp. 153–169, 2015.

[7] R. Ramprasad, R. Batra, G. Pilania, A. Mannodi-Kanakkithodi, and C. Kim, "Machine learning in materials informatics: recent applications and prospects," *npj Computational Materials*, vol. 3, no. 1, p. 54, 2017.

[8] T. Belytschko and T. Black, "Elastic crack growth in finite elements with minimal remeshing," *Int J Num Meth Eng*, vol. 45, no. 5, pp. 601–620, 1999.

[9] N. Moës, J. Dolbow, and T. Belytschko, "A finite element method for crack growth without remeshing," *Int J Num Meth Eng*, vol. 46, no. 1, pp. 131–150, 1999.

[10] V. Hakim and A. Karma, "Laws of crack motion and phase-field models of fracture," *J Mech Phy Solid*, vol. 57, no. 2, pp. 342–368, 2009.

[11] C. Miehe, F. Welschinger, and M. Hofacker, "Thermodynamically consistent phase-field models of fracture: Variational principles and multi-field fe implementations," *Int J Num Meth Eng*, vol. 83, no. 10, pp. 1273–1311, 2010.

[12] C. V. Verhoosel, M. A. Scott, R. De Borst, and T. J. Hughes, "An isogeometric approach to cohesive zone modeling," *Int J Num Meth Eng*, vol. 87, no. 1-5, pp. 336–360, 2011.

[13] Z. C. Lipton, "The mythos of model interpretability," in *ICML WHI*, 2016.

[14] T. Lookman *et al.*, "A perspective on materials informatics: State-of-the-Art and challenges," in *Information Science for Materials Discovery and Design*. Springer, 2016.

[15] T. Müller, A. G. Kusne, and R. Ramprasad, "Machine learning in materials science: Recent progress and emerging applications," *Rev Comput Chem*, vol. 29, no. 1, pp. 186–273, 2016.

[16] S. R. Kalidindi and M. De Graef, "Materials data science: Current status and future outlook," *Annu Rev Mater Res*, vol. 45, no. 1, pp. 171–193, 2015.

[17] N. Wagner and J. M. Rondinelli, "Theory-Guided machine learning in materials science," *Front Mater*, vol. 3, p. 28, 2016.

[18] A. Jain, G. Hautier, S. P. Ong, and K. Persson, "New opportunities for materials informatics: Resources and data mining techniques for uncovering hidden relationships," *J Mater Res*, vol. 31, no. 8, pp. 977–994, 2016.

[19] A. Agrawal and A. Choudhary, "Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science," *APL Materials*, vol. 4, no. 5, p. 053208, 2016.

[20] J. Hill, G. Mulholland, K. Persson, R. Seshadri, C. Wolverton, and B. Meredig, "Materials science with large-scale data and informatics: Unlocking new opportunities," *MRS Bulletin*, vol. 41, no. 5, pp. 399–409, 2016.

[21] H. Pierreval, "Rule-based simulation metamodels," *Eur J Oper Res*, vol. 61, no. 1, pp. 6–17, 1992.

[22] K.-P. Huber and M. R. Berthold, "Simulation data analysis using fuzzy graphs," in *IDA*, 1997, pp. 347–358.

[23] A. Agrawal, P. D. Deshpande, A. Cecen, G. P. Basavarsu, A. N. Choudhary, and S. R. Kalidindi, "Exploration of data science techniques to predict fatigue strength of steel from composition and processing parameters," *Integr Mater Manuf Innov*, vol. 3, no. 1, 2014.

[24] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, 2017.

[25] N. Bhattacharya, B. Tudu, A. Jana, D. Ghosh, R. Bandhopadhyaya, and M. Bhuyan, "Preemptive identification of optimum fermentation time for black tea using electronic nose," *Sens Actuators B Chem*, vol. 131, no. 1, pp. 110–116, 2008.

[26] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer, 2001.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[28] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Ann Stat*, vol. 28, no. 2, pp. 337–407, 2000.

[29] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *SIGKDD*, 2016, pp. 785–794.

[30] M. Gevrey, I. Dimopoulos, and S. Lek, "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecol Modell*, vol. 160, no. 3, pp. 249–264, 2003.

[31] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[32] T. F. Brady and E. Yellig, "Simulation data mining: A new form of computer simulation output," in *Proc Winter Simul Conf*, 2005, pp. 285–289.

[33] S. Burrows, B. Stein, J. Frochte, D. Wiesner, and K. Müller, "Simulation data mining for supporting bridge design," in *AusDM*, 2011, pp. 163–170.

[34] M. Steinhauser, *Computational Multiscale Modeling of Fluids and Solids*. Springer, 2017.

[35] E. Winsberg, "Computer simulations in science," in *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, 2015.

[36] Wikipedia Contributors, "Computer simulation," en.wikipedia.org/w/index.php?title=Computer_simulation&oldid=803382766, 2017, accessed: 2017-10-7.