Lehrstuhl für Technische Mechanik

Titel der Abschlussarbeit

Art der Arbeit

Name des Studenten
Matrikelnummer

Datum der Abgabe
Name des Betreuers

Friedrich-Alexander-Universität
Technische Fakultät

Masterarbeit

im Studiengang Maschinenbau

The title of your thesis goes here

von

Firstname Lastname

Prüfer:	Prof. Dr.-Ing. habil. P. Steinmann
Betreuer:	Dr.-Ing. S. Pfaller
Ausgabe:	DD.MM.YYYY
Abgabe:	DD.MM.YYYY

Universität Erlangen-Nürnberg
Lehrstuhl für Technische Mechanik
Prof. Dr.-Ing. habil. P. Steinmann

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift

Contents

Acronyms

API application programming interface.

CAE computer aided engineering.

EER evaluated strain reactions.

ER evaluated reaction.

ESR evaluated stress reactions.

FEM finite element method.

MD molecular dynamics.

MDB model database.

MSE mean squared error.

ODB output database.

OLR optimized load reactions.

OMP optimized material parameters.

PBC periodic boundary conditions.

RLR reference load reactions.

RMSE root mean squared error.

List of Figures

List of Tables

1 Introduction

research questions

MUSS IN EINLEITUNG GANZ ZUM SCHLUSS The developed process should avoid data transfer between different programs for a higher user-friendliness. Furthermore, it should produce reliable results within a limited timeframe.

2 Basics

This chapter lays the foundations to understand this thesis. First molecular dynamics (MD) as simulation method is introduced. Afterwards the constitutive model is presented. In the last section the used scripting tool with its plug-in for periodic boundary conditions is explained.

2.1 Molecular dynamics

Adhesive joints are important because of XXX FÜR POLYMERE. The extension of usage in further applications depends on a profound understanding of their material behaviour. For investigations on atomistic level molecular dynamics (MD) is a widely used approach [1]. From the interactions with neighbouring atoms Newtons equation is solved for every atom. These interactions are modelled via potentials. Non-bonded interactions like van der Waals potentials are considered within a cutoff radius. The total potential energy of the system helps to identify the acting forces and accelerations of each particle. To follow the movements of the particles time integration is necessary. Usual are small time step sizes in femtoseconds what makes only small time scales possible with suitable computational costs [1]. Similar restriction holds for the system size, due to the increasing number of interactions with increasing domain dimensions. However, small dimensions lead to large surface-to-volume ratios which result in significant free surface effects. To avoid them periodic boundary conditions (PBC) are used. They constrain the simulated region as if it is integrated in an infinitely large volume. Regarding the particle tracking, a particle which leaves the system at one surface enters the system then at the opposite surface. For the deformation of the whole system is restricted in a way that parallel surfaces remain parallel during loading procedures. This boundary conditions result in a simulation of an infinitively long concatenation of the same element in each direction [2]. With these adaptions the results from MD simulations can be transferred to a larger system. Thus, MD simulations allow building samples with prescribed properties followed by deformation tests to study the material behaviour [3]. During a deformation test stress and strain values are measured for every simulation time step. This leads to discrete points describing the stress and strain evolutions over the loading process. To deduce general stress-strain curves from this discrete points, a mathematical expression is required. This is done by constitutive models, which describe the general qualitative relation between stresses and strains [4]. Through material parameters the material specific properties are considered. Depending on the deformation regime for that the constitutive model holds, different material parameters are useful. Polymers are usually modelled with elastoplastic models. In the elastic regime the material parameters Young's modulus E and Poisson's ratio ν are used to specify the material. They show elastic behaviour until a yield strength is reached. Then, the plastification begins which can be described by various hardening models depending on the material characteristics [4].

This work focusses on the investigations by Ries et al. [5] who studied the curing and deformation properties of epoxy through MD simulation. They developed models with

numerous mixing ratios of resin and hardener[1]. Their performed deformation tests build the motivation for the here developed optimization process. Ries et al. [5] ran uniaxial tensile tests loading a sample with a linear strain up to a maximum value of 20 %. The test sample is constrained by PBC which allow lateral contractions. To record the stress-strain response without viscous amounts they developed a procedure to approximate the quasi-static material response. Then only elastic and plastic reactions are considered. Their choice of constitutive models is based on the assumption of isotropic material behaviour. To describe the elastic material behaviour they used the Neo-Hookean hyperelasticity model. The plastic reactions are modelled via the VOCE-model which defines the stresses during the hardening process through (Zitat voce)

$$\sigma = \sigma_0 + \alpha(1 - \exp(-\beta\varepsilon_{pl})) + \gamma\varepsilon_{pl} \tag{2.1}$$

$$\begin{aligned} \sigma_0 &: \text{ Yield stress} \\ \varepsilon_{pl} &: \text{ Plastic strain} \\ \alpha, \beta, \gamma &: \text{ Hardening parameters} \end{aligned}$$

Together with the elastic material parameters Young's modulus E and Poisson's ratio ν, six constitutive parameters are available to fit the stress-strain pairs measured through MD simulation. Their values are calculated with an external minimization algorithm. The detailed procedure is described in [5]. The procedure of Ries et al. [5] is important since their data are used for the model assessment of the optimization procedure developed in this work. A detailed description of the optimization setup is given in chapter 3. In the verification studies the optimization procedure is tested with mixing ratios 4:3, 6:3 and 8:3. To evaluate its performance the optimized material parameters are compared to the material parameter governed by Ries et al. [5]. Though a valid comparison is only possible if, first, the stress strain data are collected under similar loading conditions. And, second, the same constitutive model is used to govern the material parameter values. Thus, a detailed understanding of the methods used by Ries et al. [5] is necessary, since they are adopted to the simulation process used in this work.

2.2 Finite Element Method

The finite element method (FEM) is a widely used approach for XXX. The purpose of FEM is to find solutions for field problems in complex regimes [6]. However, an analytical solution is only possible for simple problem formulations. Therefore, in FEM the regime is discretized into a finite number of elements. The element behaviour is characterized by approximation functions with a finite number of parameters. Assembling the approximation functions of all elements leads to an equation system to approximate the solution for the whole regime [7]. The FEM is mostly used in structural mechanics to provide information about forces and deformations. The general procedure of the FEM is presented in the following:

[1] The mixing ratio is specified in the notation resin:hardener

1. Discretization
2. Construction of stiffness matrix
3. Coordinate transformation
4. Assembling
5. Application of boundary conditions
6. Solving equation system

In the first step we discretize the continuum in finite elements. The shape and size of the elements depend on the geometry of the regime and the required level of precision. To achieve more accurate solutions, smaller elements are necessary. Then, we create for every element a local stiffness matrix $\boldsymbol{K}$, which connects the acting forces $\boldsymbol{S}$ with the element deformations $\boldsymbol{u}$ via

$$\boldsymbol{S} = \boldsymbol{K} \cdot \boldsymbol{u}. \tag{2.2}$$

Although the equation holds for an element, the calculated forces and deformations are defined at the element nodes. The entries in the stiffness matrix depend on the used element type.They contain material specific information through material parameters. The stiffness matrices were constructed in local coordinate systems. To connect them, a transformation into one global system is necessary. In the fourth step the equation systems of all elements are combined into one global system. In the assembly neighbouring elements share their nodes, which needs to be considered during the construction of the global equation system. Through prescribed loadings at the boundary of the continuum, certain forces or deformations are known. They are inserted in the equation system as boundary conditions. In the last step the equation system is solved. The results are the deformations for every node [6][7].
A main advantage of FEM is its high flexibility. Many different geometries can be modelled through an adequate choice of element shapes. The accuracy can be adjusted with the element size. Decreasing element sizes lead to more accurate results but require higher computational effort. However, FEM simulations are normally quite fast, since the computations occur on easy element geometries. In addition, multiple commercial tools are available to construct a FEM simulation. They have multiple options to define the properties during the whole procedure, what makes them useable for a wide range of problems.
As described in XXX, the aim of this work is to create a new procedure to determine the material parameters of materials, investigated with MD simulations. Since a FEM simulation requires significantly less computational effort and still produces sufficiently accurate results, we decided to base this work on FEM. Therefore, we need to transfer the model used in the MD simulations into a FEM model first. As reference, we use the MD simulations performed by Ries et al. [5]. To do this, we must transfer their model properties and their testing conditions into a finite element setting. Especially the material behaviour, the boundary conditions and the applied load must be transferred as exactly as possible. A description of the realisation is given in chapter 3.

2.3 ABAQUS Scripting Interface

The task addressed in this thesis was implemented using Abaqus/CAE 2024. The commercial software is a widespread tool for FEM analysis. At the Institute of Applied

Mechanics (LTM) of the FAU, the software is frequently used, which simplifes its use for potential users. In addition, ABAQUS has an integrated PYTHON-based scripting tool called "Abaqus Scripting Interface". It works as an application programming interface (API) to use the object-oriented programming language PYTHON in the ABAQUS environment (USERGUIDE QUELLE). The "Abaqus Scripting Interface" allows access to the functionalities of ABAQUS/CAE from scripts. Functions, such as the creation and modification of models and jobs, can be controlled via code. As well, the output data written for successful executed jobs can be processed (USWRGUIDE QUELLE). Since it is a PYTHON extension, the standard programming functionalities are available too. The integration of ABAQUS functionalities in a standard program structure, makes the "Abaqus Scripting Interface" the perfect tool for the task of this work. Due to this property, the realisation of the project is possible in a single script, whose structure is shown in chapter 3. NOCH WAS ZU INPUT FILE MIT PARAMETERN UND SCHNELL DADURCH?

ZIEL MONTAG: BASICS FERTIG STELLEN, DANN JONAS UND THERESA GEBEN. IPNUT FILE, EASYPBC, RMSE

2.3.1 EasyPBC plugin

EasyPBC is an ABAQUS plugin to automatically create periodic boundary conditions (PBC). The plugin is developed by XX. The plugin is no official tool, thus it is not online available. Due to the utilisation in previous works, the plugin was easily available. EasyPBC is a tool which creates PBC via constraints in ABAQUS

- bei eva nachgucken - create PBCs auomatisch - needs elastic material parameters - reference points are connected to surfaces –> apply load homogeneous on surface

2.4 Mathematical basics

To find the values of material parameters fitting best the material behaviour measured in the MD-simulation a mathematical formulation is necessary. This leads to an optimization problem, where a calculated error, defined as an objective function of the material parameter values, should be minimized. We first discuss the numerical method to minimize the error and then the construction of the error value.

2.4.1 Numerical optimization

To solve the optimization problem various mathematical algorithms are available. We decided to use the Nelder-Mead algorithm, which is a widely used gradient-free optimization algorithm [8]. In a gradient-free algorithm the derivates of the function are not included in the process. Our objective function is based on results from a finite-element-analysis, which makes it impossible to determine its derivatives directly. Therefore, only gradient-free algorithms come into account. In addition, ignoring the derivatives saves significant computational costs, which leads to fast convergence times [9]. Due to its simple structure the algorithm is a standard feature in many numerical libraries [10]. In PYTHON it is available in the SciPy.optimize-class. In section 3.4 the function call is described in detail. Here we focus on the procedure of the algorithm. The algorithm is capable to find a local minimum of a scalar function depending on n optimization variables. In this work the optimization variables are the material parameters. The definition of the

objective function can be found in chapter 3. Assuming the objective function is known, the first step is to create $n+1$ points $\mathbf{P}$ in an n-dimensional space. In the initial step of the algorithm the position of the points has to be determined. This is done by an initial guess $\hat{x}$ for every optimization variable value. To process six optimization variables the initial guess would look like

$$\hat{\mathbf{x}} = [\hat{x}^0, \hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4, \hat{x}^5]$$
$$\text{with } \hat{x}^i \equiv \text{initial guess of the } i\text{-th optimization variable}$$

Based on this the initial points $\hat{\mathbf{P}}_\mathbf{i}$ are constructed. The first one is defined as $\hat{\mathbf{P}}_\mathbf{1} = \hat{\mathbf{x}}$. For the other points the value of one variable in the initial guess is changed each. The points result in an n dimensional simplex. In the next step the function values corresponding to the points $\mathbf{P}_\mathbf{i}$ are evaluated and sorted by size. The highest function value y_h thus maps the worst value combination $\mathbf{P}_\mathbf{h}$ of the optimization parameters. Afterwards a centroid of all points of the simplex except $\mathbf{P}_\mathbf{h}$ is determined. Now there are four possible operations to improve the position of $\mathbf{P}_\mathbf{h}$. Reflection and expansion of $\mathbf{P}_\mathbf{h}$ at the centroid are the first two. Before the new point $\mathbf{P}^*$ is positioned the corresponding function value needs to be evaluated. Only if y^* is smaller than y_l, $\mathbf{P}^*$ is set as new point $\mathbf{P}_\mathbf{i}$ in the simplex. If y^* is larger than y_l, the new point is even worse than $\mathbf{P}_\mathbf{h}$. Therefore, the operations contraction or shrinking have to be performed. They should find a position $\mathbf{P}^{**}$ between $\mathbf{P}_\mathbf{h}$ and its reflection $\mathbf{P}^*$ which leads to a better function value y^{**}. This needs multiple iterations because for every guess $\mathbf{P}^{**}$ the function has to be evaluated. Only when a better position $\mathbf{P}_\mathbf{h}$ is replaced by $\mathbf{P}^*$ or $\mathbf{P}^{**}$ and the algorithm starts again with the new simplex [11]. Therefore, multiple function evaluations are necessary during one iteration of the optimization. If the variations of the functions values y_i fall under a certain limit, the minimum with its corresponding parameter values is found. To ensure a successful search the initial simplex should be scaled regularly [12] which is possible through a regular distribution of the points $\hat{\mathbf{P}}_\mathbf{i}$ in space. This can be difficult if the values of the optimization variables differ much in size. Therefore, it is necessary to normalize the variable values within the range of 0 to 1.

2.4.2 Root mean squared error

The numerical algorithm can only find adequate results, if the definition of the objective function is suitable.

3 Script setup

The remarks in chapter 2 demonstrate the necessity of an easy and fast procedure to determine material parameters for materials modelled with MD simulations. Deformation tests in MD simulations provide information about the mechanical behaviour of the materials. The aim of the developed optimisation process is to find material parameters which best represent this behaviour. In the following chapter we describe workflow of the optimisation script. First we have a closer look at the structure of the process. Then we introduce the required input data. At the end the implementation is explained.

In deformation tests performed with MD simulations the material behaviour during the loading process is recorded (see section 2.1). Therefore stress and strain values in all directions at discrete simulation time steps are available. In the following stress and strain data, measured during a loading process, are referenced as load reactions. In subsection 3.1.2 we present their structure in detail. To extract material parameters from these load reactions, a constitutive model is required, which describes the stress-strain relation of a material through a functional relation. The constitutive model, with its corresponding material parameters, used in this work is presented in section 2.1. Therefore an evaluation of the material parameters found by the optimisation algorithm is possible. This is done with a FEM simulation as described in section 2.2. It returns the stress strain reactions from a material corresponding its prescribed material parameters, constitutive model and the loading process. Consequently, load reactions measured in MD simulations can be compared with the ones computed in FEM. To represent the mechanical behaviour measured in MD simulations best, a small difference between the load reactions is favourable. Since the material parameters define the load reactions from FEM, their quality is implicitly measured. In other words, we have to minimize the difference between the load reactions to find the best material parameters. We use the Nelder-Mead algorithm, introduced in subsection 2.4.1, to perform this optimisation. This numerical algorithm is capable to minimize the value of a scalar function by optimizing multiple parameters. Optimising all material parameters from our constitutive model is easily possible, while the scalar minimization function poses a challenge. Its function value defines the quality of the optimized material parameters. As explained before, we use the difference between the load reactions to define the optimized material parameters quality. Since we want to fit the whole loading process, the difference at all time steps should be taken into account. To achieve this, we design a procedure, explained in section 3.2, to summarize all these differences into one error value. However, some terms need to be introduced first.

3.1 Input data

In the following sections the required input data are introduced. There are multiple types of input data which are processed at different places in the code. In order to ensure the traceability, clear definitions are required for every type of input.

3.1.1 Load cases and evaluated reactions

A load case defines the direction in which a load acts. Since we want reproducible and easy cases, we only allow loading in normal and principal shear directions. We use the ABAQUS plug-in EasyPBC to apply these loadings. For a consistent naming, we adopt the signatures from EasyPBC for the load cases, defined in Table 3.1a. To model a more complex loading situation, it is possible to apply different load cases one after another. For example, we can apply a normal strain in xx-direction followed by a shear strain in xz-direction. Nevertheless, in one load case only one direction is loaded to avoid mutual influence. The optimisation algorithm requires the load reactions without any constrains for every load case. Only PBC are applied which are described in section 2.1. After the application of a load case, we have to decide which material responses we use to compare with our load parameters. We have the possibility to read out the stresses and strains in all normal and shear directions (see Table 3.1b). The quantities we choose for the comparison are called evaluated reaction. For a high result quality of our material parameters, we try to choose evaluated reactions where we can extract the most information about our material behaviour. These measurements vary depending on the applied load case. In Figure 3.1 an exemplary load case E11 (green) with possible corresponding evaluated reactions (yellow) is depicted.

Table 3.1: Mapping of load directions and evaluated reactions.

(a) Mapping of load directions to load cases.

Load direction	Load case
xx	E11
yy	E22
zz	E33
xy	G12
yz	G23
xz	G13

(b) List of possible evaluated reactions.

Evaluated stress reactions	Evaluated strain reactions
σ_{xx}	ε_{xx}
σ_{yy}	ε_{yy}
σ_{zz}	ε_{zz}
σ_{xy}	ε_{xy}
σ_{yz}	ε_{yz}
σ_{xz}	ε_{xz}

For the verification of the code we use a simple tensile load case E11. In all other directions we impose no restrictions except PBC. As evaluated reaction we use σ_{xx} and the lateral strains ε_{yy} and ε_{zz}. The normal stress contains information about the Young's modulus and the plastic parameters. The lateral strains are necessary for the identification of the Poisson's ratio. Applying a strain in xx-direction will lead to decreasing dimensions in yy- and zz-direction. This reaction is necessary to keep a state of minimum stress. Simultaneously this means that the lateral stresses do not contain any useful information, because they are numerically zero. The validation study is realized with the same load case. Similar to the verification study we take σ_{xx}, ε_{yy} and ε_{zz} to extract information about all material parameters. In the next step we handle load case E11, then G12 and finally combine them. Through the additional obtained information we try to improve the uniqueness of the determined material parameter values. As evaluated reaction for the load case G12 we use the stresses σ_{xy}. As a last study we investigate the

application of cyclic loading as E11 load case. We perform this study with varying load parameters (see subsection 3.1.2).

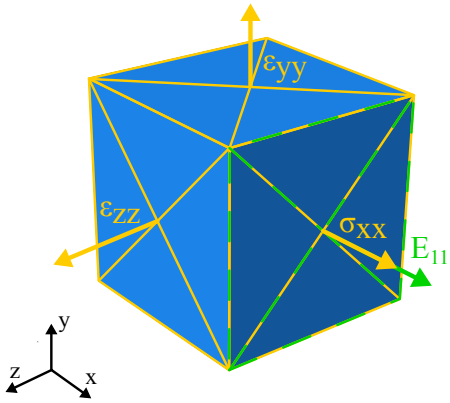


Figure 3.1: Illustration of load case E11 with its corresponding evaluated stress and strain reactions $\sigma_{xx}, \varepsilon_{yy}$ and ε_{zz}.

3.1.2 Load parameters and load reactions

For the optimisation process we use the data from MD simulations as input-file. This data are referenced as reference data in the following. They contain stress and strain values for all time steps during the loading process in all normal and shear directions. Thus, we can register different trends of loading with the corresponding responses. We split the reference data into load parameters and reference load reactions. The load parameters define the quantitative values of the prescribed load case. Since we defined the load cases similar to the ones in the MD simulations, we can process the data from the load parameters directly. If we know the load case, we can easily extract the load parameters from the reference data and transfer them into the Abaqus model. A detailed description of the load application in Abaqus can be found in section 3.3. The reference load reactions represent the material response during the MD simulation. From this, we extract the stress and strain values according to the chosen evaluated reaction. We neglect the remaining stress and strain values, since they probably contain little information about the material behaviour. To perform the comparison of material behaviours we need analogous data from the FEM simulation. In the way described in section 3.4 we can read out stress and strain values in all directions. Similar to the reference load reactions we extract the values corresponding to the evaluated reaction. These values are called optimized load reactions.
Table 3.2 shows the investigated load cases and load parameters. In the verification and validation studies we apply linear strain up to a maximum value of 20 %. For the validation studies we use different mixing ratios which show different mechanical behaviours. Then we investigate a tensile strain following a sinus function over time with a maximum amplitude of 15 % strain. We only consider the first quarter of one period up to the maximum value as a preparation for studies with cyclic loading. In this preparing study we want to investigate how non-linear loadings are handled. Then we use the same load parameters but apply them as shear strain. In the next step we use the same load parameters but combine the load cases of E11 and G12. Important to notice is that the

previously introduced load parameters proceed in a wide strain range. Assuming that the material starts to plastify quite fast, the majority of the loading steps are located in the plastic domain of the material. Conversely, the load parameters contain only little information about the elastic material behaviour. In section 4.1 the issue about this unequal distribution in the material domains becomes clear. As a last study we investigate a full period of a sinusoidal loading for a tensile load case in xx-direction. We use amplitudes of 1 %, 5 % and 8 %. Through the use of this load parameters we try to get a larger proportion of data points in the elastic domain.

Table 3.2: Overview of the performed tests with corresponding input parameters.

Test series	**Load case**	**Load parameters**		**Mixing ratio**	**Evaluated reaction**
		Trajectory	**Amplitude**		
Verification	E11	Linear	20%	6:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
Validation I	E11	Linear	20%	4:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
Validation II	E11	Linear	20%	6:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
Validation III	E11	Linear	20%	8:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
Normal strain	E11	Sinus ($\frac{1}{2}\pi$)	15%	6:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
Shear strain	G12	Sinus ($\frac{1}{2}\pi$)	15%	6:3	σ_{xy}
Normal & Shear strain	E11 G12	Sinus ($\frac{1}{2}\pi$)	15%	6:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}, \sigma_{xy}$
Normal strain	E11	Sinus (2π)	1%	6:3	$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
			5%		$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$
			8%		$\sigma_{xx}, \varepsilon_{yy}, \varepsilon_{zz}$

3.2 Error calculation

As described in section 2.4 the numerical optimisation method requires a scalar function to minimize. Since the trend of the loading process should be represented, we need a method to condense the information of all load reactions into a single value. For a representative value we use the root mean squared error (RMSE), which is explained in subsection 2.4.2. First we extract the reference load reactions and the optimized load reactions in the way described in section 3.4. In the next step we have to build the difference between them. We compute the difference between the load reactions at one load step and then iterate over all load steps. Figure 3.2 displays this procedure for an exemplary set of reference load reactions and optimized load reactions. Here σ_{xx} is the selected evaluated reaction. For every load step LS_i their corresponding reference load reactions (RLR) $\sigma^{\mathrm{RLR}}_{\mathrm{LS}_i}$ and optimized load reactions (OLR) $\sigma^{\mathrm{OLR}}_{\mathrm{LS}_i}$ is logged. The blue arrow highlights their difference $\Delta\sigma_{\mathrm{LS}_i}$ for one exemplary load step, according to Equation 3.1. We square each of these differences to avoid negative values. As described in subsection 3.1.2, the distribution of the data points is unfavourable for the determination of the elastic parameters. To support the algorithm to find the elastic parameters any-

way, we applied a weight of 100 at the data point in the elastic domain. This is done via an array $\boldsymbol{w}$ which entries are one except the first entry w_{LS_1}. In the next step we build the mean value of the weighted arrays. The resulting value is called mean squared error (MSE). We compute the MSE for one evaluated reaction according to Equation 3.2. We compute mse_σ or mse_ε for every selected evaluated reaction.

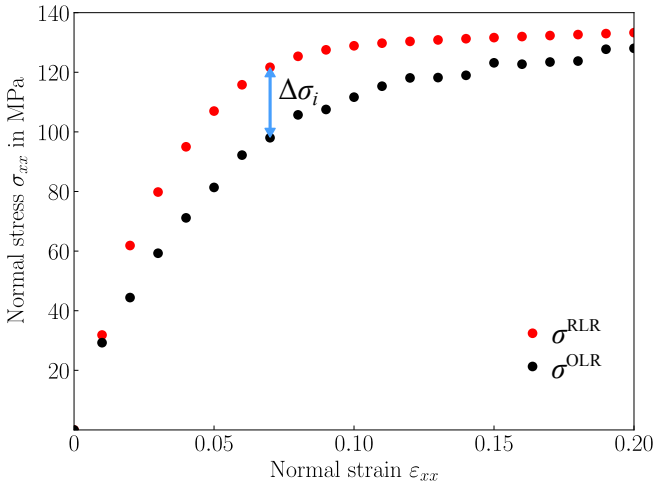


Figure 3.2: Exemplary reference load reactions σ^{RLR} and optimized load reactions σ^{OLR} with visualization of error calculation.

$$\Delta\sigma_{\mathrm{LS}_i} = \sigma^{\mathrm{RLR}}_{\mathrm{LS}_i} - \sigma^{\mathrm{OLR}}_{\mathrm{LS}_i} \qquad \Delta\varepsilon_{\mathrm{LS}_i} = \varepsilon^{\mathrm{RLR}}_{\mathrm{LS}_i} - \varepsilon^{\mathrm{OLR}}_{\mathrm{LS}_i} \tag{3.1}$$

$$\mathrm{mse}_\sigma = \frac{\sum_{\mathrm{LS}} w_{\mathrm{LS}}(\Delta\sigma^2_{\mathrm{LS}})}{\sum_{\mathrm{LS}} w_{\mathrm{LS}}} \qquad \mathrm{mse}_\varepsilon = \frac{\sum_{\mathrm{LS}} w_{\mathrm{LS}}(\Delta\varepsilon_{\mathrm{LS}})^2}{\sum_{\mathrm{LS}} w_{\mathrm{LS}}} \tag{3.2}$$

For the tensile load case for example we must perform this for the evaluated reaction $\sigma_{xx}, \varepsilon_{yy}$ and ε_{zz}. If we now construct a single value out of these MSEs we must ensure a common scale. Otherwise, their influence on the overall error may vary significantly. In general, the MSEs of evaluated strain reactions are much smaller than the ones from evaluated stress reactions, such that loading dependent weights w_σ and w_ε are introduced. The exact weights depend on the load case and the used load parameter set. In general, the MSEs of evaluated strain reactions are much smaller than the ones from evaluated stress reactions, such that a weight w_ε of around 10^4 is necessary for mse_ε. After that we sum up the weighted MSE and construct the RMSE as shown in Equation 3.3. Since our code is able to process multiple load cases in one optimisation process we can calculate the RMSE for every load case and apply weights w_{LC} depending on the load case. Then we sum up all these weighted RMSE values. Additionally, multiple load parameters sets can be processed which leads to a repetition of the described procedure for every load parameter set. Then we can apply weights w_{LP} for every load parameter set and sum it again, which results in a double sum according to Equation 3.4. This value is the one we return our minimization function.In the following sections we have a closer look at the implementation of this minimization process.

$$\text{rmse} = \sqrt{\frac{\sum_{\text{ESR}} w_\sigma \cdot \text{mse}_\sigma + \sum_{\text{EER}} w_\varepsilon \cdot \text{mse}_\varepsilon}{N_{\text{ESR}} + N_{\text{EER}}}} \tag{3.3}$$

$$\text{error} = \sum_{\text{LP}} \sum_{\text{LC}} w_{\text{LP}} w_{\text{LC}} \cdot \text{rmse}_{\text{LC,LP}} \tag{3.4}$$

N_{ESR} : Number of evaluated stress reactions
N_{EER} : Number of evaluated strain reactions

3.3 Preprocessing

Table 3.3: Input parameters for optimisation process.

Input parameter	Directions	Category	Data format	Unit
Young's modulus	–	value	array	MPa
	–	minimum	scalar	MPa
	–	maximum	scalar	MPa
Poisson's ratio	–	value	array	–
	–	minimum	scalar	–
	–	maximum	scalar	–
Plastic Yield	–	value	array	MPa
	–	minimum	scalar	MPa
	–	maximum	scalar	MPa
Alpha, beta, gamma	–	value	array	–
	–	minimum	scalar	–
	–	maximum	scalar	–
Load parameters	–	filename	string	–
	–	weight	scalar	–
Load case	E11, E22, E33, G12, G23, G13	active	boolean	–
		weight	scalar	–
Stress evaluation	xx, yy, zz, xy, yz, xz	active	boolean	–
		weight	scalar	–
Strain evaluation	xx, yy, zz, xy, yz, xz	active	boolean	–
		weight	scalar	–
Load weighting	normal stress, normal strain, shear stress, shear strain	weight	scalar	–

Before starting with the optimisation process, we need preprocessing steps to create a working ABAQUS model with the required properties. In Figure 3.6 the complete

structure of the code is depicted. The white boxes show the individual steps referred to in the text in italics. The coloured boxes represent the performed loops. The upper part belongs to the preprocessing, which start with the step *Read input file*. Table 3.3 lists an extract of this file containing only parameters relevant for the optimisation process. The input file contains more parameters for different namings, which are neglected here. In attaachment XX the whole input file is included. The user has multiple options to specify the optimisation process. It is possible to test multiple initial value combinations for the material parameters calling the script once. In Table 3.3 this is visible in the column *Data format* of the values of all material parameters. This function is important, to verify the optimisation results with varying input values. For every material parameter we write an array of initial values. Then the code loops over all array entries at a time to extract one initial value for each parameter. Therefore, the entries with the same index add up to one initial value combination which is visualized in Table 3.4a. As a consequence all arrays need to be of same length. For all initial value combinations the code creates a new model database (MDB) in Abaqus, and a new folder structure to set the working directory and store the results.

Material Parameter	**Combination**				
	1	**2**	**3**	**4**	**5**
E	E_1	E_2	E_3	E_4	E_5
ν	ν_1	ν_2	ν_3	ν_4	ν_5
σ_{pl}	σ_{pl1}	σ_{pl2}	σ_{pl3}	σ_{pl4}	σ_{pl5}
α	α_1	α_2	α_3	α_4	α_5
β	β_1	β_2	β_3	β_4	β_5
γ	γ_1	γ_2	γ_3	γ_4	γ_5

(a) Arrangement of initial value combination of material parameters.

Model	**Load case**	**Load parameters**
Model 0	E11	Data set 1
Model 1	E11	Data set 2
Model 2	E11	Data set 3
Model 3	G12	Data set 1
Model 4	G12	Data set 2
Model 5	G12	Data set 3

(b) Model creation for load case and parameter combinations.

Table 3.4: Loop conditions in preprocessing.

Afterwards we start the first loop with *Create model*. As discussed in section 2.2 we model a cube with size 1x1x1. We mesh the cube with 6x6x6 elements. Because of the regular geometry, we work with hexagonal structured elements to keep the model simple. The number of elements is a compromise between a coarse mesh for fast computation and a minimum number to avoid convergence errors. Although we use a hyperelastic material in our optimisation process, we first have to build the model with elastic material. We apply isotropic behaviour and define Young's modulus and Poisson ratio as the initial values. The elastic material is necessary due to the usage of EasyPBC in the step *Create job*. We use the load case defined in the input file to create a job. As discussed in subsection 2.3.1, we use EasyPBC for the automatic construction of PBC. Aside from that, we adopt the generated load application corresponding to the load case. Since the load acts on a reference point, connected to the whole loaded surface, a homogeneous load contribution is ensured. However, the settings from EasyPBC contain some default values, we adjust in the step *Modify properties*. EasyPBC applies for every load case a uniform displacement with a standard fixed value, while we want to study the complete loading process. For a correct comparison of the loading reactions we have to evaluate the stresses and strains in the FEM simulation at the same step sizes as in the MD sim-

ulation. In ABAQUS we can solve this issue by creating an amplitude to apply the load gradually (see Figure 3.3). Therefore, we enter the load parameters as an amplitude at certain time steps. For the time steps we use consecutive numbers, since the steps sizes are already defined through the amplitude. The value in the boundary condition editor is then set to one because this only defines the factor by which the amplitude is multiplied (see Figure 3.4). Afterwards we modify the increment settings. EasyPBC automatically creates increments with fixed size and without non-linear geometry effects. In order to avoid convergence errors, we use automatic incrementation. Especially in the first load steps, we observe large deformations. If we try to resolve such large deformations in one incrementation step, ABAQUS runs into convergence errors. With automatic incrementation ABAQUS can adapt the number of increments per load step dynamically dependent of the current deformation. The non-linear geometry effects have to be considered for the same reason. In the last step of preprocessing we store the model in a dictionary. We use this dictionary later to call the models for the optimisation. We perform the preprocessing for all prescribed load cases, highlighted in green in the flowchart. Furthermore, the purple area visualizes the loop over the load parameters. This leads to individual models for every load case and every load parameter set, which is visualized in Table 3.4b.

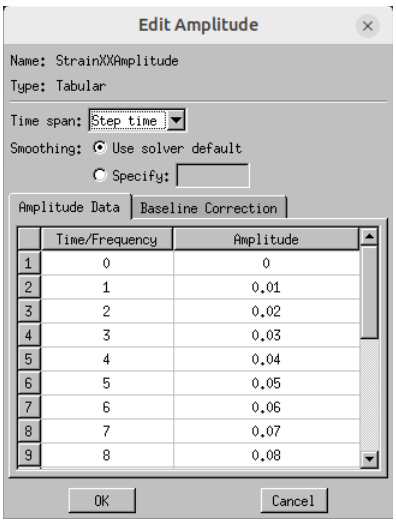


Figure 3.3: Definition of load amplitude in ABAQUS.

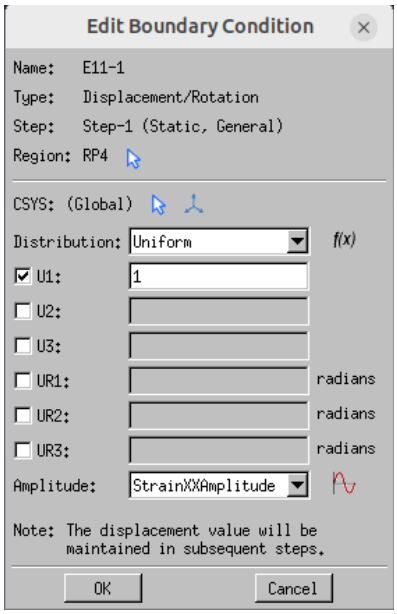


Figure 3.4: Boundary condition menu in ABAQUS.

Figure 3.5: Loop conditions in preprocessing.

3.4 Optimisation process

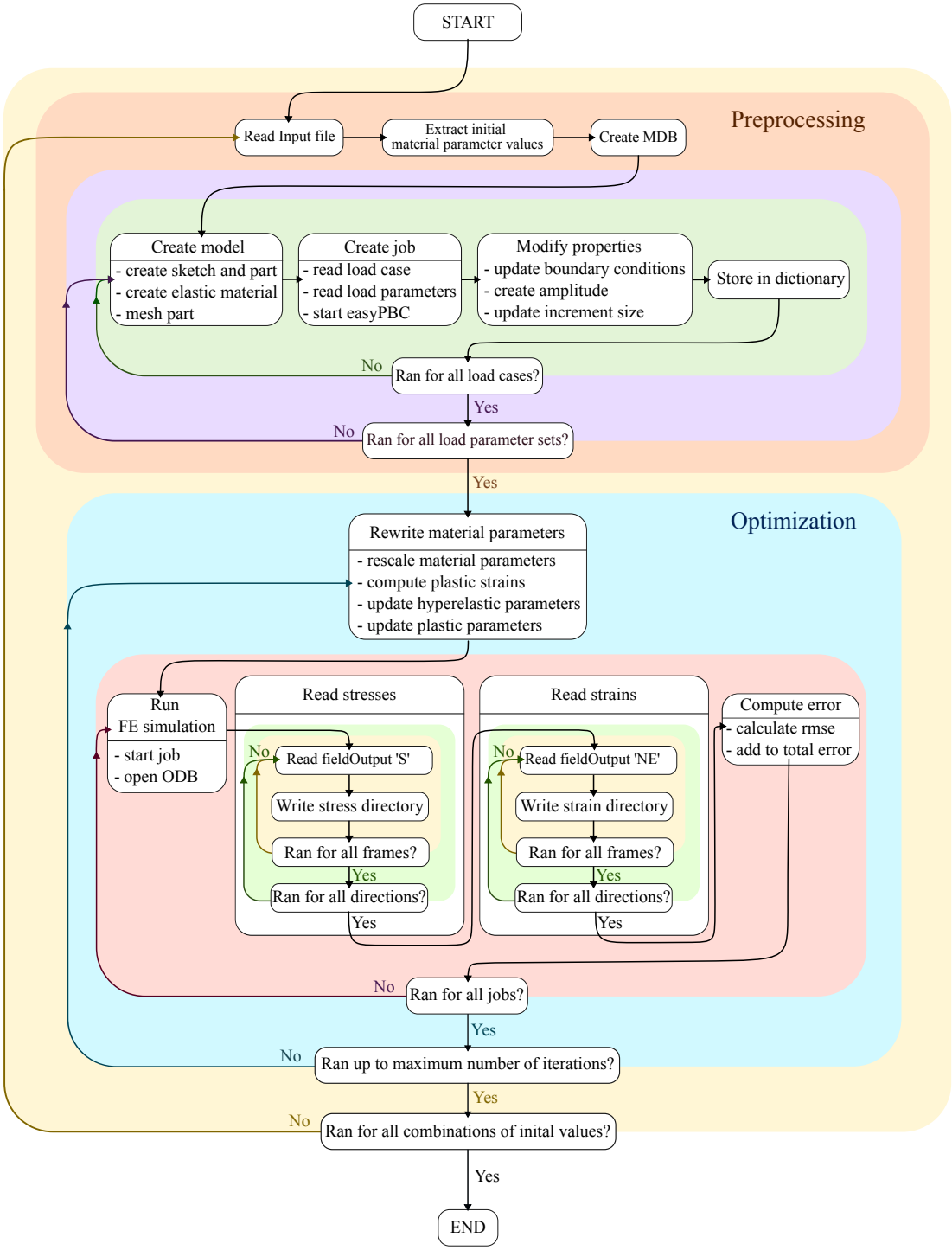


Figure 3.6: Flowchart code.

Table 3.5: Input parameters for SciPy minimize function.

Parameter	Content	Data format	Explanation
Objective function	Optimisation function	–	Function whose scalar value should be minimized
Initial guess	Material parameters	array	Scaled initial values for the optimisation parameters
Additional arguments	Cube parameters	object	Model information from input file
	Load parameters	dictionary	Load parameters from MD-simulations
	Work directory	string	Path to store results
	Evaluation counter	scalar	Counter for the performed function evaluations
method	Nelder-Mead	–	Numerical algorithm
bounds	Limits for material parameter	array	Upper and lower boundary values for every optimisation parameter
maxiter	Number of iterations	scalar	Maximum number of iterations as termination criterion

In the following section, we describe the optimisation process. We start the process by calling the Scipy-minimize function. We pass this function various parameters, listed in Table 3.5. The minimize function itself calls our self-written optimisation function, where the evaluation takes place. The initial guess stores an array with start values for all parameters that should be optimized. Additionally, we pass information about the models that we created in the preprocessing and the load parameters. We use all models created in the purple box for one optimisation call. We start the process with the step *Rewrite material parameters* for all models. Since they all describe the same material, we write the same values for every model. For the minimization computation, optimisation parameters are scaled in the bounds from zero to one. To rewrite the values in the models, we have to rescale them first. Then we can use the rescaled parameters to compute the values for the plastic stress function with the formula for VOCE-hardening (Equation 2.1). In the following we remove the elastic material and substitute it with a hyperelastic material which is suitable for high non-linear deformations. Therefore, we have to convert the Young's modulus and the Poisson's ratio into the hyperelastic parameters C_{10} and D_1 via the relations

$$C_{10} = \frac{E}{4(1+\nu)} \tag{3.5}$$

$$D_1 = \frac{6(1-2\nu)}{E}. \tag{3.6}$$

Now we can update all material values. In the next steps we handle the models successively. We start the job of the first model to perform the Abaqus analysis and open the resulting ODB in *Run FE-simulation*. With step *Read stresses* the evaluation begins. We do this by reading the 'FieldOutput' variable 'S' and write the data in a

stress directory. Since we need stress-values at all steps defined by the amplitude, we read out every frame from the ODB. One frame corresponds with one step in the loading process. Additionally, we loop over all directions (xx, yy, zz, xy, yz, xz). The same procedure is done for the strain values in *Read strains*. Here it is important to read out the correct strain variable 'NE' (nominal strain). For hyperelastic materials, ABAQUS uses the logarithmic strain ('LE') as standard value. Because of its logarithmic scaling, we cannot compare them to the reference data. Then we store all values for all frames and directions in a dictionary, similar as for the stresses. Now we have collected all required data to perform the step *Compute error*. We do this in the way described in section 3.2. For a better structure of the code this part is outsourced in a separate function. We call this function and pass the stress and strain directories as well as the corresponding load parameters. Then the computation runs and the function returns the RMSE for this job. Multiplied with its corresponding weights for its load case and load parameters we add this value to the total error value. Now we restart the red loop by starting the FE-simulation for the next job which is visualized in the red box in the flowchart. Once all the jobs are processed, and we computed the total error value, we return it to our minimization function. Through the internal Nelder-Mead algorithm the error is reduced and it returns the corresponding material parameters. Now one optimisation iteration is performed and it starts again. In the flowchart, the loop is visualized with the blue box. This process will run until our defined number of maximum iterations is reached or convergence is reached. When the changes in the error value are very small, the internal convergence criterion is reached and the algorithm stops.

3.5 Storage of data

To evaluate the optimization process at the end, parameter values need to be recorded during the preprocessing and the optimisation process. This is the only way to ensure that the values of all iterations are saved, since the variable values are rewritten in every new iteration. In the preprocessing, we extract the input file with the current initial value combination. All files created from ABAQUS like the CAE, ODB and the job-files are stored in a subfolder. In the optimisation part, multiple functions are called after the step *Compute error* to store the current variable values. The material parameters for every iteration are stored. In addition, we store multiple interim results during the error calculation. For every job the weighted MSEs are stored for every iteration, so that the impact and evolution of every evaluated reactions can be understood. To track the impact of the applied weights, all RMSE values are stored individually and weighted. In addition, we store the total error, which is the sum of all weighted RMSEs of one iteration. Finally, we have one file each for the material parameters, MSEs, RMSEs, and the total errors. All these files are handled in the same way. After the computation of the total error, all files are opened and the variable values of the current iteration are stored in a new line. The stress and strains are stored in separate files for each iteration. We create for each job a subfolder, where its stress and strain values are stored. For one iteration, a file is created which contains the stress and strain dictionaries from the steps *Read stress* and *Read strain*. This leads to as many files as iterations were performed for every job. At the end of the optimisation process, we store the final message returned from the Scipy.minimize function. PYTHON automatically sends a message where the cause of termination is stated. This message becomes important if the algorithm stops before the maximum number of iterations is reached. Then we can reproduce whether this happens because the convergence limit is reached or a numerical error occurred.

The described procedure to store the data is done for every initial value combination separately.

TABELLE MINIMIZE FUNCTION IN FLOWCHART INTEGRIEREN

4 Results

In this chapter the results from different test cases presented in section XX will be evaluated. First, we discuss the verification results to understand the general behavior of the optimization process. In the next step we validate the optimization performance using different data sets. Finally, we present the results of the cyclic load cases.

4.1 Verification

In this section, we discuss the results of the data set used for the verification of our optimization process. To evaluate the quality of the optimization process, we define characteristic quantities and investigate their evolution. Noch den load case kurz aufschreiben.

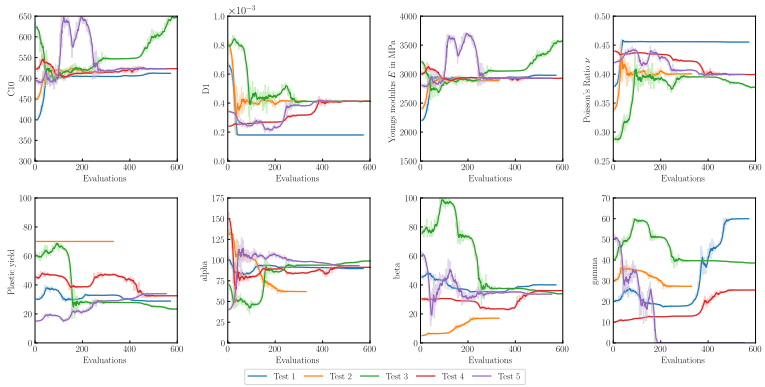
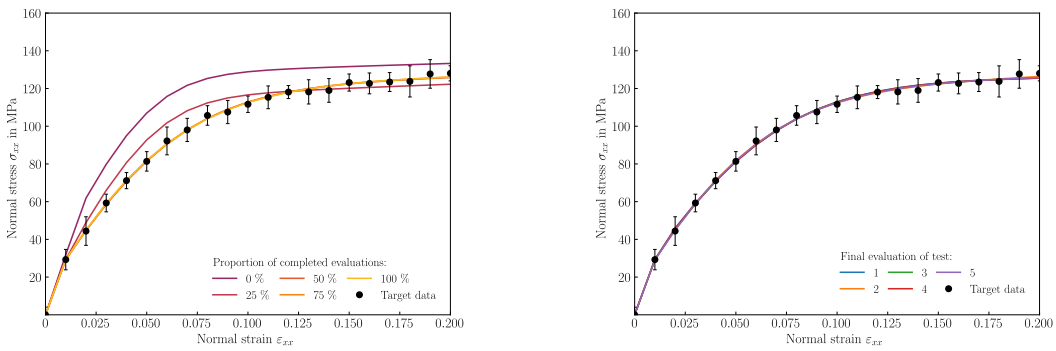


Figure 4.1: optimization progress of material parameters.

As process quantities we log the evolution of the material parameters during the optimization. Since these are our input parameters we performed multiple tests with varying parameter combinations to evaluate the stability of our program. The results are presented in 4.1. In the first row the elastic material parameters are presented. For a better understanding, we transformed the hyper-elastic parameters $C10$ and $D1$ into Young's modulus E and Poisson ratio ν. In the second row the plastic material parameters are presented. For elastic and plastic parameters we can observe convergence for every parameter combination. However, the converged solutions differ from each other for most of the combinations. For a detailed discussion of possible reasons we separate between elastic and plastic material behavior. Since our tested material shows an elastic response only up to the second data point, there might be not enough optimization points to find an unique solution for the elastic material parameters. Nevertheless, we have to investigate other characteristic quantities to ensure our assumption and understand the influence of the plastic parameters. In **??** the progress of the stress-strain curve in normal direction during one exemplary test together with the target curve from the MD simulation is presented. The optimized curve matches the target curve after only 25% of the

evaluations. Since the stress-strain curve is one of our target data this progress indicates a correct optimization behavior of our algorithm. To support this assumption the final stress-strain curves of the test series are depicted in **??**. Despite the high variance of the final material parameters, the stress-strain curves all matches the target data for the stress-strain curve in normal direction. Because of this deviation we depicted the influence of the parameters on the trend of the VOCE-hardening curve shown in Figure 4.3. The parameter alpha has the greatest impact on the shape of the curve, while a variety of 50% in the parameter gamma has hardly any visible effect. This leads to a high flexibility in adjusting the shape of the curve and as a consequence to multiple possible parameter combinations to fit the target curve. To support our assumption we check the quality of the optimization result by plotting the progress of the root mean squared error (RSME) for all the tests in Figure 4.4. We observe that a common minimum RMSE value is reached in all optimization runs, indicating that the results are of equivalent quality. The results of this verification tests lead to some states about the quality of the optimization algorithm. The results of the stress-strain data show a good match of the optimized curve with the target data for every test which is confirmed by the RMSE. In contrast we observe a high variance in final the material parameters found by the algorithm. These results suggest that the algorithm can generally find parameter values to match the target data. However, the variance in these optimal parameters shows that multiple parameter combinations lead to the same quality of result. This behavior may be due to the relatively large number of optimization parameters compared to the dimension of the target data. To verify this assumption we reduced the number of material parameters. Since only the first point of the target data lies in the elastic domain, we fixed the elastic material behavior and computed them directly from the data of the first point. Then we tested this new configuration of the algorithm with the current target data and two other data sets.



(a) Evolution of stress–strain curve.

(b) Final stress–strain curves.

Figure 4.2: (a) Evolution of the stress–strain curves during optimization for an exemplary test, (b) final stress–strain curves for the complete test study..

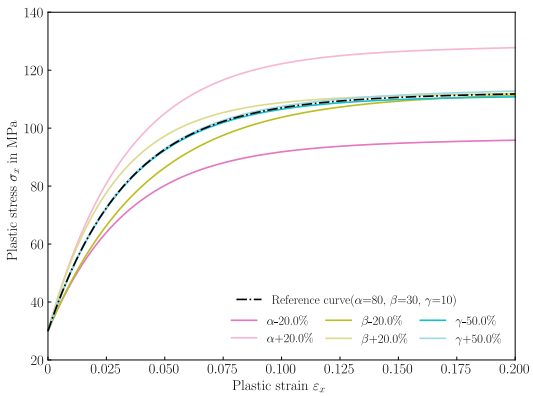


Figure 4.3: parameter influence on voce-hardening curve.

PICTURES with Krümmung und Steigung –> parameters to characterize curve

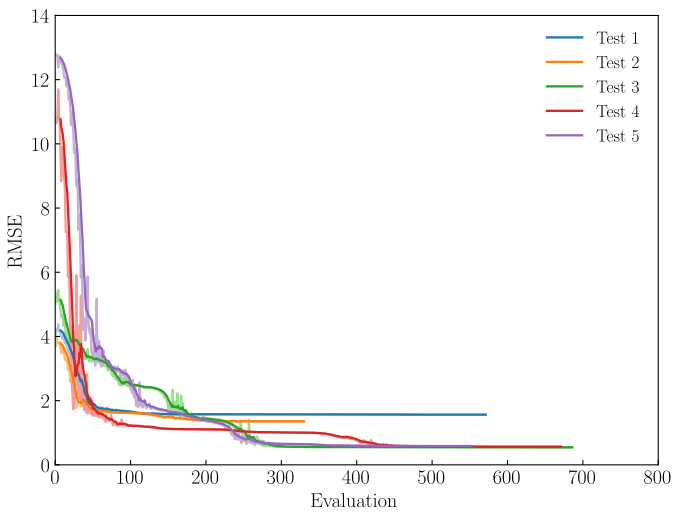


Figure 4.4: rmse for multiple tests.

4.2 Validation

To improve our algorithm we reduced the number of optimization parameters by fixing the elastic material parameters. The results of this implementation for three different data sets will be discussed in this section.

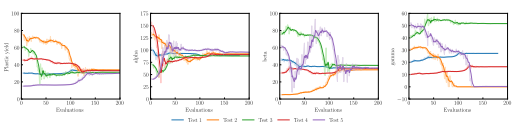
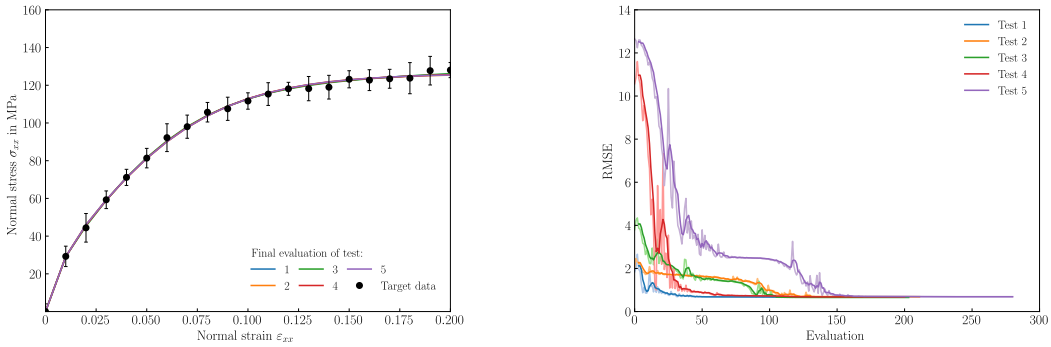


Figure 4.5: progress of material parameters for validation tests.

In the first step we used the same data set as in **??** to test the modified algorithm. The optimized plastic material parameters are shown in Figure 4.5. In all tests, the

values of the plastic yield, alpha and beta demonstrate a converging trend towards a singular solution. The only exception to this is gamma whose converged values vary for each individual test. As was outlined in the preceding discussion, gamma exerts minimal influence on the trend of the hardening curve. Consequently, the focus shall be directed towards the plastic yield, alpha and beta, which indicate an improvement in their optimization behavior. The quality of this optimized parameters is ensured through the match of the stress-strain curves with the target data. As demonstrated in Figure **??** the final stress-strain curves exhibit a strong correlation with the target data. The evolution of the RMSE XXX supports this results with small values for every test. A comparison of the results of the present study with those of the verification study reveals an equivalent level of optimization quality. Additionally, the results of the material parameters indicate a positive impact of the algorithm modification, showing a unique solution for the important parameters.

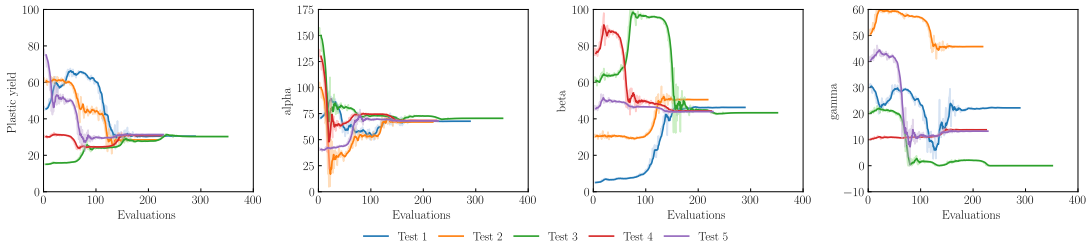


(a) Final stress-strain curves.

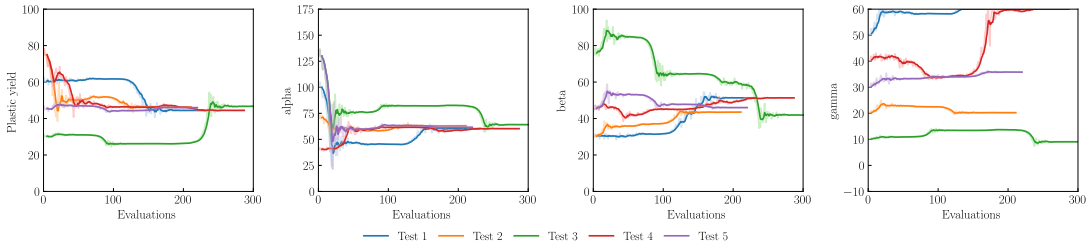
(b) RMSE evolution.

Figure 4.6: Results of validation tests with 6to3 dataset.

To verify our algorithm independent of the used target data set, we made the same tests with two additional data sets. In the following we present the results of studies for mixing ratio 4:3 and 8:3.



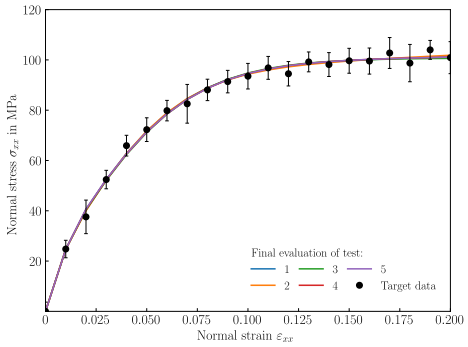
(a) Evolution of material 4:3.



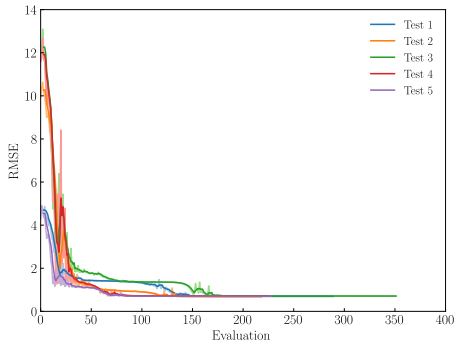
(b) Evolution of 8:3.

Figure 4.7: Evolution of material parameters for (a) mixing ratio 4:3 and (b) mixing ratio 8:3..

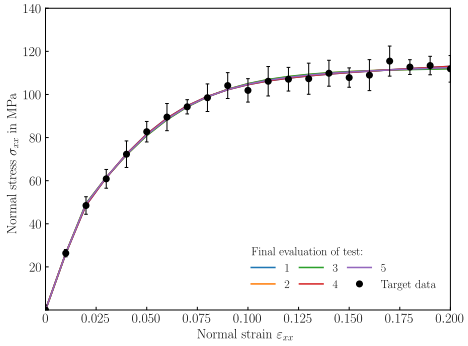
The evolution of the plastic material parameters for the mixing ratios 4:3 and 8:3 in plotted in **??**. We can observe a similar convergence behaviour as in the validation study with mixing ratio 6:3. Only test case 3 for mixing ratio 8:3 shows a deviation provided that all values converge quite late. Since we chose the initial values randomly this might occur through an unfavourable combination of values. In **??** we represent the optimized stress-strain curves and the evolution of the RMSE. For all tests the stress-strain values correlate with the target data. The equivalent level of the RMSE for the converged solutions indicate a similar quality of the optimization result for all tests. These results indicate an improvement of the solution results through determining the elastic parameters.



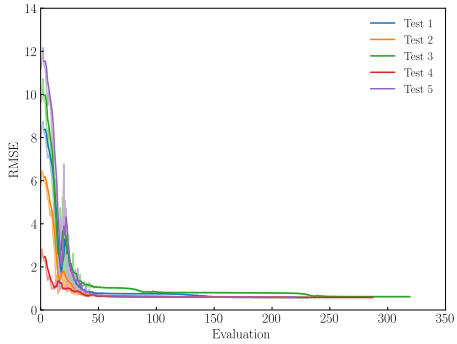
(a) Stress–strain curves (4:3).



(b) RMSE evolution (4:3).



(c) Stress–strain curves (8:3).



(d) RMSE evolution (8:3).

Figure 4.8: Results of validation tests with mixing ratios 4:3 and 8:3: (a,b) show the final stress–strain curves and RMSE evolution for mixing ratio 4:3; (c,d) show the corresponding results for mixing ratio 8:3..

Overall these results demonstrate the reliability of the optimization algorithm for the load case of a single tensile strain in one direction with fixed elastic parameters. The specification of the elastic parameter values improves the optimization performance in a way that for the plastic yield, alpha and beta independent of the initial values a singular solution can be found. However, the manual specification of Youngs modulus and Poisson ratio is only possible for target data sets with exactly one data point in the elastic domain of the material. For data sets with multiple points in the elastic domain a manual specification becomes complicated quite fast. Additional, for materials with completely unknown material behaviour, the point of transition between elastic and plastic behaviour is still unknown. In order to process such data sets too, we need to integrate the elastic parameters in the optimization process. In doing so the singularity of the solution should be maintained. Therefore the algorithm needs additional information about the mechanical material behaviour. In the following step we tried to do so through the combination of two load cases. Additional to the already tested tensile strain we applied a shear strain. As described in section XX the shear modulus contains information about Youngs modulus and Poisson ratio what might improve the performance of the optimizatoin process. The information contained within the shear modulus about Youngs modulus and Poisson ratio might enclose the necessary restrictions to reduce the solution variability.

Warum sinusförmige belastung? Jz schon mit zyklischen versuchen kommen?

4.3 shear and normal strain tests combined

4.4 cyclic tests

5 Conclusion

Bibliography

[1] M. Ries. "Mechanical behavior of adhesive joints: A review on modeling techniques". In: *Computer Methods in Materials Science* 24.4 (2024). doi: 10.7494/cmms.2024.4.1010. url: https://www.cmms.agh.edu.pl/2024_4_1010/ (visited on 10/06/2025).

[2] S. Gorbunov, A. Volkov, & R. Voronkov. "Periodic boundary conditions effects on atomic dynamics analysis". In: *Computer Physics Communications* 279 (Oct. 2022), p. 108454. doi: 10.1016/j.cpc.2022.108454. url: https://linkinghub.elsevier.com/retrieve/pii/S0010465522001734 (visited on 10/06/2025).

[3] O. Büyüköztürk et al. "Structural solution using molecular dynamics: Fundamentals and a case study of epoxy-silica interface". In: *International Journal of Solids and Structures* 48.14 (July 2011). Publisher: Elsevier BV, pp. 2131–2140. doi: 10.1016/j.ijsolstr.2011.03.018. url: https://linkinghub.elsevier.com/retrieve/pii/S002076831100120X (visited on 07/24/2025).

[4] J. Mergheim. "Lecture Notes Materials Modelling and Simulation". In: ().

[5] M. Ries et al. "DECIPHERING ELASTOPLASTIC PROPERTIES FROM ATOMISTIC STRUCTURE: REACTIVE COARSE-GRAINED MD FOR EPOXIES". In: ().

[6] K. Willner. "Vorlesungsskript Methode der finiten Elemente". Vorlesungsskript. Vorlesungsskript. Erlangen. (Visited on 10/09/2025).

[7] V. Jagota, A. P. S. Sethi, & K. Kumar. "Finite Element Method: An Overview". In: *Finite Element Method* ().

[8] F. Gao & L. Han. "Implementing the Nelder-Mead simplex algorithm with adaptive parameters". In: *Computational Optimization and Applications* 51.1 (Jan. 2012). Publisher: Springer Science and Business Media LLC, pp. 259–277. doi: 10.1007/s10589-010-9329-3. url: http://link.springer.com/10.1007/s10589-010-9329-3 (visited on 07/24/2025).

[9] N. Pham, A. Malinowski, & T. Bartczak. "Comparative Study of Derivative Free Optimization Algorithms". In: *IEEE Transactions on Industrial Informatics* 7.4 (Nov. 2011), pp. 592–600. doi: 10.1109/TII.2011.2166799. url: https://ieeexplore.ieee.org/document/6011694/ (visited on 10/04/2025).

[10] S. Singer & S. Singer. "Efficient Implementation of the Nelder–Mead Search Algorithm". In: *Applied Numerical Analysis & Computational Mathematics* 1.2 (2004). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/anac.200410015, pp. 524–534. doi: 10.1002/anac.200410015. url: https://onlinelibrary.wiley.com/doi/abs/10.1002/anac.200410015 (visited on 10/04/2025).

[11] J. A. Nelder & R. Mead. "A Simplex Method for Function Minimization". In: *The Computer Journal* 7.4 (Jan. 1, 1965). Publisher: Oxford University Press (OUP), pp. 308–313. doi: 10.1093/comjnl/7.4.308. url: https://academic.oup.com/comjnl/article-lookup/doi/10.1093/comjnl/7.4.308 (visited on 07/26/2025).

[12] M. Baudin. "Nelder-Mead User's Manual". In: ().

[13] D. P. Kraj & D. M. Šumarac. "DAMAGE MECHANICS - BASIC PRINCIPLES". In: ().

[14] S. Fear. *Publication quality tables in Latex – the booktabs package*. 2016.

[15] S. Pfaller. "Multiscale Simulation of Polymers – Coupling of Continuum Mechanics and Particle-Based Simulation". In: *Schriftenreihe Technische Mechanik*. Vol. 16. ISSN: 2190-023X. Universität Erlangen-Nürnberg: Lehrstuhl für Technische Mechaink - Universität Erlangen-Nürnberg, 2015.

[16] M. A. Luersen & R. Le Riche. "Globalized Nelder–Mead method for engineering optimization". In: *Computers & Structures* 82.23 (Sept. 2004). Publisher: Elsevier BV, pp. 2251–2260. doi: 10.1016/j.compstruc.2004.03.072. url: https://linkinghub.elsevier.com/retrieve/pii/S0045794904002378 (visited on 07/24/2025).

[17] B. S. Anglin, B. T. Gockel, & A. D. Rollett. "Developing constitutive model parameters via a multi-scale approach". In: *Integrating Materials and Manufacturing Innovation* 5.1 (Dec. 2016). Publisher: Springer Science and Business Media LLC, pp. 212–231. doi: 10.1186/s40192-016-0053-4. url: http://link.springer.com/10.1186/s40192-016-0053-4 (visited on 07/24/2025).

[18] N. O. Garifullin et al. "Dependence of the Physical-Mechanical Properties of Cured Epoxy-Amine Resin on the Ratio of its Components". In: *Key Engineering Materials* 816 (Aug. 2019), pp. 146–150. doi: 10.4028/www.scientific.net/KEM.816.146. url: https://www.scientific.net/KEM.816.146 (visited on 10/03/2025).

[19] A. Roche, P. Dole, & M. Bouzziri. "Measurement of the practical adhesion of paint coatings to metallic sheets by the pull-off and three-point flexure tests". In: *Journal of Adhesion Science and Technology* 8.6 (Jan. 1994), pp. 587–609. doi: 10.1163/156856194X00366. url: http://www.tandfonline.com/doi/abs/10.1163/156856194X00366 (visited on 10/03/2025).

[20] V. Dötschel et al. "Reactive coarse-grained MD models to capture interphase formation in epoxy-based structural adhesive joints". In: *European Journal of Mechanics - A/Solids* 116 (Mar. 2026), p. 105801. doi: 10.1016/j.euromechsol.2025.105801. url: https://linkinghub.elsevier.com/retrieve/pii/S0997753825002359 (visited on 10/03/2025).

[21] R. Schwerz & M. Roellig. "Non-linear viscoelastic Material Models of Polymers for Electronics Simulation - Measurement, Modelling, Validation". In: *2024 25th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*. 2024 25th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE). Catania, Italy: IEEE, Apr. 7, 2024, pp. 1–7. doi: 10.1109/EuroSimE60745.2024.10491560. url: https://ieeexplore.ieee.org/document/10491560/ (visited on 10/03/2025).

[22] J.-H. Yi & S. Mun. "Backcalculating pavement structural properties using a Nelder–Mead simplex search". In: *International Journal for Numerical and Analytical Methods in Geomechanics* 33.11 (2009). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nag.769, pp. 1389–1406. doi: 10.1002/nag.769. url: https://onlinelibrary.wiley.com/doi/abs/10.1002/nag.769 (visited on 10/04/2025).

[23] M. Luersen, R. Le Riche, & F. Guyon. "A constrained, globalized, and bounded Nelder–Mead method for engineering optimization". In: *Structural and Multidisciplinary Optimization* 27.1 (May 1, 2004), pp. 43–54. doi: 10.1007/s00158-003-0320-9. url: https://doi.org/10.1007/s00158-003-0320-9 (visited on 10/04/2025).

[24] D. M. Olsson. "A Sequential Simplex Program for Solving Minimization Problems". In: *Journal of Quality Technology* 6.1 (Jan. 1974), pp. 53–57. doi: 10.1080/00224065.1974.11980616. url: https://www.tandfonline.com/doi/full/10.1080/00224065.1974.11980616 (visited on 10/04/2025).

[25] S. Wessing. "Proper initialization is crucial for the Nelder–Mead simplex search". In: *Optimization Letters* 13.4 (June 2019), pp. 847–856. doi: 10.1007/s11590-018-1284-4. url: http://link.springer.com/10.1007/s11590-018-1284-4 (visited on 10/04/2025).

[26] K. Klein & J. Neira. "Nelder-Mead Simplex Optimization Routine for Large-Scale Problems: A Distributed Memory Implementation". In: *Computational Economics* 43.4 (Apr. 2014), pp. 447–461. doi: 10.1007/s10614-013-9377-8. url: http://link.springer.com/10.1007/s10614-013-9377-8 (visited on 10/06/2025).

[27] R. Hill. "Elastic properties of reinforced solids: Some theoretical principles". In: *Journal of the Mechanics and Physics of Solids* 11.5 (Sept. 1963), pp. 357–372. doi: 10.1016/0022-5096(63)90036-X. url: https://linkinghub.elsevier.com/retrieve/pii/002250966390036X (visited on 10/06/2025).

[28] "Abaqus Analysis User's Guide, vol3". In: ().

[29] *Finite-Elemente-Methode*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. doi: 10.1007/978-3-540-72236-6. url: http://link.springer.com/10.1007/978-3-540-72236-6 (visited on 10/09/2025).

[30] K. Knothe & H. Wessels. *Finite Elemente*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. doi: 10.1007/978-3-662-49352-6. url: http://link.springer.com/10.1007/978-3-662-49352-6 (visited on 10/09/2025).

[31] Voce. "A Practical Strain-Hardening Function". In: Metallurgia (1948).

A Appendix

A.1 Section

B Appendix