Karlsruher Institut für Technologie

DLR

Master's Thesis

Prediction of Mechanical Properties for tape-layered Specimen using Finite Element Method and Machine Learning

Carolin Lupprian

01.09.2025

Supervisor: Prof. Dr. Christian Wieners

Department of Mathematics
Karlsruhe Institute of Technology

Abstract

The production of fibre composite components using the Automated Fibre Placement (AFP) process is often confronted by manufacturing-related defects, such as gaps or overlaps between tapes. These defects can influence the material properties in different ways, depending on their characteristics. While the Finite Element (FE) Method can be used to investigate these effects, it requires expert knowledge and significant computational resources. The aim of this work is therefore to predict the mechanical properties using machine learning (ML) methods trained on compression tests simulated with LS-DYNA. First, an FE modelling approach and material model are developed with the goal of minimising both computation time and deviation from experimental test data. Subsequently, 1000 FE models are generated with random defect configurations by varying parameters such as position, size, and number of defects with the help of a developed Python routine. Two different ML methods, namely Random Forest (RF) and Support Vector Regression (SVR), are then applied and further improved in terms of their performance. Both models achieved promising results, with stiffness and strength predictions reaching a Mean Absolute Percentage Error (MAPE) of 3 to 7%. However, the prediction of post-failure behaviour showed limited accuracy, with a MAPE of around 50% due to numerical issues.

Contents

Symbols

$\boldsymbol{a}$	Acceleration
b	Bias in H in SVR
$\boldsymbol{b}$	Body forces
B	Number of trees in a RF
C	Constant for slack variables in SVR
$\boldsymbol{C}$	Green deformation tensor, damping matrix
$\mathcal{D}$	Dataset
$\{e_1, e_2, e_3\}$	Orthonormal basis of $\mathbb{R}^3$
e_c, e_d	Failure modes for MAT_54
$\boldsymbol{E}$	Material strain tensor
E, E_1, E_2	Young's modulus
$\boldsymbol{f}$	Body force density
$\boldsymbol{F}, \boldsymbol{F_e}$	Deformation gradient
G_{12}	Shear modulus
H	Mapping from $\mathbf{x}$ to y
K	Kernel
K_e	Finite element
$\hat{K}$	Reference element
$\boldsymbol{M}$	Mass matrix
$\boldsymbol{n}$	Unit normal vector
N	Number of samples in $\mathcal{D}$
N_I	Shape function
p	Number of features
$\boldsymbol{P}$	Time dependent prescribed loads
$\boldsymbol{R}$	Vector of internal forces
$\boldsymbol{S}$	Compliance matrix
t	Time
$\boldsymbol{t}$	Force per unit area
T	End time simulation
$\boldsymbol{u}$	Displacement
$\boldsymbol{v}$	(spatial) Velocity
V	Relative Volume
$\boldsymbol{V}$	Material velocity
$\boldsymbol{w}$	Weight vector in H in SVR
$\boldsymbol{x}$	Spatial points
$\mathbf{x}, \mathbf{x}_i$	Features
$\boldsymbol{X}$	Material points
y, y_i	Target variable
$\hat{\mathrm{y}}, \hat{\mathrm{y}}_i$	Predicted value for target variable
α_i, α_i^*	Lagrange multipliers
γ	Shear strain, parameter for RBF kernel
ν_{12}, ν_{21}	Poisson's ratio
$\boldsymbol{\xi}$	Coordinates of reference element
ρ	Density
Δt	Time step
$\Delta t_{\max}$	Maximum time step for implicit time integration

ϵ	Strain, error margin for SVR
$\boldsymbol{\epsilon}$	Linearized strain tensor
$\boldsymbol{\zeta}, \boldsymbol{\zeta}^*$	Slack variables
$\boldsymbol{\sigma}$	Cauchy stress tensor
τ	Shear stress
ϕ	Configuration or deformation, mapping of $\mathbf{x}$
$\boldsymbol{\psi}$	Test function
Ω	Body, set in $\mathbb{R}^3$
$\mathcal{C}, \mathcal{C}(\Omega)$	Set of all configurations of Ω
EA	MAT_54: Longitudinal elastic modulus
EFS	MAT_54: Effective Failure Strain
SC	MAT_54: Shear Strength
$SLIMC1$	MAT_54: Factor for minimum stress (fibre compression)
$SLIMC2$	MAT_54: Factor for minimum stress (matrix compression)
$SLIMS$	MAT_54: Factor for minimum stress (shear)
XC	MAT_54: Longitudinal compressive strength
YC	MAT_54: Transverse compressive strength
$YCFAC$	MAT_54: Strength reduction factor

List of Acronyms

AFP Automated Fibre Placement

CFRP Carbon Fibre Reinforced Plastics

CV Cross-validation

FE Finite Element

FI Feature Importance

IF Isolation Forest

LOF Local Outlier Factor

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

ML Machine Learning

RBF Radial Basis Function

RF Random Forest

SVR Support Vector Regression

UD Unidirectional

1 Introduction

Automated Fibre Placement (AFP) is an advanced manufacturing method for Carbon Fibre Reinforced Plastics (CFRP) aerospace components. This technique offers great potential in reducing manufacturing time and costs. However, complex geometries can lead to gaps or overlaps between the tapes, which are not corrected during the manufacturing process. These defects can significantly impact the mechanical properties of the component [1]. Therefore, a method is needed to quickly assess whether a component with a defect still meets quality requirements and to predict all the mechanical properties.

One approach is the use of Finite Element (FE) simulations, but they are too time-consuming in this case. First, a suitable simulation model with a mesh representing the sample and a validated material model has to be set up. Then, the model must be simulated at best on a workstation with high computing power, followed by post-processing to obtain the desired mechanical properties. All these steps take time and expertise in FE simulation. Additionally, access to the workstation is not always available during production, which would increase the simulation time further. Therefore, a different method is needed for predicting the mechanical properties in real-time.

An alternative approach is the use of Machine Learning (ML) methods. These algorithms can analyse and determine complex non-linear relationships between input properties based on existing information. Once trained, ML models provide real-time results with low computational power requirements. Unlike FE simulations, they require no expert knowledge and can be integrated into a user-friendly software [2], [3]. The ML models can be trained with experimental test data, but they are expensive and labour intensive to acquire. Additionally, experimental data is prone to errors arising from testing inaccuracies or machine related issues. Training the ML model with simulation data is another method, which offers advantages such as reduced material consumption and equipment requirements.

The objective of this thesis is to develop a ML model that can predict the mechanical properties of tape-layered samples with defects based on compression tests. Compression properties are critical design parameters for CFRP components because these components must withstand crash and impact events [4]. The ML model can serve as the basis for a manufacturing tool that quickly determines whether a component with defects is still usable or not. The ML model will be trained on simulation results, enabling the creation of a larger and more comprehensive dataset. The simulation models must be robust, validated and computationally efficient to generate sufficient and high quality data in the limited time span of this work. Random Forest (RF) and Support Vector Regression (SVR) models are chosen because of their promising results in previous studies, as will be discussed in Section 2.6. Feature engineering and hyperparameter optimization will be employed to further enhance model performance.

The thesis begins by introducing the necessary theoretical fundamentals for AFP (Section 2.1), FE simulation (Section 2.2) and ML (Section 2.5). Then, the development of the simulation model in LS-DYNA is described, involving the set up of a implicit simulation and the optimization of the material model to experimental results. Afterwards, the process of generating the dataset from simulation results for various specimens is depicted, which is analysed for outliers or unusual behaviour. The subsequent section

covers the development, training and evaluation of the RF and SVR models. It involves feature engineering and a hyperparameter optimization to improve the performances of the models. The final section will apply these models to unknown datasets and conduct a short study to investigate the impact of different defects on sample behaviour.

2 Fundamentals

In this section, the fundamental concepts relevant to this thesis are presented. They provide background information necessary to establish a simulation in LS-DYNA for tape-layered specimens and for developing a ML model for a regression problem to predict the mechanical properties of the samples.

2.1 In-situ Automated Fibre Placement and Defects

AFP is an additive manufacturing process for producing large-scale and lightweight composite structures made from CFRP. It is commonly used in the aerospace industry. During this process, a robot-guided placement head lays narrow Unidirectional (UD) prepreg tapes into a mould of the desired geometry, making it a fast production method for composite components. Figure 2.1 illustrates the placement head for AFP manufacturing. Additionally, AFP has minimal material waste because each tape can be cut individually and it is flexible regarding fibre orientation along load direction. Traditional manufacturing methods typically require parts to be cured using an autoclave. However, thermoplastic in-situ AFP skips this as it enables the production of components in a single step by integrating in-line quality assurance methods [4], [5].

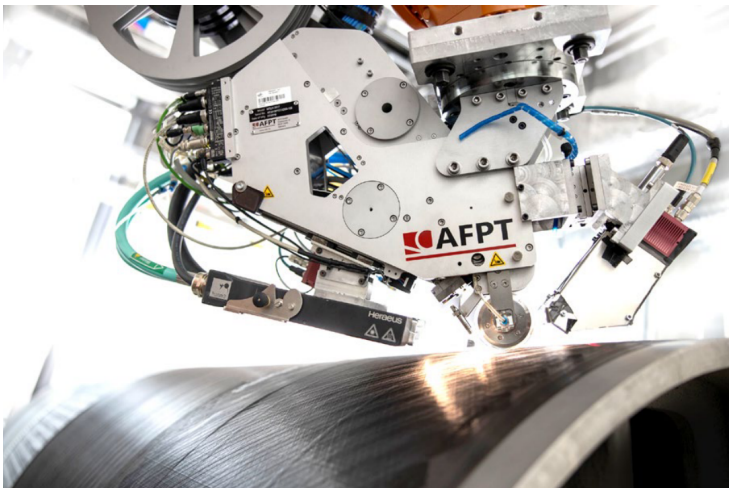


Figure 2.1: Placement head for AFP [6]

One challenge of in-situ AFP is the production of parts with complex shapes. The curvatur of the surface can lead to defects in AFP-manufactured structures, which are not corrected during production and cannot be mitigated by subsequent autoclave or press-consolidation processes. Therefore, Sacco et al. [7] have already employed a ML model for AFP manufacturing to identify and classify defects based on vision. During production, this method is used for quality assessment. The three independent factors fibre angle deviation, fibre steering and gap or overlap defects determine the type and magnitude of the defects [8].

The effects of defects in AFP-manufactured components have been studied by various researches. For example, Raps et al. [8] conducted tensile and compression tests for AFP laminates with gap defects. The results showed that the tensile strength remained unaffected, while the compression strength decreased for the specimens with gap defects. In a subsequent study, Raps et al. [1] successfully mitigated the effects of gap defects on

the compressive strength by filling the gaps with fused granular fabrication. Other studies have shown that overlaps do not have a significant impact on strength and stiffness in tension and compression tests [4], [5]. The defects can have a rectangular or a triangular geometry, which occur in double-curved parts. Triangular defects also induce a fibre angle deviation [1]. Figure 2.2 displays the triangular and rectangular defects with gaps and overlaps.

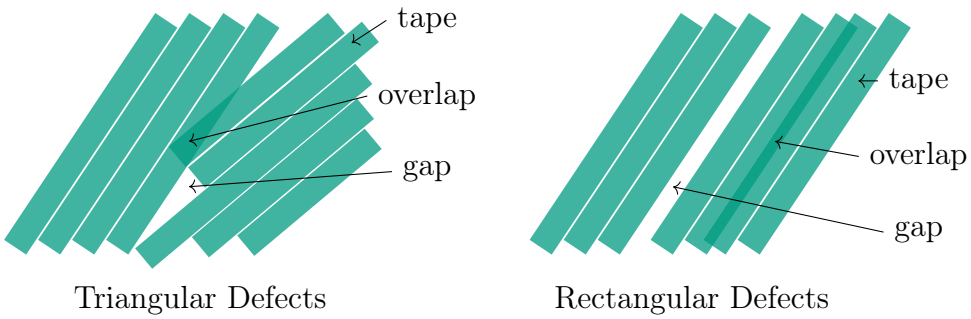


Figure 2.2: Illustration of the different defects (adapted from [6])

2.2 Mathematical Foundations

In this section, the necessary mathematical foundations are presented for establishing a structural-mechanical problem and solving it using the FE method.

Definition 2.1 ([9]). *A **tensor** $\boldsymbol{T}$ is a linear map between two vector spaces $\mathcal{V}$ and $\mathcal{W}$ with*

$$\begin{aligned} \boldsymbol{T} : \mathcal{V} &\to \mathcal{W}, \\ \boldsymbol{v} &\mapsto \boldsymbol{w} = \boldsymbol{T}\boldsymbol{v}, \quad \boldsymbol{v} \in \mathcal{V},\ \boldsymbol{w} \in \mathcal{W}, \\ \boldsymbol{T}(\boldsymbol{u} + \boldsymbol{v}) &= \boldsymbol{T}\boldsymbol{u} + \boldsymbol{T}\boldsymbol{v}, \\ \boldsymbol{T}(\alpha\boldsymbol{u}) &= \alpha\boldsymbol{T}\boldsymbol{u}. \end{aligned}$$

A special tensor is the dyad consisting of vectors defined in $\mathcal{V}$ and $\mathcal{W}$

$$\boldsymbol{T} = \boldsymbol{a} \otimes \boldsymbol{b}, \qquad \boldsymbol{a} \in \mathcal{W}, \boldsymbol{b} \in \mathcal{V}.$$

2.2.1 Kinematics of Bodies

In the context of this problem, the Lagrangian description is employed, and we assume a three-dimensional Euclidean space denoted as $\mathbb{R}^3$, with an origin o and an orthonormal basis $\{e_1, e_2, e_3\}$. We consider a body $\Omega \subset \mathbb{R}^3$, where Ω is the closure of an open set in $\mathbb{R}^3$ with piecewise smooth boundary. The points within Ω are referred to as material points and denoted as $\boldsymbol{X} = (X_1, X_2, X_3) \in \Omega$. In contrast, points in $\mathbb{R}^3$ are denoted as $\boldsymbol{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$, and are named spatial points [10], [11]. The relationship between the following definitions is illustrated in Figure 2.3.

Definition 2.2 ([10]). *Let $\Omega \subset \mathbb{R}^3$ be a body. A **configuration** or a **deformation** of Ω is a mapping $\phi : \Omega \to \mathbb{R}^3$. The set of all configurations of Ω is denoted $\mathcal{C}(\Omega)$ or $\mathcal{C}$. It follows $\boldsymbol{x} = \phi(\boldsymbol{X})$.*

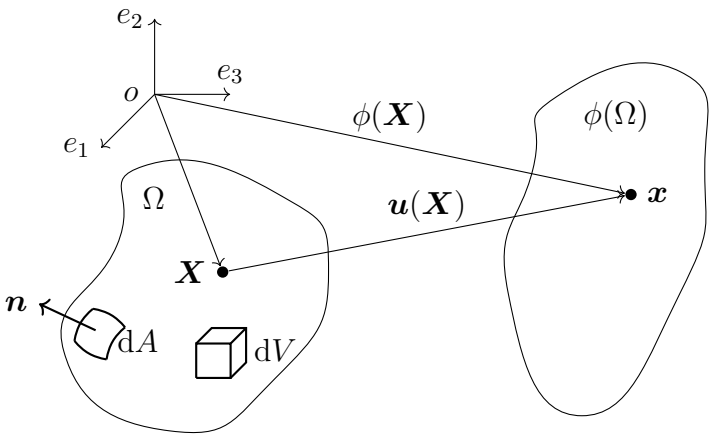


Figure 2.3: Geometry of a deformation (adapted from [11])

Definition 2.3 ([10]). *The **motion** of body Ω is defined as a curve in $\mathcal{C}$, which is a time-dependent family of configurations. It is represented by a mapping $t \to \phi_t \in \mathcal{C}$ of $\mathbb{R}$ to $\mathcal{C}$. We write $\boldsymbol{x} = \phi(\boldsymbol{X}, t)$.*

Definition 2.4 ([10]). *The **material velocity** $\boldsymbol{V}_t : \Omega \to \mathbb{R}^3$ of the motion is defined by*

$$\boldsymbol{V}_t(\boldsymbol{X}) = \boldsymbol{V}(\boldsymbol{X}, t) = \frac{\partial \phi(\boldsymbol{X}, t)}{\partial t}.$$

*If ϕ_t is a C^1 regular motion, the **spatial velocity** $\boldsymbol{v} : \phi_t(\Omega) \to \mathbb{R}^3$ of the motion is given by*

$$\boldsymbol{v}_t = \boldsymbol{V}_t \circ \phi_t^{-1}.$$

Definition 2.5 ([10]). *The vector field $\boldsymbol{u} : \Omega \times [0, T] \to \mathbb{R}^3$ on Ω is called the **displacement** and defined by*

$$\boldsymbol{u}(\boldsymbol{X}, t) = \phi(\boldsymbol{X}, t) - \boldsymbol{X}.$$

Definition 2.6 ([10]). *The **deformation gradient** $\boldsymbol{F}$ of ϕ is a two point tensor. It is given by*

$$\boldsymbol{F} = D\phi, \qquad F_{ij} = \frac{\partial \phi_i}{\partial X_i}.$$

Definition 2.7 ([10]). *The **(Green) deformation tensor** or right Cauchy-Green tensor $\boldsymbol{C}$ is denoted as*

$$\boldsymbol{C} = \boldsymbol{F}^\top \boldsymbol{F}.$$

Definition 2.8 ([10]). *The **material strain tensor** $\boldsymbol{E}$ is defined as*

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}).$$

If $\boldsymbol{E} = 0$, which implies $\boldsymbol{C} = \boldsymbol{I}$, then it follows that points in Ω do not experience relative motion under the mapping ϕ.

The strain tensor $\boldsymbol{E}$ can be expressed as the derivative of the displacement tensor $\boldsymbol{u}$, given by the equation

$$\boldsymbol{E} = \frac{1}{2}(D\boldsymbol{u} + (D\boldsymbol{u})^\top + (D\boldsymbol{u})^\top D\boldsymbol{u}). \tag{2.1}$$

For small deformation, it is often assumed that $\phi(\boldsymbol{X}, t) \approx 0$, which implies $D\boldsymbol{u}(\boldsymbol{x}, t) \approx 0$. In this case the formulation (2.1) of the strain tensor can be linearized to

$$\boldsymbol{E} \approx \frac{1}{2}(D\boldsymbol{u} + (D\boldsymbol{u})^\top) =: \boldsymbol{\epsilon}. \tag{2.2}$$

The linearized strain tensor $\boldsymbol{\epsilon}$ is a fundamental concept in classical theory of elasticity and will be addressed in the linear case in Section 2.3 [9].

Let $\boldsymbol{b}(x, t)$ be the body forces that act on Ω. The vector field $\boldsymbol{t}(\boldsymbol{x}, t, n)$ represents the force per unit area at position x and time t across a surface element with unit normal $\boldsymbol{n}$. According to Newton's second law, the continuum analogue states that for any subbody $\mathcal{U} \subset \Omega$

$$\frac{d}{dt}\int_{\mathcal{U}_t} \rho\boldsymbol{v}\ \mathrm{d}x = \int_{\partial\mathcal{U}_t} \boldsymbol{t}(\boldsymbol{x}, t, \boldsymbol{n})\ \mathrm{d}a + \int_{\mathcal{U}_t} \rho\boldsymbol{b}\ \mathrm{d}x. \tag{2.3}$$

This assertion is known as the balance of (linear) momentum [10].

Theorem 2.9 (Cauchy's theorem,[10])**.** *The balance of momentum (2.3) holds, and $\phi(\boldsymbol{X}, t)$ is a C^1 function, while $\boldsymbol{t}(\boldsymbol{x}, t, \boldsymbol{n})$ is a continuous function of its arguments. Under these conditions, there exists a unique $\binom{2}{0}$ tensor field $\boldsymbol{\sigma}$, depending on x and t, such that*

$$\boldsymbol{t}(\boldsymbol{x}, t, \boldsymbol{n}) = \langle \boldsymbol{\sigma}(\boldsymbol{x}, t), \boldsymbol{n} \rangle$$

In component form, $\boldsymbol{\sigma}(\boldsymbol{x}, t)$ is a 3×3 matrix with components σ_{ij} and $\boldsymbol{t}$ is a vector with components t_i such that $t_i = \sum_j \sigma_{ij} n_j$. We call $\boldsymbol{\sigma}$ the Cauchy stress tensor.

Proof. The proof is given in Theorem 2.2 in [10]. □

2.2.2 Initial Problem

Consider a body Ω. The objective is to determine the time-dependent deformation of a point within Ω, initially located at $\boldsymbol{X}$, which moves in a fixed rectangular Cartesian coordinate system to a point $\boldsymbol{x}$ in the same coordinate system. The goal is to find a solution to the momentum equation

$$\nabla \cdot \boldsymbol{\sigma} + \rho\boldsymbol{f} = \rho\ddot{\boldsymbol{u}} \tag{2.4}$$

where $\boldsymbol{u}$ is the displacement, $\ddot{\boldsymbol{u}} = \frac{\partial^2 \boldsymbol{u}}{\partial t^2}$ denotes the acceleration, $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{f}$ is the body force density, and ρ is the current density. The parameter $\boldsymbol{f}$ represents the external forces acting on the body Ω. The boundary conditions for this problem are

$$\boldsymbol{u}(\boldsymbol{X}, t) = \boldsymbol{D}(t) \qquad \text{on } \Gamma_D, \tag{2.5}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{t}(t) \qquad \text{on } \Gamma_N, \tag{2.6}$$

$$(\boldsymbol{\sigma}^+ - \boldsymbol{\sigma}^-) \cdot \boldsymbol{n} = 0 \qquad \text{on interface } \Gamma_I. \tag{2.7}$$

Multiplying the momentum equation by a test function $\boldsymbol{\psi}$ and integrating over Ω leads to

$$\int_{\Omega} (\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{f} - \rho \ddot{\boldsymbol{u}}) \cdot \boldsymbol{\psi} \, \mathrm{d}\Omega = 0. \tag{2.8}$$

From this, the weak formulation of the momentum equation can be derived, satisfying all boundary conditions

$$\int_{\Omega} \boldsymbol{\sigma} : \nabla \boldsymbol{\psi} \, \mathrm{d}V + \int_{\Omega} \rho \ddot{\boldsymbol{u}} \cdot \boldsymbol{\psi} \, \mathrm{d}V = \int_{\Omega} \rho f \cdot \boldsymbol{\psi} \, \mathrm{d}V + \int_{\Gamma_N} \boldsymbol{t}(t) \cdot \boldsymbol{\psi} \, \mathrm{d}A \tag{2.9}$$

where $\boldsymbol{\sigma} : \nabla \psi = \sum_{i,j} \sigma_{ij} \frac{\partial \psi_i}{\partial x_j}$ [12].

2.2.3 Spatial Discretization with Isoparametric Finite Elements

One method for solving the weak formulation (2.9) is to discretize the geometry of body Ω in its initial configuration using a FE mesh consisting of n_e FEs, $K_e \subset \Omega_h$, such that

$$\Omega \approx \Omega^h = \bigcup_{e=1}^{n_e} K_e. \tag{2.10}$$

An overlapping of the FEs is not allowed, and the assembled elements must be continuous in Ω. To approximate the primary field variables, which is in the case of equation (2.9), the displacement $\boldsymbol{u}$, an interpolation function must be selected. The exact solution for the weak formulation (2.9) can then be approximated within one FE by

$$\boldsymbol{u}_{exact}(\boldsymbol{X}) \approx \boldsymbol{u}_h(\boldsymbol{X}) = \sum_{I=1}^{n} N_I(\boldsymbol{X}) \boldsymbol{u}_I. \tag{2.11}$$

where $N_I(\boldsymbol{X})$ are the shape functions defined in K_e, and $\boldsymbol{u}_I$ represents the unknown nodal quantities. The approximation u_h must be chosen such that it converges to the true solution u_{exact} of the underlying partial differential equation (2.4). One method for constructing shape or interpolation functions is the isoparamtric scheme, which is a widely used and popular approach for engineering problems. In this method, both geometry and variables are interpolated using the same shape function via a reference element $\hat{K}$. This implies that the initial $\boldsymbol{X}$ and spatial configuration $\boldsymbol{x}$ are interpolated by the same shape function N_I. Within one finite element, we have

$$\boldsymbol{X}_e = \sum_{I=1}^{n} N_I(\boldsymbol{\xi}) \boldsymbol{X}_I, \qquad \boldsymbol{x}_e = \sum_{I=1}^{n} N_I(\boldsymbol{\xi}) \boldsymbol{x}_I \tag{2.12}$$

Typically, N_I is a polynomial function. The shape functions within a FE in its initial configuration K_e are now replaced in equation (2.12) by shape functions $N_I(\boldsymbol{\xi})$, which are defined within the reference element $\hat{K}$. Consequently, a transformation (2.12) for each element K_e must be performed, which relates $\boldsymbol{X}_e = \boldsymbol{X}_e(\boldsymbol{\xi})$ to the coordinates $\boldsymbol{\xi}$ of the reference element $\hat{K}$ [9].

Figure 2.4 illustrates the transformation process. To perform this process, an approximate deformation map ϕ_e of the deformation ϕ from Definition 2.2 is utilized to map an element of the initial configuration K_e to its current configuration $\phi(K_e)$. The deformation gradient of ϕ_e, denoted here as $\boldsymbol{F}_e$ (see Definition 2.6), is also required for the mapping.

It follows that

$$\boldsymbol{F}_e = \boldsymbol{j}_e \boldsymbol{J}_e^{-1} \tag{2.13}$$

with

$$\boldsymbol{j}_e = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}, \qquad \boldsymbol{J}_e = \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{\xi}}. \tag{2.14}$$

The equation (2.13) shows that the deformation gradient is defined by the isoparametric mapping from $\hat{K}$ to the initial configurations K_e and to the current configuration $\phi(K_e)$ [9].

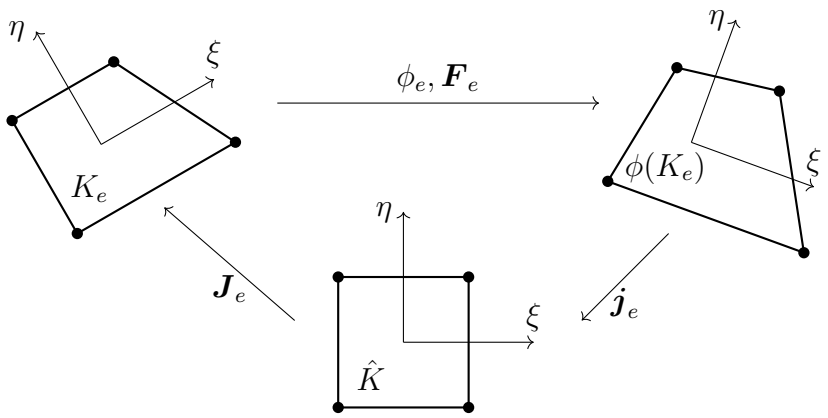


Figure 2.4: Isoparametric mapping of the deformation of a FE K_e (adapted from [9])

2.2.4 Time Integration Methods

After discretizing the weak formulation (2.9) with the FE method, the equation can be transformed into the general equation of motion with N unknowns

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{C}\dot{\boldsymbol{u}} + \boldsymbol{R}(\boldsymbol{u}) = \boldsymbol{P} \qquad \forall \boldsymbol{u} \in \mathbb{R}^N, \tag{2.15}$$

where $\boldsymbol{M}$ denotes the mass matrix, $\boldsymbol{R}(u)$ is the vector of the internal forces (stress divergence), which depends on the deformation $\boldsymbol{u}$ and the stress $\boldsymbol{\sigma}(\boldsymbol{u})$. Note that $\boldsymbol{R}(u)$ contains all non-linearities present in this equation. The time-dependent prescribed loads are represented by $\boldsymbol{P}$, while $\boldsymbol{C}$ is the damping matrix. Equation (2.15) can be written As a first-order differential equation system by defining the independent variables $\dot{\boldsymbol{u}} = \boldsymbol{v}$ and $\ddot{\boldsymbol{u}} = \dot{\boldsymbol{v}}$ with

$$\dot{\boldsymbol{u}} = \boldsymbol{v} \tag{2.16}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{M}^{-1} \left(\boldsymbol{P} - \boldsymbol{C}\boldsymbol{v} - \boldsymbol{R}(\boldsymbol{u}) \right). \tag{2.17}$$

In this context, $\boldsymbol{a}$ denotes the acceleration $\ddot{\boldsymbol{u}}$ and $\boldsymbol{v}$ is the velocity $\dot{\boldsymbol{u}}$. The discretized equation of the linear momentum at time t_{n+1} is given by

$$\boldsymbol{M}\boldsymbol{a}_{n+1} + \boldsymbol{C}\boldsymbol{v}_{n+1} + \boldsymbol{R}(\boldsymbol{u}_{n+1}) = \boldsymbol{P}_{n+1}. \tag{2.18}$$

To define an initial value problem, the initial conditions at $t = t_0$ are denoted as

$$\boldsymbol{u}_0 = \overline{\boldsymbol{u}},$$
$$\boldsymbol{v}_0 = \overline{\boldsymbol{v}}.$$

In order to determine the time-dependent response of the deformation $\boldsymbol{u}(t)$ of (2.15), two options are available, which will be discussed in the following [9].

Explicit Time Integration

Explicit methods are straightforward to implement, as the solution at time t_{n+1} depends only on the values at time t_n. However, they have limitations when it comes to time step size due to a stabilization criterion. As a result, explicit methods often require small time steps, which makes them well-suited for engineering applications involving impact problems where high-frequency parts or shock waves occur [9].

The central difference scheme is a widely used method for solving equations of motion in solid mechanics and structural problems. This scheme approximates the velocities $\boldsymbol{v}$ and the accelerations $\boldsymbol{a}$ at a time t_n using

$$\boldsymbol{v}_n = \frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_{n-1}}{2\Delta t}, \tag{2.19}$$
$$\boldsymbol{a}_n = \frac{\boldsymbol{u}_{n+1} - 2\boldsymbol{u}_n + \boldsymbol{u}_{n-1}}{\Delta t^2}. \tag{2.20}$$

After inserting these relations into equation (2.15) at time, we obtain

$$\boldsymbol{M}(\boldsymbol{u}_{n+1} - 2\boldsymbol{u}_n + \boldsymbol{u}_{n-1}) + \frac{\Delta t}{2}\boldsymbol{C}(\boldsymbol{u}_{n+1} - \boldsymbol{u}_{n-1}) + \Delta t^2 \boldsymbol{R}(\boldsymbol{u}_n) = \Delta t^2 \boldsymbol{P}_n. \tag{2.21}$$

This equation (2.21) can be rearranged to solve for the unknown displacement $\boldsymbol{u}_{n+1}$ at time t_{n+1}

$$\left(\boldsymbol{M} + \frac{\Delta t}{2}\boldsymbol{C}\right)\boldsymbol{u}_{n+1} = \Delta t^2 \left(\boldsymbol{P}_n - \boldsymbol{R}\left(\boldsymbol{u}_n\right)\right) + \frac{\Delta t}{2}\boldsymbol{C}\boldsymbol{u}_{n-1} + \boldsymbol{M}(2\boldsymbol{u}_n - \boldsymbol{u}_{n-1}). \tag{2.22}$$

Since the mass matrix $\boldsymbol{M}$ and the damping matrix $\boldsymbol{C}$ are constant, a triangular decomposition can be used for the coefficient matrix $\boldsymbol{M} + \frac{\Delta t}{2}\boldsymbol{C}$, enabling efficient solution of equation (2.22).

As mentioned above, explicit methods are not unconditionally stable. The critical time step limit for non-linear problems is given by

$$\Delta t \leq \delta \frac{h}{c_L}, \tag{2.23}$$

where h is the size of the smallest element in the FE mesh and c_L is the velocity of a compression wave in a linear solid. It is defined by $c_L = 3K\frac{1-\nu}{\rho(1+\nu)}$ with modulus of compression K, Poisson's ratio ν and density ρ. The constant δ with $0.2 < \delta < 0.9$ serves as a reduction factor to be selected according to the non-linear properties of the problem [9].

Implicit Time Integration

Implicit methods replace the time derivatives with quantities depending on the last time step at t_n as well as on the still unknown quantities at time $t_{n+\alpha}$. This leads to a non-linear algebraic equation system at every time step. In contrast to explicit methods, implicit schemes can be designed to be unconditionally stable, meaning the time step size has no limit. This makes them particularly suitable for problems where the response of the dynamical system depends mainly on lower frequencies, such as simulating engine vibrations [9].

The most popular scheme for solving non-linear implicit dynamic problems is the Newmark method. As equation (2.18) is a function of $\boldsymbol{a}_{n+1}$, $\boldsymbol{v}_{n+1}$ and $\boldsymbol{u}_{n+1}$, the idea is to eliminate $\boldsymbol{a}_{n+1}$ and $\boldsymbol{v}_{n+1}$ with the following scheme, so that they are only dependent on the displacements $\boldsymbol{u}_{n+1}$

$$\boldsymbol{a}_{n+1} = \alpha_1(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n) - \alpha_2\boldsymbol{v}_n - \alpha_3\boldsymbol{a}_n \tag{2.24}$$
$$\boldsymbol{v}_{n+1} = \alpha_4(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n) + \alpha_5\boldsymbol{v}_n + \alpha_6\boldsymbol{a}_n \tag{2.25}$$

with the constants

$$\begin{aligned} &\alpha_1 = \tfrac{1}{\beta\Delta t^2}, \quad \alpha_2 = \tfrac{1}{\beta\Delta t}, \quad \alpha_3 = \tfrac{1-2\beta}{2\beta}, \\ &\alpha_4 = \tfrac{\gamma}{\beta\Delta t}, \quad \alpha_5 = \left(1 - \tfrac{\gamma}{\beta}\right), \quad \alpha_6 = \left(1 - \tfrac{\gamma}{2\beta}\right)\Delta t. \end{aligned}$$

Here, β and γ denote the free parameters of integration. When $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$, the method becomes the trapezoidal rule, which is energy conserving. However, if

$$\begin{aligned} \beta &> \frac{1}{4}\left(\frac{1}{2} + \gamma\right)^2, \\ \gamma &> \frac{1}{2}, \end{aligned}$$

then numerical damping is introduced into the solution, resulting in a loss of energy and momentum [9], [12].

After inserting the relations 2.24 and 2.25 in the linear momentum equation 2.18, we obtain the following non-linear algebraic equation system for the unknown displacements $\boldsymbol{u}_{n+1}$

$$\begin{aligned} \boldsymbol{G}(u_{n+1}) =& \boldsymbol{M}\left(\alpha_1\left(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n\right) - \alpha_2\boldsymbol{v}_n - \alpha_3\boldsymbol{a}_n\right) \\ &+ \boldsymbol{C}\left(\alpha_4\left(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n\right) + \alpha_5\boldsymbol{v}_n + \alpha_6\boldsymbol{a}_n\right) \\ &+ \boldsymbol{R}(\boldsymbol{u}_{n+1}) - \boldsymbol{P}_{n+1} = 0 \quad [9]. \end{aligned} \tag{2.26}$$

This non-linear system of equations can be solved with various methods, such as the full Newton schemes or quasi-Newton methods, which include the BFGS method or the Broyden method [12].

2.3 Fundamentals Fibre composites

Fibre composites are made from high-strength fibres that absorb applied forces and a matrix supporting and fixating the fibres in a predefined position. One category of fibre composites is CFRP, consisting of carbon fibres and a polymer matrix. Carbon fibres have several desirable properties, including their lightweight nature and high strength. They perform well not only under static load but also under fatigue loading conditions, making them well-suited for structures in aircraft construction. However, carbon fibres also have certain limitations. Damage caused by impact is often not visible to the naked eye. This includes fractures and delamination, which is the separation of the plies reducing the bending stiffness. Therefore, components at risk of impact, which applies to several structures in aircraft and vehicle construction, are tested with compression after impact test. Another limitation is the fibre-parallel compressive strength, which is lower than the tensile strength and sensitive to defects or deviations from the ideal state, such as fibre crimps. Hence, this strength factor is often the limitation for composite structures [13].

The polymer matrix in a CFRP laminate fixates and bonds the fibres, transferring forces into and between them. It also carries mechanical loads transverse to the fibre direction and supports fibres during compressive stress. However, the matrix is often the weakest element in the laminate due to its comparatively low strength [13].

Lightweight structures are often built with thin-walled, flat geometries to minimize weight while maintaining structural integrity. As the forces act in different directions, the load-bearing fibres must also be arranged in various directions. To achieve this, plies with different fibre orientations are stacked on top of each other. This layering is typically done using UD plies, where the fibres run parallel to a single direction, resulting in direction-dependent mechanical properties known as anisotropy. By adjusting various parameters, it is possible to create laminates with desired stiffness and strength properties. The following variables can be adjusted

- Number of plies
- Proportions of fibre and matrix within a ply
- Fibre direction of the individual plies
- Thickness of the individual plies
- Sequence of the plies [13]

In the following, assume a real three-dimensional vector space $\mathbb{R}^3$ with an orthonormal basis $\{e_1, e_2, e_3\}$ as in Section 2.2.1 [14]. As previously mentioned, UD plies exhibit different mechanical properties in different directions. In general, nine different stresses act on a material volume element (see Figure 2.5), including the three normal stresses $\sigma_1, \sigma_2, \sigma_3$, and six shear stresses $\tau_{23}, \tau_{32}, \tau_{13}, \tau_{31}, \tau_{12}, \tau_{21}$ [13]. In the given vector space, the form the components of the Cauchy stress tensor $\boldsymbol{\sigma}$ from Theorem 2.9:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 & \tau_{12} & \tau_{13} \\ \tau_{21} & \sigma_2 & \tau_{23} \\ \tau_{31} & \tau_{32} & \sigma_3 \end{bmatrix} \text{ [14].} \tag{2.27}$$

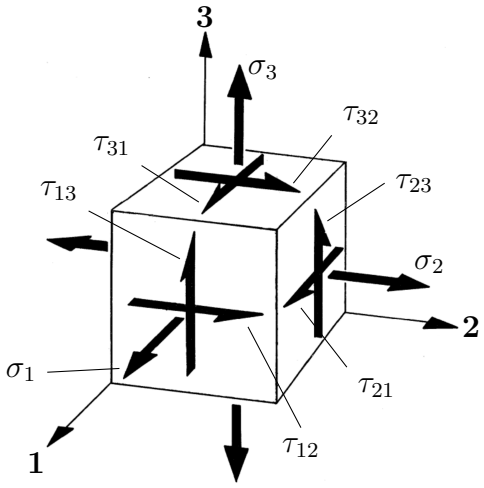


Figure 2.5: Material volume element with stresses (adapted from [13])

Due to anisotropy and its moment equilibrium, the shear stresses are assigned to each other in pairs, reducing the number of six to three, resulting in a total of six stresses. According to the law of linear elasticity, also known as Hooke's law, there is a direct relationship between stress and strain, which can be expressed as

$$\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \gamma_{23} \\ \gamma_{31} \\ \gamma_{12} \end{bmatrix} = \underbrace{\begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} & S_{15} & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & S_{25} & S_{26} \\ S_{31} & S_{32} & S_{33} & S_{34} & S_{35} & S_{36} \\ S_{41} & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ S_{51} & S_{52} & S_{53} & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & S_{63} & S_{64} & S_{65} & S_{66} \end{bmatrix}}_{\text{Compliance matrix } \boldsymbol{S}} \cdot \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_{23} \\ \tau_{31} \\ \tau_{12} \end{bmatrix} \tag{2.28}$$

For compliance matrix $\boldsymbol{S}$ follows $S_{ij} = S_{ji}$, reducing the number of unknowns [13]. With the stresses $\sigma_1, \sigma_2, \sigma_3,\ \tau_{23}, \tau_{31}, \tau_{21}$, it is possible to calculate the vector of internal forces $\boldsymbol{R}$ from equation (2.15) [9]. In the given vector space, the strains $\epsilon_1, \epsilon_2, \epsilon_3$ and $\gamma_{23}, \gamma_{31}, \gamma_{21}$ form the components of the linearized strain tensor $\boldsymbol{\epsilon}$ from equation (2.2):

$$\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 & \frac{1}{2}\gamma_{21} & \frac{1}{2}\gamma_{31} \\ \frac{1}{2}\gamma_{21} & \epsilon_2 & \frac{1}{2}\gamma_{23} \\ \frac{1}{2}\gamma_{31} & \frac{1}{2}\gamma_{23} & \epsilon_3 \end{bmatrix} \text{ [14].} \tag{2.29}$$

A special case of anisotropy is orthotropy, which requires only nine independent constants to formulate the law of elasticity due to the presents of three orthogonal symmetry planes. A UD ply is a transversal isotropic material which is a special case of orthotropy. As shown in Figure 2.6, a UD ply has orthogonal to the fibre direction an infinite amount of symmetry plane that all have the same properties. Hooke's law can now be formulated as

$$\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \gamma_{23} \\ \gamma_{31} \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{12} & 0 & 0 & 0 \\ S_{12} & S_{22} & S_{23} & 0 & 0 & 0 \\ S_{12} & S_{23} & S_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & S_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & S_{66} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_{23} \\ \tau_{31} \\ \tau_{12} \end{bmatrix} \tag{2.30}$$

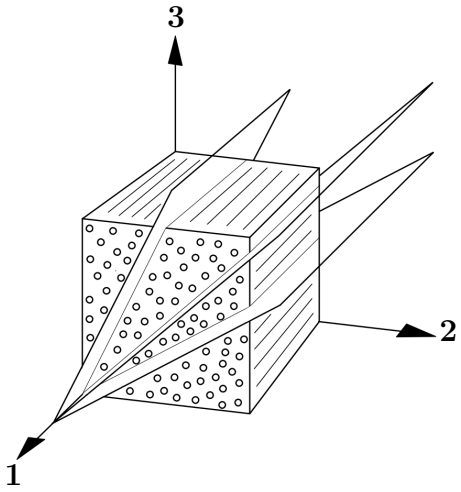


Figure 2.6: Volume element of a UD ply with symmetry planes (adapted from [13])

The law of elasticity is often formulated with the constants Young's modulus E, shear modulus G, and Poisson's ratio ν. The Young's modulus is the ratio of stress and strain in case of ideal, linear elasticity. In case of shear load, it is referred to as shear modulus. The Poisson's ratio is the ratio of transverse and longitudinal expansion. The values in the compliance matrix from equation (2.30) can be written as

$$\begin{aligned} S_{11} &= \tfrac{1}{E_1}, \quad S_{12} = \tfrac{-\nu_{12}}{E_2}, \quad S_{22} = \tfrac{1}{E_2}, \; S_{23} = \tfrac{\nu_{23}}{E_2}, \\ S_{44} &= \tfrac{1}{G_{23}}, \quad S_{55} = \tfrac{1}{G_{12}}, \quad S_{66} = \tfrac{1}{G_{12}}. \end{aligned}$$

The shear modulus G_{23} can be expressed in terms of E_2 and ν_{23}, resulting in five independent parameters to describe the law of elasticity [13].

2.4 LS-DYNA

LS-DYNA is a widely used multi-physics software with its core-competency in non-linear transient dynamic FE analysis. One of its key strengths lies in its use of keyword inputs, which form a flexible and logically organized database that are easy to understand. This enables users to group similar functions under the same keyword. For example, the keyword *ELEMENT includes solid, beam, shell, spring elements, and many others [15]. A FE simulation model typically consists of

- Material model
- Mesh with element formulation
- Contacts between the parts
- Time integration method

In the following sections, a few features for LS-DYNA relevant to this work will be discussed.

2.4.1 Thick Shell Elements

Thick shell elements (*TSHELL) are a type of eight node bending element that lies between a thin shell and solid elements in terms of its complexity. The shape functions N_I used in these elements are identical to those of a solid eight node element

$$N_I(\xi, \eta, \zeta) = \frac{1}{8}(1 + \xi\xi_I)(1 + \eta\eta_I)(1 + \zeta\zeta_I), \tag{2.31}$$

where $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ are the coordinates of the reference element $\hat{K}$ introduced in Section 2.2.3. Depending on the element formulation, thick shell elements can be extruded thin shell elements or layered brick elements. These latter elements can employ fully reduced or selectively reduced integration rules, depending on the desired level of accuracy and computational efficiency. The user can also specify the number of through-thickness integration points. In addition, a composite option allows users to specify an own material, thickness, and material angle for each through-thickness integration point, making it possible to combine multiple plies in one element. In this work, the element formulation 7 is utilized, representing layered solids with 3D stress updates based on an enhanced strain model with 2×2 plane integration [12], [15].

2.4.2 Mortar Contacts

Mortar contacts are designed to work with implicit analysis and are a type of segment-to-segment contact based on a penalty formulation. This means that there are always small penetrations between the parts in contact, which are necessary for transferring the contact force. The contacts are nonsymmetric, with a tracked and a reference surface. Only the nodes of the tracked surface are checked to see if they penetrate the reference surface. The segment-to-segment contact works by considering two overlapping and penetrating segments on each side of the contact interface. A consistent nodal force assembly is performed for these segments, taking into account their individual shape functions. This process allows for the transfer of contact forces between the contacting parts [15], [16]. The Mortar contact can be theoretically treated as a generalized FE. In this context, each element is composed of two contact segments that have their own isoparametric representation, inherited from the underlying FE formulation [12].

2.4.3 MAT_54

The material model *MAT_ENHANCED_COMPOSITE_DAMAGE or MAT_54 [17] is a built-in composite material model in LS-DYNA designed to handle orthotropic materials such as UD composite laminates with distinct properties in the longitudinal and transverse directions (see Section 2.3). This model can be utilized for simulating thin shells, thick shells, and solids element. MAT_54 is one of the most widely used material models for composite simulations due to its simplicity and small number of input parameters. This reduces the computational resources required for a simulation and simplifies material testing to determine the input parameters. However, MAT_54 suffers from oversimplification of the complex physical mechanisms occurring during failure, leading to inaccuracies [18].

MAT_54 is a progressive failure model. Some of its parameters can be obtained through material testing, while others are non-physical and must be determined through trial and error [18]. Table B.1 in Appendix B provides an overview of all input parameters used

in this work. In the elastic region, the strain in each ply can be calculated using the following equations

$$\text{Fibre (longitudinal, 1-direction): } \epsilon_1 = \frac{1}{E_1}(\sigma_1 - \nu_{12}\sigma_2), \tag{2.32}$$

$$\text{Matrix (transverse, 2-direction): } \epsilon_2 = \frac{1}{E_2}(\sigma_2 - \nu_{21}\sigma_1), \tag{2.33}$$

$$\text{Shear (12-direction): } \gamma_{12} = \frac{1}{G_{12}}\tau_{12} + \alpha\tau_{12}^3, \tag{2.34}$$

where $\epsilon_1, \epsilon_2, \gamma_{12}$ are the components of the linearized strain tensor (see equation (2.29)) and $\sigma_1, \sigma_2, \tau_{12}$ are the components of the Cauchy stress tensor (see equation (2.27)). The parameters E_1, E_2, G_{12}, ν_{12} and ν_{21} represent the Young's modulus, shear modulus, and major/minor Poisson's ratios in each direction. The parameter α is a weighting factor for the non-linear shear stress term called $ALPH$ for MAT_54 [18]. The equations (2.32) to (2.34) are a special case of the Hooke's law for transversal orthotropy from equation (2.30) in the 2D-plane with an additional non-linear stress term.

In the plastic region of MAT_54, the model accounts for four different failure modes (e_f, e_s, e_m, e_d) under plane stress conditions. For example, the compressive failure mode e_c for the fibres with $\sigma_1 < 0$ is defined by

$$e_c^2 = \left(\frac{\sigma_1}{XC}\right)^2 - 1 \quad \begin{cases} \geq 0 \text{ failed} \\ < 0 \text{ elastic.} \end{cases} \tag{2.35}$$

Upon reaching failure follows $E_1 = E_2 = \nu_{12} = \nu_{21} = 0$. The failure mode e_d describes the compressive matrix failure mode activating when $\sigma_{22} < 0$ and is defined by

$$e_d^2 = \left(\frac{\sigma_2}{2SC}\right)^2 + \left(\left(\frac{YC}{2SC}\right)^2 - 1\right)\frac{\sigma_2}{YC} + \left(\frac{\tau_{12}}{SC}\right)^2 - 1 \quad \begin{cases} \geq 0 \text{ failed} \\ < 0 \text{ elastic.} \end{cases} \tag{2.36}$$

When failure is reached, it follows $E_2 = \nu_{12} = \nu_{21} = G_{12} = 0$.

Here, XC is the fibre compressive strength, YC the matrix compressive strength and SC the shear strength. When any of the conditions above are exceeded within an element in a ply, the elastic properties for that ply are set to zero [18]. MAT_54 includes non-physical parameters known as $SLIM$ parameters, mapping the behaviour after reaching the maximum stress. When the values for XC, YC or SC are reached within an element, the stress is reduced based on the factors $SLIMC1$, $SLIMC2$, and $SLIMS$ respectively. The stress is then held constant until the value of EFS is reached, at which point the element is deleted [18], [19]. Figure 2.7 shows a typical compressive stress-strain curve for fibres in MAT_54, highlighting the influence of the parameters.

The non-physical parameter EFS is called effective failure strain and is given by

$$EFS = \sqrt{\frac{4}{3}(\epsilon_{11}^2 + \epsilon_{11}\epsilon_{22} + \epsilon_{22}^2 + \epsilon_{12}^2)}. \tag{2.37}$$

The critical EFS value can either be calculated by determining the different strains at failure or obtained through trial and error. If the strains within an element exceed the EFS value, it will be deleted. MAT_54 also accounts for the decrease in longitudinal

compressive strength (XC) of a ply when transverse matrix failures occur. This phenomenon is caused by a reduction in the efficiency of the matrix in supporting the fibres against micro-buckling. The reduction factor $YCFAC$ is used to represent this effect, with

$$XC = YCFAC \cdot YC. \tag{2.38}$$

The value of $YCFAC$ can only be determined by trial and error [18], [19].

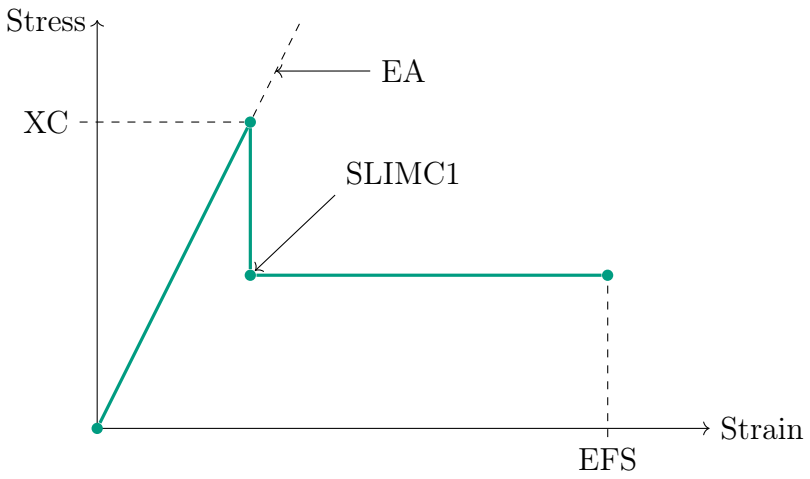


Figure 2.7: Typical stress-strain-curve for MAT_54 (adapted from [19])

2.5 Regression in Machine Learning

ML can be viewed as a subfield of Artificial Intelligence. ML allows a system to learn on their own, rather than being explicitly programmed for an explicit task. It develops models that can identify patterns in historical data and use these patterns to make predictions. There are three major categories in ML: supervised, unsupervised and reinforcement learning. Unsupervised learning involves training a model on unlabeled data and learns to extract patterns and features from it without any prior knowledge. In reinforcement learning, the model is trained on the environment in which it operates by maximizing a reward signal. In this work, we will focus on supervised learning, which involves training the model on labeled data. and then using it to make predictions for unlabeled data [20]. Given a dataset $\mathcal{D}$, which is a collection of information about N cases, the dataset contains a set of p features $\mathbf{x} = (\mathrm{x}_1, \ldots, \mathrm{x}_p) \in \mathbb{R}^p$ and a target variable $\mathrm{y} \in \mathbb{R}$

$$\mathcal{D} := \{(\mathbf{x}_i, \mathrm{y}_i),\ i = 1, \ldots, N\}. \tag{2.39}$$

The main task in supervised learning is to learn from $\mathcal{D}$ a mapping H of the features $\mathbf{x}$ onto the target values y

$$(\mathrm{x}_1, \ldots, \mathrm{x}_p) \xrightarrow{H} \mathrm{y} \qquad [21]. \tag{2.40}$$

In classification problems, the labels are categorical and represent different classes or categories. In regression problems, the labels are continuous values that represent the actual value [20]. In regression, the parameters of the mapping $H(\mathbf{x})$ are adjusted in each iteration to minimize the total error. To compute the total error, a loss function is used,

which measures the difference between the actual and predicted values. One of the most common loss functions is the squared error, defined as

$$\sum_{i=1}^{N}(\mathrm{y}_i - H(\Theta, \mathbf{x}_i))^2. \tag{2.41}$$

To identify the best parameters Θ of function H, we start with initial values of Θ and then move in the negative of the gradient descent direction [22].

Before training a ML model, it is essential to split the dataset into training and testing data. The training data is used to train the ML model, allowing it to learn patterns from the features. Once the model is trained, it can make predictions on the test data. Since the labels of the test data are known, it is now possible to assess the quality of the prediction [20]. Figure 2.8 illustrates the basic architecture of supervised learning process.

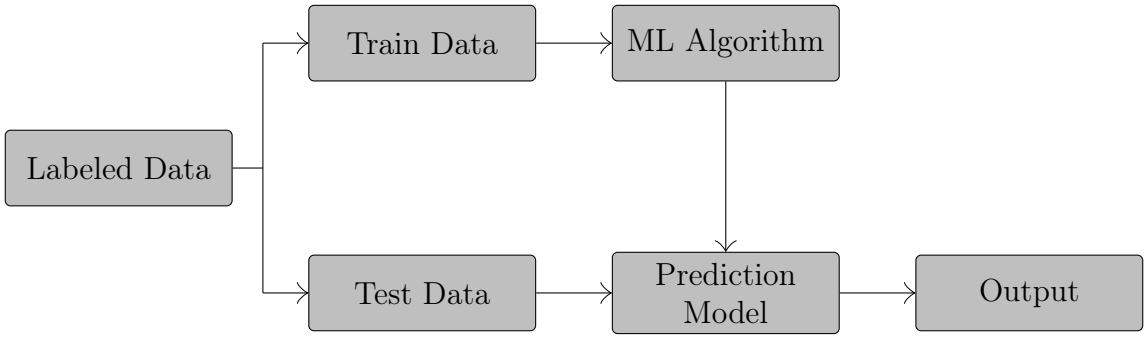


Figure 2.8: Basic architecture of a supervised learning process (adapted from [20])

To evaluate the performance of a regression ML process, several different scoring functions and evaluation metrics can be used. One popular metric is the Mean Absolute Error (MAE), which corresponds to the expected value of the squared error or loss. For a predicted value $\hat{\mathrm{y}}_i$ of the i-th observation with y_i being the corresponding true value, the MAE is defined as

$$\mathrm{MAE}(\mathrm{y}, \hat{\mathrm{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}|\mathrm{y}_i - \hat{\mathrm{y}}_i| \tag{2.42}$$

with N being the number of samples. Another widely used metric is the Mean Absolute Percentage Error (MAPE). The idea behind this metric is to be sensitive to relative errors and not affected by the scaling of the target variable. For a predicted value $\hat{\mathrm{y}}_i$ of the i-th observation and the corresponding true value y_i, the MAPE can be calculated with

$$\mathrm{MAPE}(\mathrm{y}, \hat{\mathrm{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}\frac{|\mathrm{y}_i - \hat{\mathrm{y}}_i|}{\max(\epsilon, |\mathrm{y}_i|)} \tag{2.43}$$

where N is the number of samples and ϵ is an arbitrary small strictly positive number to avoid undefined results when y is zero [23].

In the following sections, the steps from preprocessing the data to ML, including the algorithms RF and SVR for regression problems, are presented.

2.5.1 Feature Engineering

When training a ML model, it is common that the data used for training is in a raw form that's not suitable for training the model. Therefore, it's essential to clean, organize, and possibly alter the data. This process is called feature engineering. According to Ozdemir and Susarla [24], feature engineering "is the process of transforming data into features that better represent the underlying problem, resulting in improved ML performance". Data is typically organized in a tabular format with rows (observations) and columns (attributes). A feature is an attribute of a dataset that is important for the ML process. Feature engineering can be applied to data at any stage of the development process, making it an iterative process. To evaluate the effectiveness of a feature engineering procedure, the following steps are deployed:

1. Obtain a baseline performance of the ML model
2. Apply feature engineering techniques
3. Measure the performance of the ML model and compare it to the baseline performance
4. If the performance of the model improves, the procedure can be applied to the ML pipeline [24]

Feature Improvement

One aspect of feature engineering is feature improvement, which involves processes such as structuring unstructured data, inserting missing values, and modifying existing columns and rows. Missing values are a common occurrence in datasets with different ways to handle them. For example, the rows with missing values can be deleted or filled in with an average value or a specific indicator that signals the presence of a missing value.

Another part of feature improvement is standardizing and normalizing the data. In many cases, the columns of a dataset have vastly different means, minima, maxima and standard deviations. This can be problematic for some ML models that are affected greatly by the scale of the data. These models are often distance-based models such as SVR, where a feature with high values dominate the model. A popular method for data normalization is the z-score standardization. The values in a feature are re-scaled to have a mean of zero and a standard deviation of one. For a value x of a cell, the z-score or new value z is calculated as

$$z = \frac{x - \mu}{\sigma}, \tag{2.44}$$

where μ is the mean of the column and σ is the standard deviation of the column [24].

Feature Construction

Feature construction is the part of feature engineering that involves creating new features, which hold new information and help generate new patterns for ML processes. Often features are created from existing features. A constructed feature can be added to the original dataset or replace some existing features, leading to a dimensionality reduction. One method for feature construction is to use mathematical operators or a polynomial. Other approaches focus on transforming features into different data types, such as transforming a continuos feature into categorical data [24].

Feature Selection

Feature selection is the process that involves determining which attributes are not useful for the ML process and should therefore be removed. If given p features, the goal is to find a subset of k features, where $k < p$, that improve the performance of the ML model. Less features in a dataset lead to dimensionality reduction, resulting in reduced training time for the model. There are various statistical-based methods and model-based feature selection techniques available. Statistical-based methods include Pearson correlations, which measure the linear relationship between two features. Model-based methods, such as RF, use a metric that determines feature importance to choose its split [24].

One popular technique for evaluating feature performance is permutation feature importance. This method measures the contribution of each feature after the model has been fitted by randomly shuffling the values of a single feature and observing the resulting decline model performance. A key advantage of this method is its ability to be applied to any fitted estimator. The outline for computing the permutation feature importance is shown in Algorithm 1 [25].

Algorithm 1: Permutation Feature Importance [25]

Input: Fitted model M, dataset $\mathcal{D}$, number of repetitions K
Output: Importance scores i_j for each feature x_j

Compute reference score $s \leftarrow \text{score}(M, D)$;
foreach x_j *in* $\mathcal{D}$ **do**
  for $k \leftarrow 1$ **to** K **do**
    Generate $\tilde{\mathcal{D}}_{k,j}$ by shuffling column j of $\mathcal{D}$ randomly ;
    Compute $s_{k,j} \leftarrow \text{score}(M, \tilde{\mathcal{D}}_{k,j})$;
  Compute $i_j \leftarrow s - \frac{1}{K} \sum_{k=1}^{K} s_{k,j}$;

Outlier Detection

Outlier detection is a method in feature engineering that identifies abnormal or unusual observations in a dataset, which deviate from the rest. Outlier detection is also known as unsupervised anomaly detection. The anomalies can have a negative impact on the performance of ML models and are often removed from the dataset.

One approach for performing outlier detection is using Isolation Forest (IF), which is based on a RF. The workings of the RF will be discussed in Section 2.5.2. The IF algorithm isolates a sample by randomly selecting a feature and then randomly choosing a split value that must fall between the maximum and minimum values of the selected feature. The number of splittings required to isolate an observation is equivalent to the path length from the root node to the terminating node in the forest. This means that observations with noticeably shorter paths than the average path length of the forest are considered anomalies [26].

Another method to outlier detection is the Local Outlier Factor (LOF) algorithm, which computes a score that measure the local density deviation of an observation with respect to its neighbours. If a sample has a substantially lower density than its neighbours, which means it is isolated, then it is considered an outlier. The LOF algorithm works by obtaining the locality from k-nearest neighbours and using their to estimate the local

density. The LOF score of an observation is the ratio of the average local density of its k-nearest neighbours and its own local density. A normal sample has a local density similar to those of its neighbours, while an outlier has a much smaller local density. The strength of the LOF algorithm lies in its ability to consider both local and global properties of datasets. This means that it can still perform well on datasets where outliers have different underlying densities [26], [27].

2.5.2 Random Forest

RF is an ensemble ML method combining the prediction from multiple decision trees into a single result. An illustration of this process is shown in Figure 2.9. RF was first introduced by Breimann [28] in 2001 and is a supervised learning method that can be used for both classification and regression tasks. Individual decision trees tend to overfit and exhibit high variance, but by combining multiple diverse trees, each built from a different sample, some of these errors are cancelled out when taking an average of the predicitons. This leads to a reduced variance, but sometimes this method can slightly increase bias. However, the variance reduction is often significant, resulting in a better model [29]. Breimann [28] has shown that the generalization error converges if the number of trees goes to infinity. The advantages of a RF are include its ability to handle high-dimensional data and large-scale datasets. Additionally, RF has good generalization performance and reduces the risk of overfitting. Furthermore, it can handle missing values, outliers and even complex data with non-linear relationships [30].

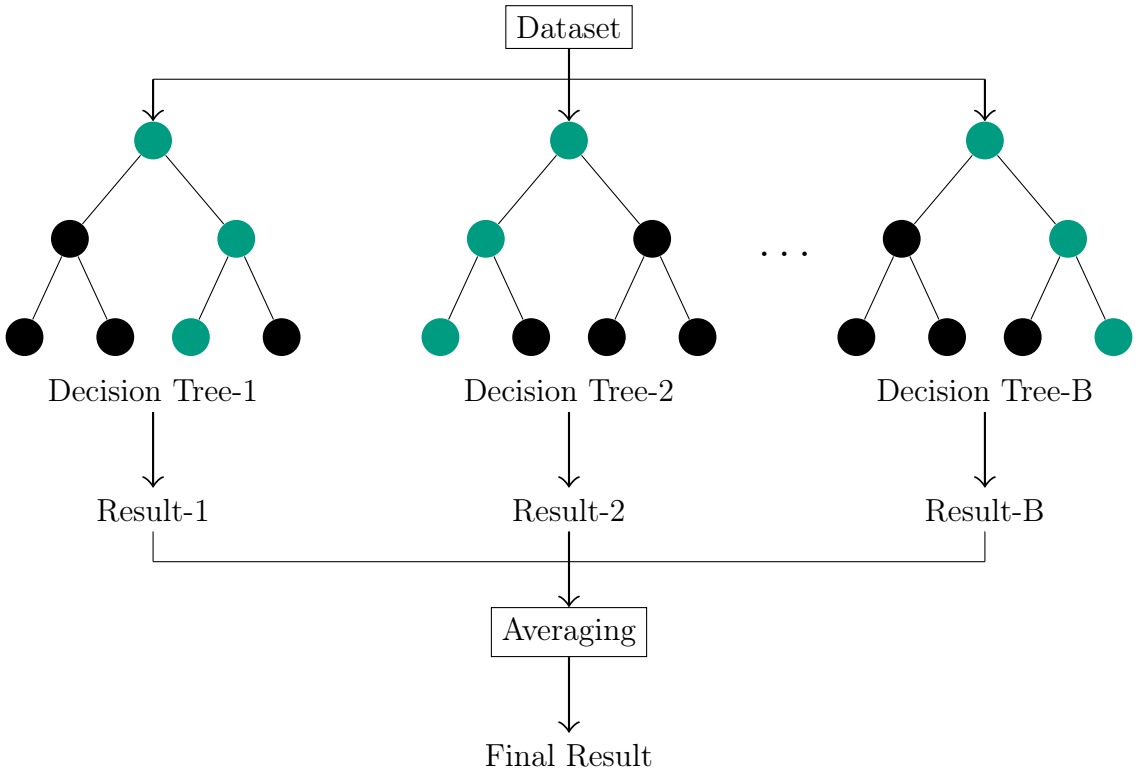


Figure 2.9: Illustration of a RF (adapted from [31])

As previously mentioned, decision trees are the basis of the RF method. Given a set of training vectors $\mathbf{x_i} \in \mathbb{R}^p$ and their associated labels $y_i \in \mathbb{R}$, a decision tree recursively partitions the feature space to group samples with similar values together. The data at

node m is represented by Q_m with n_m samples. The data is partitioned into the two subsets $Q_m^{left}(\theta)$ and $Q_m^{right}(\theta)$, for each candidate split $\theta = (j, t_m)$ consisting of a feature j and a threshold t_m

$$Q_m^{left}(\theta) = \{(\mathbf{x}, \mathrm{y}) | \mathrm{x}_j \leq t_m\}, \tag{2.45}$$

$$Q_m^{right}(\theta) = Q_m \setminus Q_m^{left}(\theta). \tag{2.46}$$

The quality of a candidate split of node m is assessed using a loss function L

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} L(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} L(Q_m^{right}(\theta)). \tag{2.47}$$

The goal is to select the parameters θ^* that minimize the impurity

$$\theta^* = \mathrm{argmin}_\theta \; G(Q_m, \theta). \tag{2.48}$$

This step is then recursively applied until the maximum allowable depth is reached. For regression problems, a common loss function is the means squared error

$$L(Q_m) = \frac{1}{n_m} \sum_{\mathrm{y} \in Q_m} (\mathrm{y} - \overline{\mathrm{y}}_m)^2 \tag{2.49}$$

where $\overline{\mathrm{y}}_m$ is the learned mean value [32].

RF employs a technique called bagging or bootstrap aggregation to build and evaluate individual trees. Bootstraping is a resampling method that randomly resamples a dataset with replacement. The idea behind bagging is to generate B new training sets from the initial training set through boosting. Once these distinct training sets are obtained, B separate decision trees are fitted, each estimating its own prediction $H_1(\mathbf{x}), H_2(\mathbf{x}), \ldots, H_B(\mathbf{x})$ at each point $\mathbf{x}$. These predictions are then summed up to obtain the final prediction

$$H_{bag}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} H_b(\mathbf{x}). \tag{2.50}$$

The RF using bagging is outlined in Algorithm 2. The value for the number of randomly selected features m is central to the RF. When m is small, the degree of decorrelation increases, leading to a higher chance of generating trees that are very different from each other. However, this results in rougher representations of the tree and a significant reduction in variance but at the expense of a larger bias. On the other hand, when the value for m is large, the trees provide a better representation, but the benefit of aggregating shrinks significantly, resulting in smaller bias and higher variance [21]. Nevertheless, considering all features is a good default value to start with. Another important parameter is the number of trees B. Generally, the more trees a model has, the better the results will be. However, with more trees, the computational time increases, and the results will stop getting significantly better beyond a certain number of trees [29].

Algorithm 2: Random Forest Regression [21]

Input: Dataset $\mathcal{D}$, number of trees B, number of features p, number of randomly selected features m, minimum node size

Output: Predicted value $\hat{y}$

for $b \leftarrow 1$ **to** B **do**
  Draw a bootstrap sample $\mathcal{D}_b$ from $\mathcal{D}$;
  while *units in node* $<$ *minimum number of units* **do**
    Draw at random m out of p features;
    Use these m features to grow a tree by binary splitting;
  Compute prediction $H_b(\mathbf{x})$;
Compute $\hat{y} \leftarrow \frac{1}{B}\sum_{b=1}^{B} H_b(\mathbf{x})$

2.5.3 Support Vector Regression

Support Vector Machines, first introduced by Cortes and Vapnik [33] in 1995, are a collection of supervised ML methods that can be used for classification, regression and outlier detection. When applied to a regression problem, Support Vector Machines are referred to as SVR. The idea behind SVR is to find a hyperplane that captures the relationship between the input and target variables, minimizing the error between predicted and actual values. To achieve this, SVR uses a kernel that maps the input features into a higher-dimensional space, which corresponds to a non-linear decision boundary in the original feature space [31]. One of the main advantages of SVR is that the computational complexity is independent from the dimensionality of the input space. Additionally, SVR is robust to outliers and has low computational cost. It performs well on non-linear problems as it exhibits great generalization ability with high prediction accuracy [30].

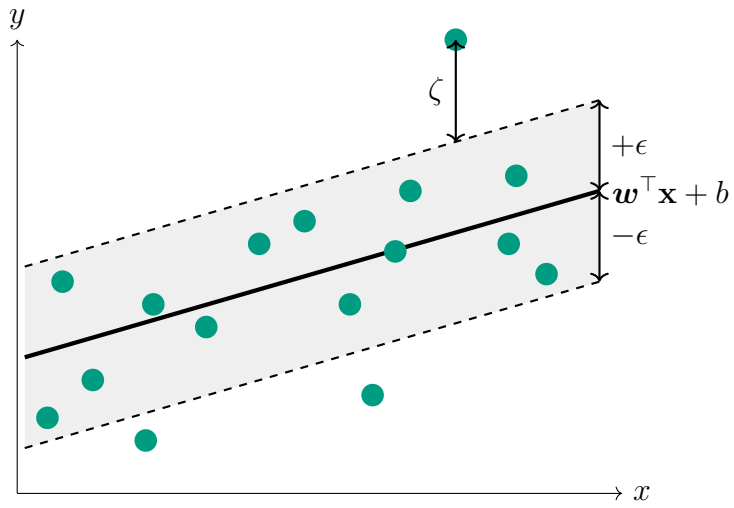


Figure 2.10: Illustration of a linearSVR (adapted from [34])

In ϵ-SVR, the goal is to find a function $H(\mathbf{x})$ that has at most ϵ deviation from the actual targets y_i for all data points. Additionally, the function should be as flat as possible, meaning that errors less than ϵ are neglected, but errors larger than ϵ are not accepted [34]. Given N training vectors $\mathbf{x}_i \in \mathbb{R}^p$ and the associated labels $y_i \in \mathbb{R}$ grouped in $\mathbf{y} \in \mathbb{R}^N$, we aim to determine $\boldsymbol{w} \in \mathbb{R}^p$ and $b \in \mathbb{R}$ for the function $H(\mathbf{x})$ of the form

$$H(\mathbf{x}) = \boldsymbol{w}^\top \phi(\mathbf{x}) + b. \tag{2.51}$$

where ϕ implicitly maps the training vectors $\mathbf{x}$ into a higher-dimensional space making SVR non-linear. The concept of flatness in this context means finding a small $\boldsymbol{w}$ that minimizes its norm. This leads to the convex optimization problem

$$\begin{aligned} \min \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to} \quad & \mathrm{y}_i - \boldsymbol{w}^\top \phi(\mathbf{x}_i) - b \leq \epsilon, \\ & \boldsymbol{w}^\top \phi(\mathbf{x}_i) + b - \mathrm{y}_i \leq \epsilon. \end{aligned} \tag{2.52}$$

However, this optimization problem is not always feasible, which means there exists no H that approximates all pairs $(\mathbf{x}_i, \mathrm{y}_i)$ with ϵ precision. To address this issue, slack variables $\boldsymbol{\zeta}, \boldsymbol{\zeta}^*$ are introduced to allow some margin. The optimization problem can now be formulated as

$$\begin{aligned} \min_{\boldsymbol{w},b,\zeta,\zeta^*} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}(\zeta_i + \zeta_i^*) \\ \text{subject to} \quad & \mathrm{y}_i - \boldsymbol{w}^\top \phi(\mathbf{x}_i) - b \leq \epsilon + \zeta, \\ & \boldsymbol{w}^\top \phi(\mathbf{x}_i) + b - \mathrm{y}_i \leq \epsilon + \zeta^*, \\ & \zeta_i, \zeta_i^* \geq 0 \end{aligned} \tag{2.53}$$

with a constant $C > 0$ determining the trade-off between the flatness of H and the amount up to which errors larger than ϵ are tolerated. Therefore, it acts as an inverse regularization parameter. The model is usually very sensitive towards C. Figure 2.10 illustrates the approach of the SVR. In most cases, the optimization problem (2.53) can be solved more easily in the dual formulation

$$\begin{aligned} \min \quad & \frac{1}{2}(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*)^\top \boldsymbol{Q}(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + \epsilon \boldsymbol{e}^\top(\boldsymbol{\alpha} + \boldsymbol{\alpha}^*) - \mathbf{y}^\top(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) \\ \text{subject to} \quad & \boldsymbol{e}^\top(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*), \\ & 0 \leq \alpha_i, \alpha_i^* \leq C. \end{aligned} \tag{2.54}$$

The parameters α_i and α_i^* are Lagrange multipliers, $\boldsymbol{e}$ is the vector of all ones and $\boldsymbol{Q}$ is a $N \times N$ positive semidefinite matrix with $Q_{ij} \equiv K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$ being the kernel. After solving the optimization problem (2.54), we receive α_i and α_i^*, which are used to formulate the function H as

$$H(\mathbf{x}) = \sum_{N}^{i=1}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + b \tag{2.55}$$

For all samples $\mathbf{x}_i$ inside the ϵ-tube (shaded region in Figure 2.10), the coefficients α_i and α_i^* vanish, meaning that these points do not contribute to the prediction. Only the samples with non-vanishing coefficients form the function H and are called support vectors [34], [35].

The kernel function can take many different forms. One popular choice is the Radial Basis Function (RBF) function with

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2}, \tag{2.56}$$

where $\gamma > 0$ is one of the important parameters that heavily influence the performance of a SVR with a RBF kernel. The parameter γ determines how far the influence of a single training example reaches. A low value means 'far', while a high value means 'close'. It can be seen as the inverse of the radius of influence of samples chosen by the model as support vectors. If γ is too small, the model becomes too constrained and cannot represent the complexity of the data. On the other hand, if γ is too large, the model is prone to overfitting, as the radius of the area of influence of the support vectors only includes the support vector itself [36].

2.5.4 Hyperparameter Optimization

Hyperparameters are the parameters of a model that cannot be learned within the model but must be set before training. Examples of hyperparameters for RF include the number of trees, while for SVR, examples include the constant C or γ for a RBF kernel. The hyperparameters of a model can be optimized resulting in better performance of a model. The process of hyperparameter optimization involves

- An estimator such as RF or SVR
- A parameter space
- A method for searching
- A cross-validation scheme
- A score function such as MAE [37]

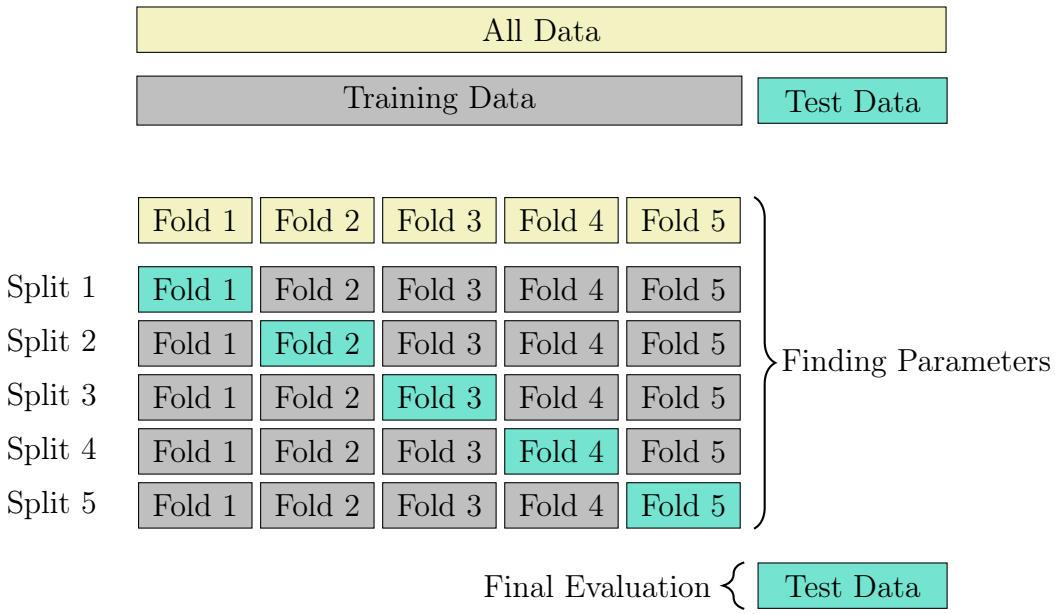


Figure 2.11: Visualization of the k-fold Cross-validation (CV) (adapted from [38])

When performing a hyperparameter optimization, there is a risk of overfitting to the test set because the parameters can be altered to optimize performance on the test set, potentially leaking knowledge about the test set into the model. To prevent this, a CV scheme is necessary for hyperparameter optimization. One approach is k-fold CV, where the training set is randomly split into k smaller sets called folds. Then for each of the k

folds, the model is trained using $k-1$ folds as training data. Subsequently, the model's performance is assessed on the remaining part k of the data serving as a test set [21], [38]. Figure 2.11 illustrates k-fold CV process with 5 folds.

A popular method for hyperparameter optimization is the grid search. In the first step, a grid of parameters is generated from the values specified in the parameter space. This grid is then exhaustively searched to find the best combination, meaning all possible combinations of hyperparameters are tested [37].

Another approach for finding optimal hyperparameter is Bayesian Optimization, which is method for optimizing functions that do not presume any functional forms. In contrast to the grid search, the results of past evaluations are included. They are used for building a probabilistic model mapping the hyperparameters to the probability of a score in the objective function. This results in a more efficient optimization process as the selection of the next hyperparameters considers promising hyperparameters in past results. The goal of Bayes search is to find the maximum value of an unknown objective function f:

$$x^* = \arg\max_{x \in \mathcal{X}} \; f(x), \tag{2.57}$$

where $\mathcal{X}$ is the search space of the hyperparameters x. In Bayesian optimization, the function f is treated as a random function, and a prior is placed over it to capture its behaviour. After collecting function evaluations, the prior is updated to construct the posterior distribution over the objective function f. The posterior distribution is then used to build an acquisition function to determine the next values for the hyperparameters [39].

2.6 State of the Art for Predicting Mechanical Properties with Machine Learning

ML has made significant progress in recent years, leading to widespread applications in predicting the properties of composite components based on experimental and simulation results. Researchers have successfully applied ML techniques to predict various properties of composite materials. For example, Wang et al. [40] predicted the mechanical properties of composite tubes using experimental data from compression tests. Additionally, Alrsai et al. [31] employed five different ML models to predict the burst pressure of hybrid steel CFRP pipes based on simulation data. The five ML models included a RF and a SVR model, with both achieving the best results out of the five models. Similarly, Zhang et al. [41] used an artificial neural network and a RF to predict the mechanical properties of composite laminates, including the failure factor of puck. Both models were trained on simulation data. While the neural network performed slightly better, the RF was faster to train.

3 Model Design

This section focuses on setting up the simulation model in LS-DYNA, which will be used to simulate various samples with different defects. The objective is a robust and reliable model producing high-quality results with minimal noise. Another consideration is the computational time of the simulation, as it should be kept at a reasonable level without compromising accuracy. This is crucial, as the goal is to generate a large amount of data for the RF and SVR models within the limited time frame of this thesis.

3.1 Hardware Specifications

For this thesis, simulations are executed with LS-DYNA on either the local computer or a workstation. The specifications of each system are shown in Tables 3.1 and 3.2. The workstation specifications for CPU, RAM and storage are listed for one node.

Table 3.1: Hardware specifications for the local computer

Hardware	Specification
Processor (CPU)	Intel Core i7-6700, 4 Cores / 8 Threads, 3.40 GHz
Memory (RAM)	8 GB, 2133 MHz
Main storage (SSD)	237 GB
Data storage (HDD)	931 GB

Table 3.2: Hardware specifications for the workstation

Hardware	Specification
Total number of nodes	20
Cores per node	32
Processor (CPU)	2 × 16-Core AMD EPYC “Milan” 7313
Memory (RAM)	16 × 64 GB DDR4, 3200 MHz
Storage	500 GB Crucial P5 Plus

3.2 Existing Model

A simulation model existed from a previous project, where the objective was to investigate the effect of defects on the mechanical behaviour of the specimen [6]. Four different specimen configurations with a size of 22.0 mm × 172.0 mm × 2.0 mm, were examined, corresponding to $\Omega = [0, 22] \times [0, 172] \times [0, 2]$. The samples were tested under compression, with varying element sizes and triangular gap defects at different positions. Each specimen had a total of four defects placed at different plies of the sample. Additionally, a fifth reference sample was used for comparison, which lacked defects and featured an element size of 1.0 mm. All five samples are depicted in Figure 3.1. The reference sample had a stacking sequence of $[0°/-45°/90°/+45°]_{2s}$. The $2s$ indicates that the stacking is repeated twice and then mirrored. In contrast, the specimens with defective plies exhibited a fibre angle deviation of 12°, resulting in a modified stacking sequence $[0°/-45°/90°/(51°|39°)]_{2s}$.

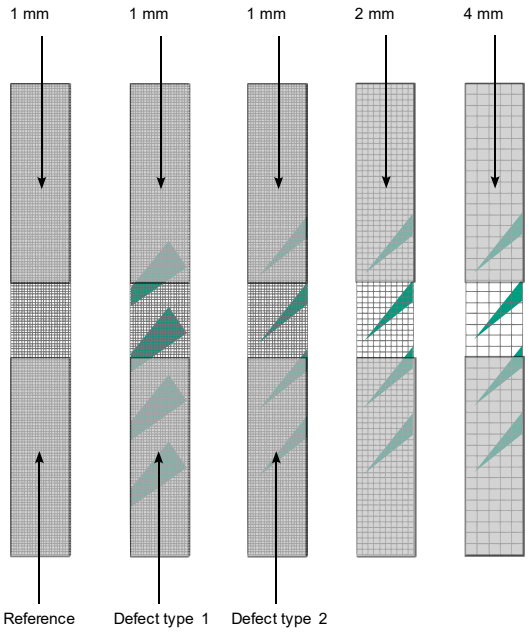


Figure 3.1: Existing models [6]

The mesh for the samples consisted of three-dimensional thick shell (TSHELL) elements, with element formulation 7 and three integration points per CFRP ply. Between the CFRP plies were cohesive layers modelled using three-dimensional eight-node solid elements, which simulated delamination. The cohesive layers use the same nodes as the adjacent TSHELL elements of the CFRP plies. Figure 3.2 shows a close-up of the side view of the mesh, where the plies are displayed in green and the cohesive elements in yellow. The mesh was generated directly from a Python script, utilizing data from the path travelled by the robot during the tape laying process. The advantage of this approach lies in its ability to work with real-world data, but it is also limited by a long execution time, and can only map defects that are present in real samples.

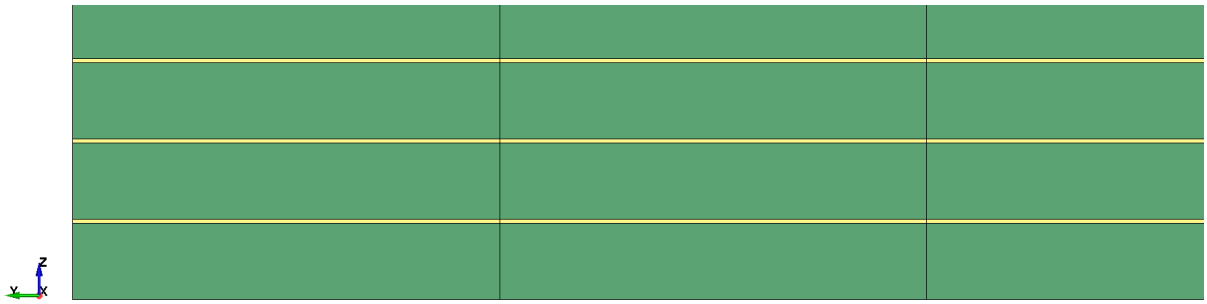


Figure 3.2: Close up of the side view of sample's mesh

The CFRP material is modelled using the physically-based MAT_262 material model, which is designed to identify the parameters from experimental results [17]. The cohesive layer material is represented by the MAT_240 material model, which is a tri-linear elastic-ideally plastic model that considers effects of plasticity and rate-dependency. However, it does not account for brittle fracture behaviour, as the entire separation at failure is considered plastic [17]. In the event of cohesive element failure, the simulation model utilized one global contact *CONTACT_ERODING_SINGLE_SURFACE between the CFRP layers. This contact type is suitable for applications where elements in the contact

interface fail and are deleted [15]. The plies have a thickness of 0.178 mm, while the cohesive layers have a thickness of 0.01 mm. When a gap is present between the tapes, the thickness of the affected ply is reduced to 0.01 mm, rather than 0, due to the fibres moving into the gap and to avoid mesh discontinuity. Figure 3.3 shows a close-up of the mesh with a defect.

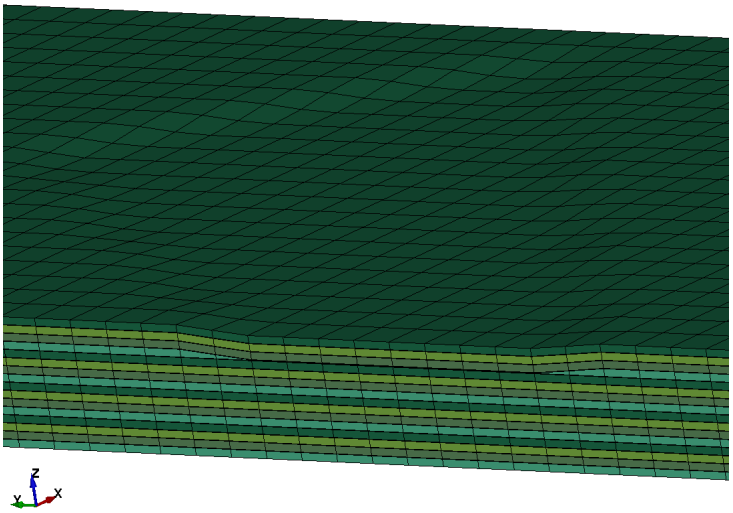


Figure 3.3: Close up of the mesh for a sample with 1 mm element size

The model was set up in the unit system [mm, kg, ms, kN, GPa] and designed to run using explicit time integration with a fixed time step of $\Delta t = 5.6 \cdot 10^{-5}$ ms, resulting to 44643 time steps for a simulation time of $T = 2.5$ ms. Figure 3.4 shows how the boundary conditions were applied. The deformation is applied to the upper part of the sample (green), where nodes undergo a motion in the y-direction that gradually builds up in velocity until reaching a maximum of 0.2 at time $t = 0.5$ ms. At the lower part of the samples (yellow), nodes are fixed with no degrees of freedom. Consequently, Γ_D from equation (2.5) covers $\Gamma_D = [0, 22] \times [0, 75] \times [0, 2] \cup [0, 22] \times [97, 172] \times [0, 2]$, because the fixation and the motion are both Dirichlet boundary conditions. The area in the middle of a sample with a size of 22 mm × 22 mm absorbs all the forces. This is equivalent to the Neumann boundary condition (equation (2.6)) on $\Gamma_N = [0, 22] \times [75, 97] \times [0, 2]$ with $\boldsymbol{\sigma} \cdot \boldsymbol{n} = 0$. The Interface Γ_I (equation (2.7)) covers all the faces between the CFRP TSHELL and the cohesive solid elements.

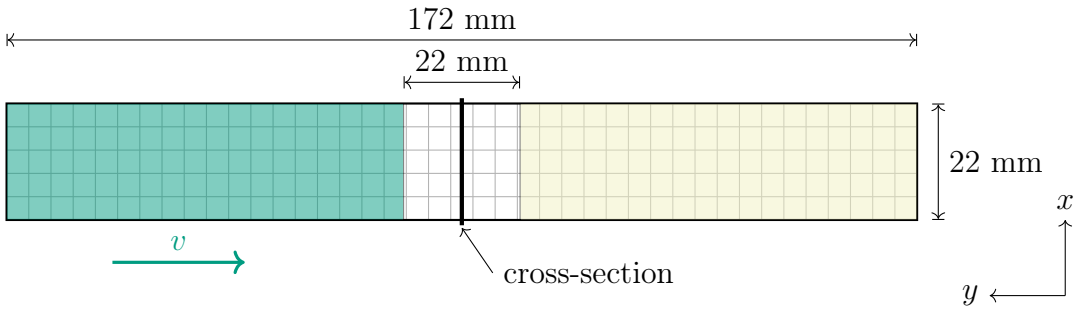


Figure 3.4: Definition of boundary conditions

In the middle of the sample, parallel to its width, a cross-section plane defined by *DATABASE_CROSS_SECTION_PLANE was placed. This plane was used to calculate the resultant forces at this location for each time step t_n and write them in the output files [15]. It is also shown in Figure 3.4. By extracting the force in the y-direction from this plane, the stress for this cross-section in y-direction at each time step t_n can be calculated using

$$\sigma_n = \frac{F_n}{A_n}, \tag{3.1}$$

where F_n is the force and A_n is the area of the cross-section plane at t_n. The strain ϵ_n for this sample at each time step t_n was defined through the displacement u_n in the y-direction for one of the nodes on the clamping surface. This leads to the strain in y-direction

$$\epsilon_n = \frac{u_n}{22.0}, \tag{3.2}$$

where 22.0 is the free length of the sample. With stress and strain in direction of loading defined, a stress-strain curve can now be calculated with equations (3.1) and (3.2) for each time step t_n. Figure 3.5 displays the resulting stress-strain curve for the reference sample without any defects and an element size of 1 mm.

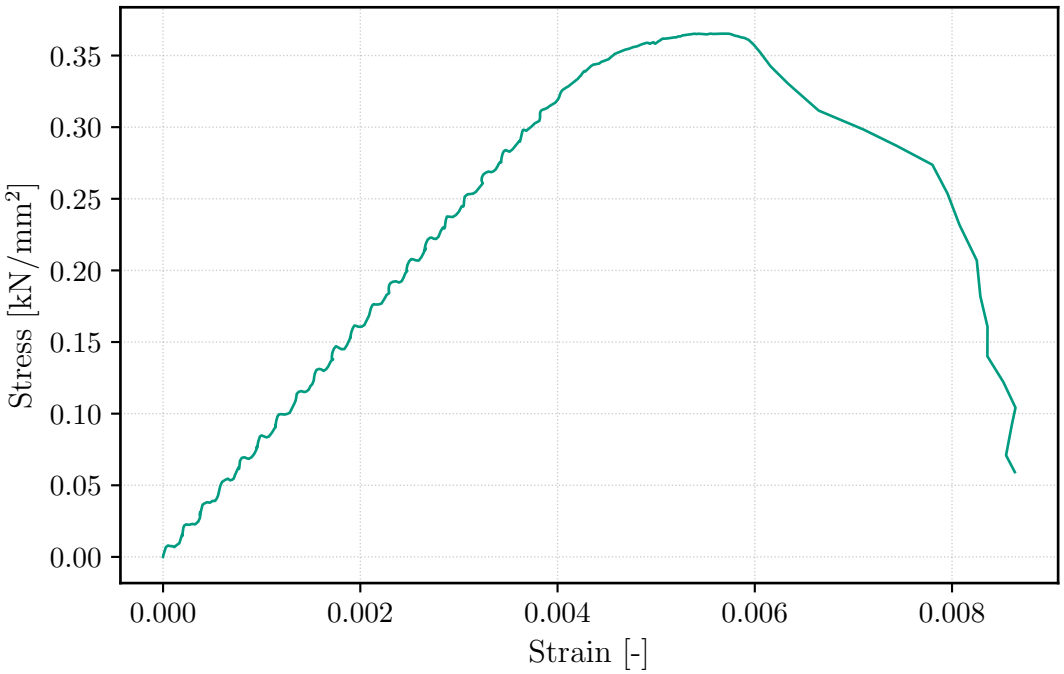


Figure 3.5: Stress-strain curve for existing model

The stress-strain curve of this model exhibits oscillating behaviour before failure due to the explicit time integration method. This indicates that this model is not suitable for generating data for ML algorithms. The noisy data can distort the target variables, such as *Young's modulus* or the *area under curve after failure*. However, this model has a good computational time of 7 minutes and 5 seconds, which is executed on the workstation with one node. Therefore, in the following sections, the model will be modified to address the issues of oscillating behaviour to produce more accurate results and optimize the computational time.

3.3 Implicit Simulation

To mitigate oscillations in the simulation results, the time integration method is switched from explicit to implicit. However, a few adjustments are necessary to the model to accommodate implicit analysis.

The material model must be changed to MAT_54, as MAT_262 cannot be used with implicit simulations. This switch benefits from MAT_54 being a simpler model with fewer input parameters than MAT_262, which should reduce computational time. Some input parameters for MAT_54 can be inherited from MAT_262. For the other parameters, the values from [18] are used for the time being and all the $SLIM$ parameters are set to 0.5. The optimization of the material parameters for MAT_54 is discussed in Section 3.4. Additionally, the contact model needs to be modified from the eroding contact, which is not suitable for implicit simulations, to the Mortar contact *CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR_ID. This Mortar contact is used between the TSHELL elements of the CFRP plies to avoid penetrations between them should the cohesive solid elements fail. This contact model was designed to handle implicit problems and its parameters are mostly set to default values [15] or adopted from [42]. In Appendix A, the complete contact card is listed.

A non-linear dynamic analysis is carried out because the sample behaves non-linearly after failure, and the dynamic setting gives the system stability. A separate control card was added for implicit analysis, with settings mostly based on the default settings provided by LS-DYNA [15], except for some parameters that are adopted from [42].

3.3.1 Time Step Study

First, the influence of the time step on the simulation results is investigated. LS-DYNA automatically adjusts the time step if the number of iterations is not within a predetermined range. This can be controlled through the parameters $ITEOPT$ and $ITEWIN$ [15]. Figure 3.6 illustrates how the automatic time step control range is calculated. If an implicit time step is successful, the required number of iterations is compared to the threshold specified in $ITEOPT$. If it exceeds the range of $ITEWIN$, the next time step is either increased or decreased. For failed implicit steps, the step size is reduced and then repeated with the new size [15].

Sometimes, it is advisable to set an upper limit for the time step, so that no critical points like failure are missed. In the following, several implicit simulations with different maximum allowable time step sizes $\Delta t_{\max}$ will be compared. The samples used for this investigation have no defects and the same dimensions (22.0 mm $\times$ 172.0 mm $\times$ 2.0 mm), boundary conditions Γ_D, Γ_N, Γ_I and stacking sequence $[0°/-45°/90°/+45°]_{2s}$ as existing models from Section 3.2. The element size is set to 2 mm $\times$ 2 mm with a thickness of 0.115 mm for CFRP TSHELL elements and 0.01 mm for cohesive solid elements, resulting in a total of 33408 nodes with 14190 elements for the cohesive plies and 15136 elements for the CFRP plies. From this only 4228 nodes, 1980 cohesive elements, and 2288 CFRP elements experience no boundary conditions and absorb all the forces. The simulation time is $T = 1.5$ ms. The stress and the strain is calculated in the same way as in equation (3.1) and (3.2), respectively, with a cross-section in the middle of the sample as in the existing model from Section 3.2. Figure 3.7 shows the resulting stress-strain curves from the simulations with different $\Delta t_{\max}$, while Figure 3.8 displays the progression of

time step size during each simulation. Additionally, Table 3.3 presents the corresponding computational times obtained by simulating the models with one node on the workstation.

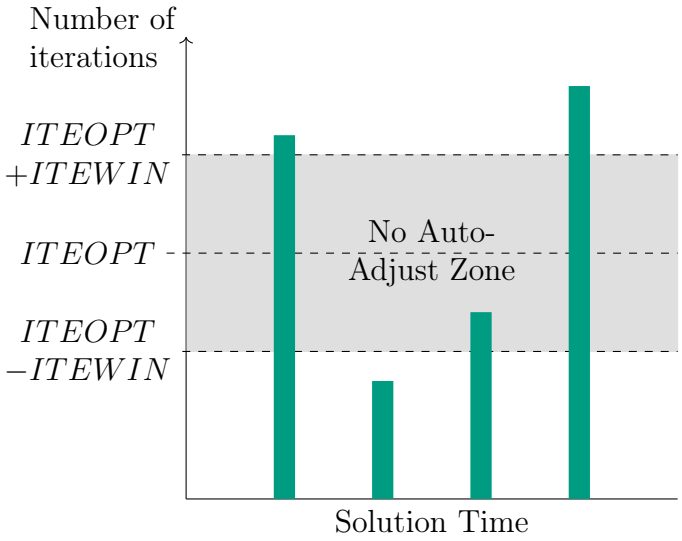


Figure 3.6: Iteration window defined by $ITEOPT$ and $ITEWIN$ (adapted from [15])

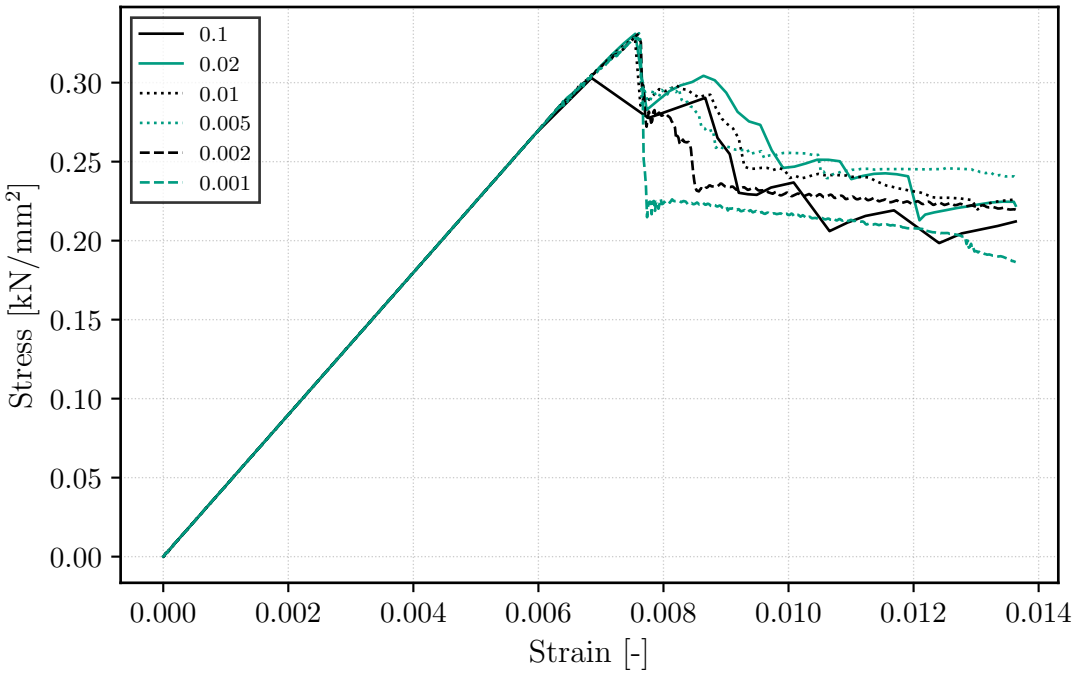


Figure 3.7: Stress-Strain curve with different maximum time steps

The stress-strain curves from the different simulations reveal that the oscillations have disappeared, and the curves are much smoother before and after failure. The various samples show similar behaviour in the pre-failure region, as well as shortly after failure, with the exception of the simulation featuring the maximum time step size of 0.1 ms. This larger time step size resulted in premature failure compared to the other samples with a smaller time steps because the large time step caused the point of maximal stress to be skipped. Furthermore, Figure 3.8 shows that the time step needed to be adjusted

frequently during the simulation, which is a time-consuming process requiring the stiffness matrix to be reformed at every adjustment of the time step. This frequent adjustment is also the reason why the simulation with the larger maximum time step size of 0.1 ms has a longer runtime compared to the simulation with a smaller maximum time step size of 0.02 ms, which required far fewer time step adjustments. As a result, this maximum time step size is not suitable for this problem. For the other simulations, the computing time grows approximately linearly. Notably, at a maximum time step size of 0.005 ms, the time step no longer needs to be adjusted during simulations.

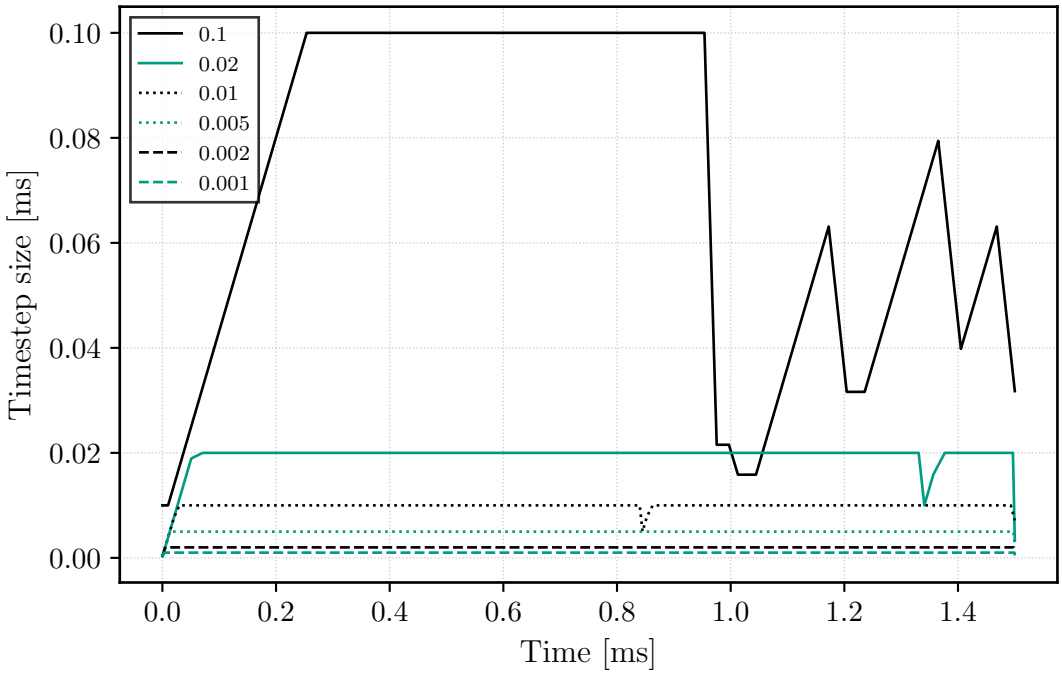


Figure 3.8: Time step progression for different maximum time steps

Table 3.3: Computing time for different maximum time steps

$\Delta t_{\max}$ [ms]	Computing time	Problem cycles
0.1	1min 22s	536
0.02	1min 12s	433
0.01	2min 49s	1217
0.005	4min 24s	1781
0.002	8min 53s	3260
0.001	17min 0s	6231

For future simulations, the maximum time step size of $\Delta t_{\max} = 0.005$ ms will be employed since it is the first time step where no adjustments are required during simulations. Additionally, this value still has good computational time of 4 minutes and 24 seconds when simulated with one node on the workstation.

3.3.2 Automatic Implicit to Explicit Switching

The implicit analysis is now tested on a sample with defects from the existing models, but unfortunately, the simulation terminates in error after reaching the failure point. This is because the samples with defects exhibit a stronger non-linear behaviour compared to a defect-free sample, which are essentially perfect compression rods. The introduction of defects into the system can lead to instabilities in the implicit analysis, resulting in convergence problems. To address this issue and keep the implicit method, an automatic implicit-to-explicit switch is implemented in the simulation. This allows the simulation to continue running even if the implicit analysis fails to find an equilibrium after several iterations, by switching to explicit analysis for finishing the simulation. The auto-switch simulation is set up with the help of chapter 16 in the Ansys Manual [16]. It is recommended to utilize Mortar contacts when implementing this automatic switch, even though they can be computational expensive for explicit analysis.

Figure 3.9 shows the stress-strain curves from both an implicit and an explicit simulation using the same settings. The stress-strain curves are calculated with the resultant forces in y-direction from the cross-section in the middle of the sample as in Section 3.2. The samples have no defects with a stacking sequence $[0°/-45°/90°/+45°]_{2s}$, size 22.0 mm × 172.0 mm × 2.0 mm, and same boundary conditions as the model in Section 3.2. The elements have a size of 2 mm × 2 mm × 0.115 mm for CFRP TSHELL and 2 mm × 2 mm × 0.01 mm for cohesive solid. The simulation time is $T = 1.5$ ms. The implicit simulation settings are identical to those from Section 3.3.1, with a maximum time step size of $\Delta t_{\max} = 0.005$ ms. The explicit simulation uses the same model and settings but with a fixed time step of $\Delta t = 5.6 \cdot 10^{-5}$ ms. The explicit simulation has a computational time of 1 minute and 6 seconds, which is significantly lower than the duration of the implicit simulation with 4 minutes and 24 seconds. Both simulation are run on the workstation with one node.

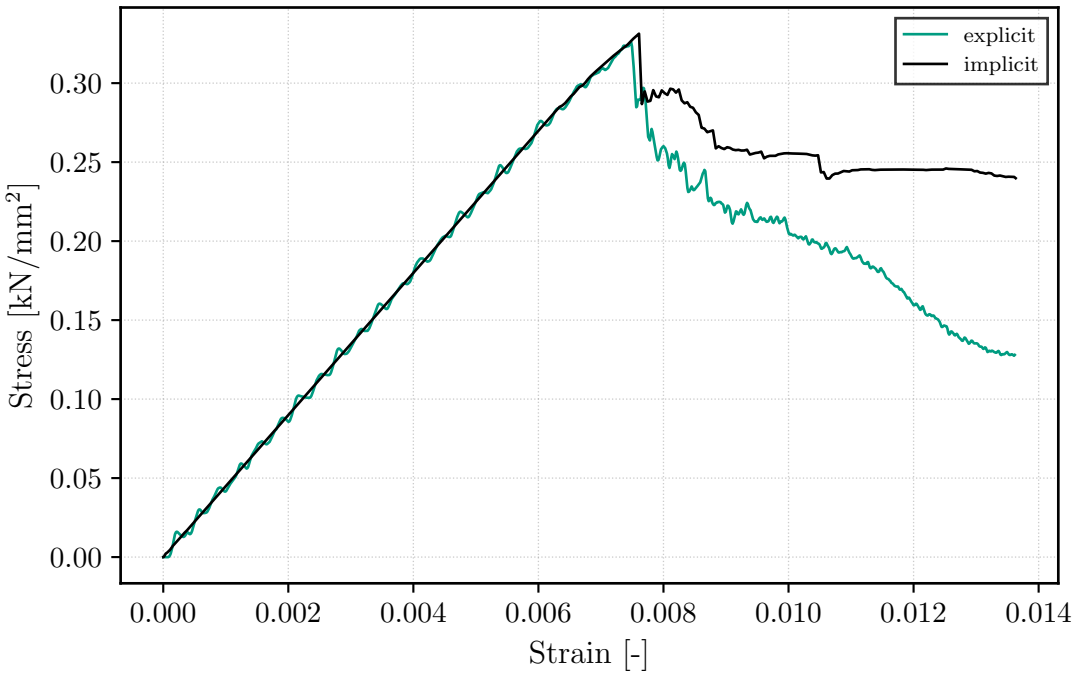


Figure 3.9: Stress-strain curve comparison for explicit and implicit

The behaviour until failure is similar for the implicit and the explicit model with explicit displaying the typical oscillations due to increased loading speed. However, the stress-strain curve of explicit analysis drops significantly lower after failure and is nosier than the implicit one. Unfortunately, this behaviour must be accepted to run the implicit method until failure while still finishing the simulation with explicit time integration without any error terminations. Importantly, four out of five target variables are extracted during the implicit analysis, with the exception being *area under curve after failure*. The automatic switch offers a reasonable compromise between maximizing information extraction and ensuring data quality. The computational time for simulations with an auto-switch will be in between the explicit and implicit times. The complete control cards for the auto-switch simulation, which will be used from now, are presented in Appendix A.

3.3.3 Mesh Reduction

To minimize computational time, the mesh is reduced to only the middle section of the sample, which is not fixed and absorbs all the loads. So far, the mesh was a representation of the sample with a size of 22.0 mm $\times$ 172.0 mm, that is used for experimental tests. However, areas which are subjected to clamping conditions do not add any value. Consequently, the clamping regions are neglected, and the boundary conditions are applied solely to the edges of the reduced mesh resulting in a mesh a size of 22.0 mm $\times$ 22.0 mm, corresponding to $\Omega = [0, 22] \times [0, 22] \times [0, 2]$ for 16 plies. This leads to $\Gamma_D = [0, 22] \times \{0\} \times [0, 2] \cup [0, 22] \times \{22\} \times [0, 2]$ and $\Gamma_N = [0, 22] \times (0, 22) \times [0, 2]$.

The smaller mesh yields identical results but with a significantly lower number of nodes and elements, resulting in a substantial reduction in computational time by a factor of four. This is because for the larger mesh the stiffness matrix also has to be set up and solved for the nodes in the boundary conditions, even if they do not contribute any meaningful results. Figure 3.10 displays a comparison between the old and the new smaller mesh, with the green area highlighting the region where nodes experience the applied boundary conditions.

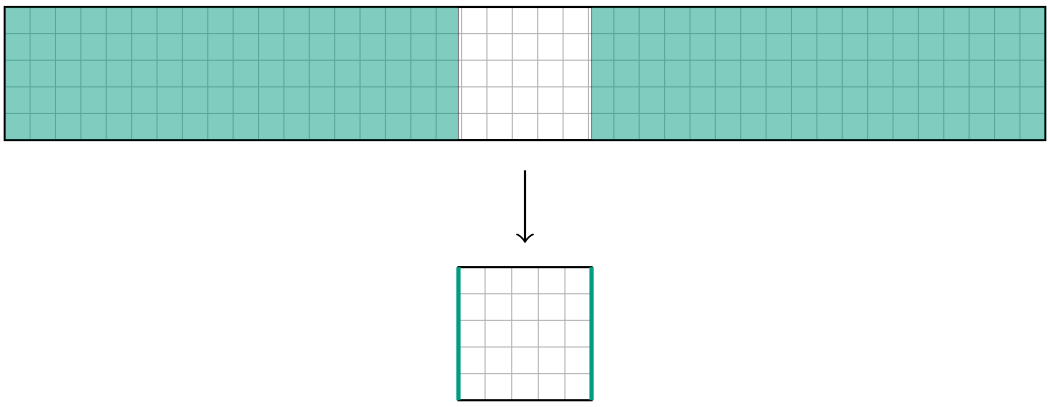


Figure 3.10: Mesh reduction

3.4 Tuning of the Parameters of MAT_54

In this section, the parameters of the new material model MAT_54 are adjusted to match the simulation results as closely as possible to experimental data obtained from

a compression test. However, care is taken to ensure that the parameters do not assume any unphysical values. Figure 3.11 shows a comparison of the stress-strain curves between the experimental and simulations results from the model in Section 3.3.1 with $\Delta t_{\max} = 0.005$ ms. The experiment and simulation is conducted on a sample with 16 plies and a stacking sequence $[0°/-45°/90°/+45°]_{2s}$. With the current parameters for MAT_54, the simulated samples have too low a strength compared to real-world samples. Additionally, the Young's modulus is not set correctly, which can lead to significantly deviating results even if the strength parameters are adjusted.

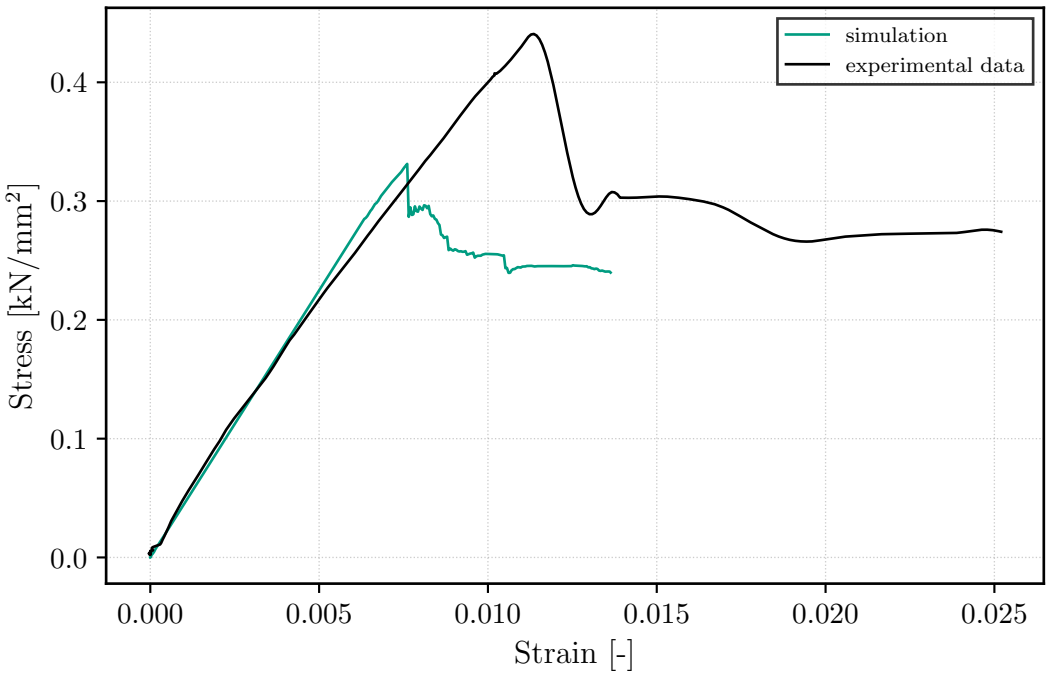


Figure 3.11: Stress-strain curve comparison for implicit analysis and test data

Initially, an optimization process for some of the material parameters is performed using LS-OPT, a graphical optimization tool with an interface to LS-DYNA that can solve many optimization problems. With LS-OPT, it is possible to solve the non-linear inverse problem of identifying material parameters to model simulation results against experimental data [43]. However, unlike individual simulations with LS-DYNA, LS-OPT has to run on the local computer increasing the computational time significantly. To mitigate this, the size of the elements is increased from 2.0 mm × 2.0 mm to 3.6 mm × 3.6 mm. The thickness of the elements stays the same with 0.115 mm for the CFRP TSHELL and 0.01 mm for the cohesive solid elements. Additionally, only the inner part of the sample with 22.0 mm × 22.0 mm is modelled with boundary conditions applied to the edges. For an element size of 3.6 mm and 16 plies, the mesh has 1568 nodes with 448 nodes experiencing a boundary condition. In the first optimization, only the parameters for the longitudinal Young's modulus EA and compressive strength XC are modified, as they have the highest influence in the first part of the simulation (Figure 2.7 in Section 2.4.3). In a second optimization, the values for EA and XC are fixed and the parameters that influence mostly the behaviour after failure are adjusted, including EFS, $SLIMC1$, $SLIMC2$ and $SLIMS$. In addition, SC and YC are also optimized. The optimizations yielded promising results, closely matching experimental data.

However, since the simulations for the data of the ML models will not be carried out in this element size, several simulations were conducted with different element sizes and the optimized material model to evaluate whether the material model is suitable for different element sizes. The resulting stress-strain curves calculated using equations (3.1) and (3.2) from the cross-section in the middle of the sample are displayed in Figure 3.12, with curve labels indicating the element size. All other settings are kept consistent across these simulations, with a mesh size of 22.0 mm × 22.0 mm × 2.0 mm, $T = 2.25$ ms, $\Delta t_{\max} = 0.005$ ms for the implicit simulation part, and $\Delta t = 5.6 \cdot 10^{-5}$ ms for the explicit simulation part.

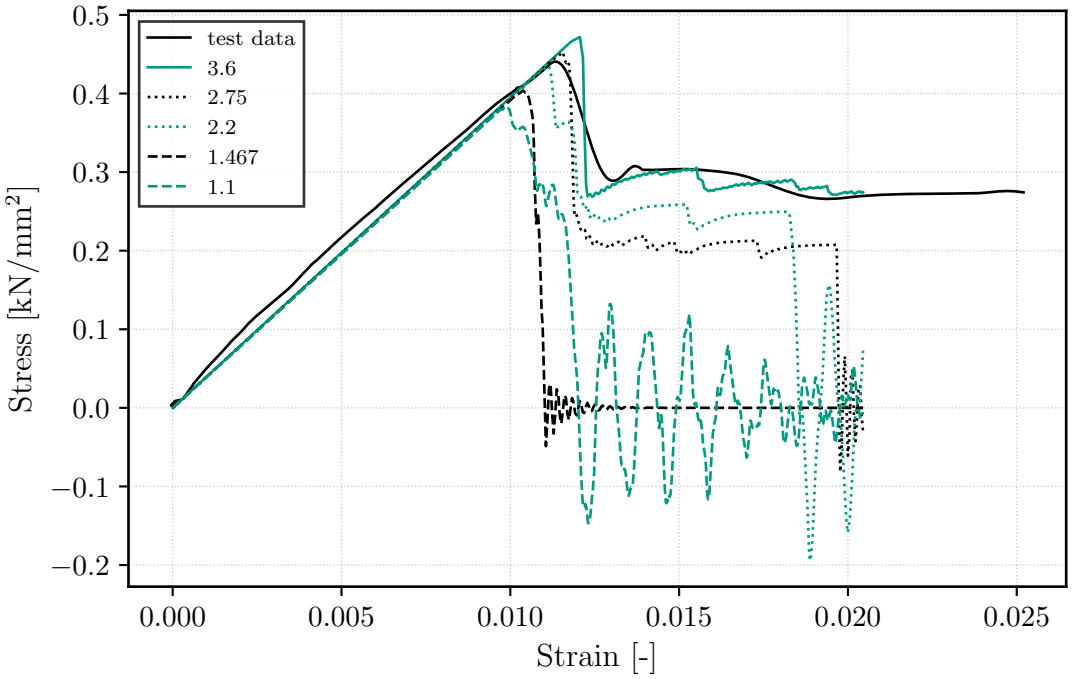


Figure 3.12: Results for different element sizes with optimized material model

A significant dependency is observed between the simulation results and the element size. The mesh with an element size of 3.6 mm, which has been optimized, is close to the experimental data. In contrast, simulations with smaller element sizes fail earlier and at lower stresses. Additionally, until an element size of 2.75 mm, the simulations run purely implicitly without switching to explicit. Given this dependence on element size, it is clear that the optimization for the material model must be carried out using the element size that will be later used for generating the data for the ML models. Therefore, the decision is made to continue with an element size of 1.1 mm × 1.1 mm × 0.115 mm for TSHELL and 1.1 mm × 1.1 mm × 0.01 mm for solid elements, resulting in 14112 nodes, 6400 TSHELL elements for CFRP and 6000 cohesive elements for 16 plies. This element size is deemed sufficient to achieve good accuracy. The other settings, such as the mesh size of 22.0 mm × 22.0 mm × 2.0 mm, stacking sequence $[0°/-45°/90°/+45°]_{2s}$, $T = 2.25$ ms, $\Delta t_{\max} = 0.005$ ms for implicit and $\Delta t = 5.6 \cdot 10^{-5}$ ms for explicit, are left unchanged.

Due to the increased computational time associated with a finer mesh, the optimization with LS-OPT is no longer a viable option. As a result, the adjustment of the material parameters is performed manually through trial and error to run the simulations on the workstation with more computing power. The values for the parameters YC and SC are taken from existing literature [44] to speed up the process. Additionally, the value for EA

can be adopted from the previous optimization since there is no element size dependency in the first gradient of the stress-strain curve. This leaves EFS, $SLIMC1$, $SLIMC2$ and $SLIMS$ open for manual adjustment. The value for EFS is set comparatively high to avoid element deletion. If an element is deleted due to failure, the adjacent elements in the same row are also deleted, leading to system instability. The $SLIM$ parameters are non-physical and therefore have no upper or lower bounds for their adjustment. After conducting some trial and error, a suitable set of parameters is identified that matches the simulation results close the experimental data. All the parameters for MAT_54 and their used values are listed in Appendix B.

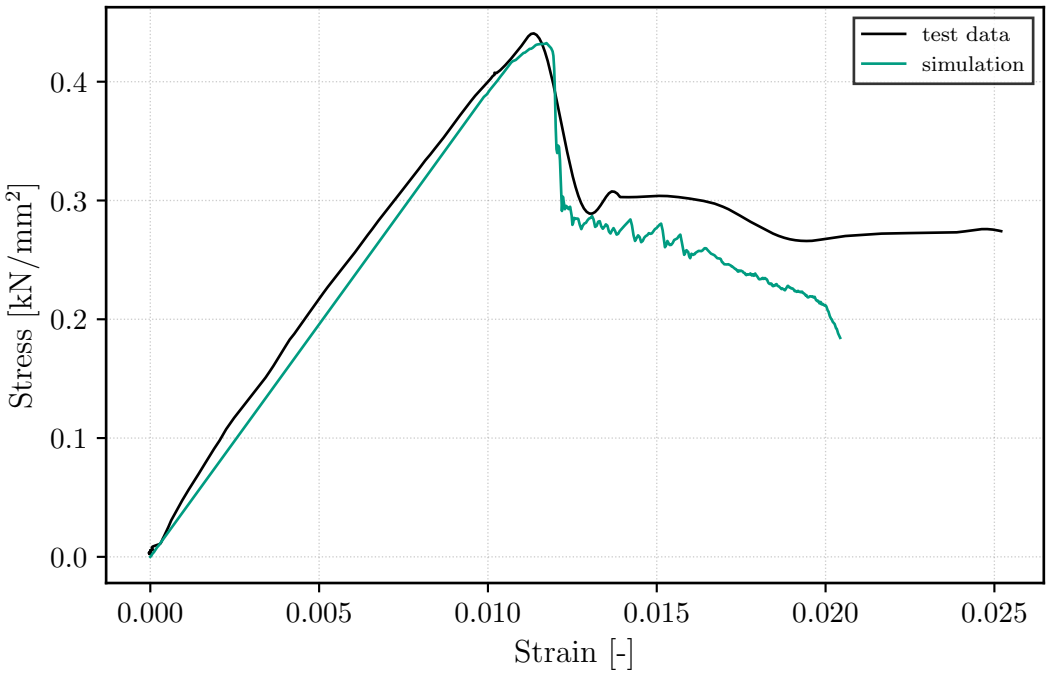


Figure 3.13: Stress-strain curve for optimized material card

Figure 3.13 shows a comparison between the stress-strain curves of the finalized simulation and the experimental data. The stress-strain curve of simulation model fails slightly later and drops off more steeply after failure. Additionally, the part after failure exhibits oscillations characteristic of explicit analysis and it dips under the experimental curve and eventually falls off towards the end. Due to time constraints, further optimization of the material model is not feasible. However, the simulation represents a significant improvement over the previous model.

3.5 Mesh Convergence

In this section, the quality of the simulation model with respect to the mesh discretization error is assessed by conducting several simulations with varying element sizes, ranging from smaller to larger than the selected element size of 1.1 mm × 1.1 mm. The stress-strain curves obtained from these simulations are displayed in Figure 3.14. The settings are identical to those used in the simulation from Figure 3.13, with a mesh size of 22.0 mm × 22.0 mm × 2.0 mm, TSHELL thickness 0.115 mm, cohesive element thickness 0.01 mm, stacking sequence $[0°/-45°/90°/+45°]_{2s}$, $T = 2.25$ ms, $\Delta t_{\max} = 0.005$ ms for

implicit, and $\Delta t = 5.6 \cdot 10^{-5}$ ms for explicit. The computational times for each simulation are listed in Table 3.4. All simulations are executed on the workstation with one node.

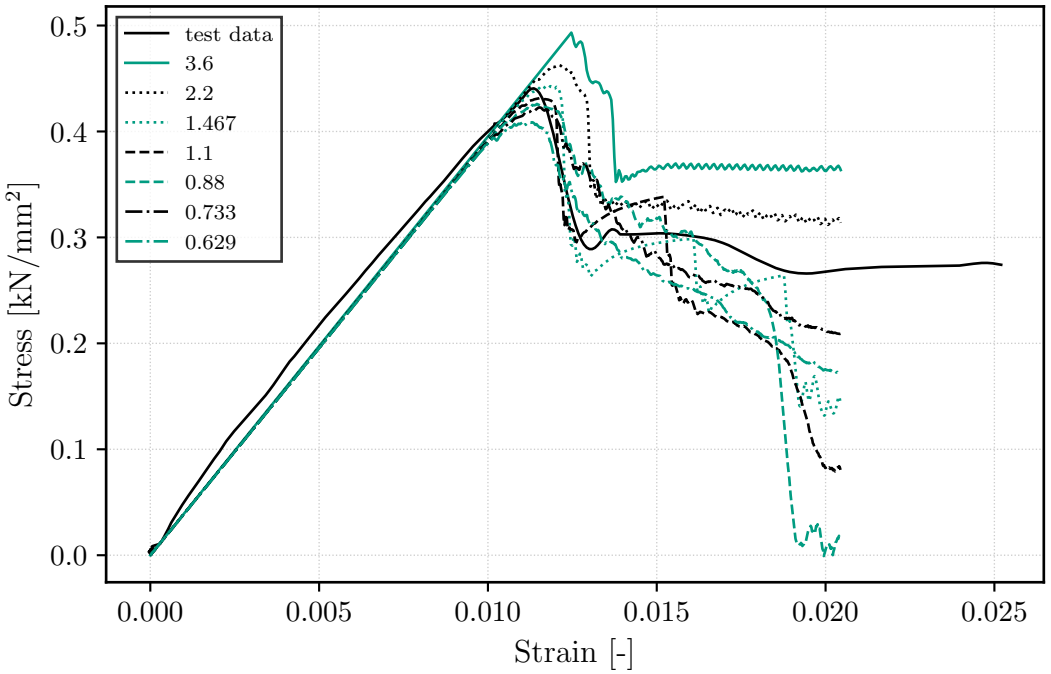


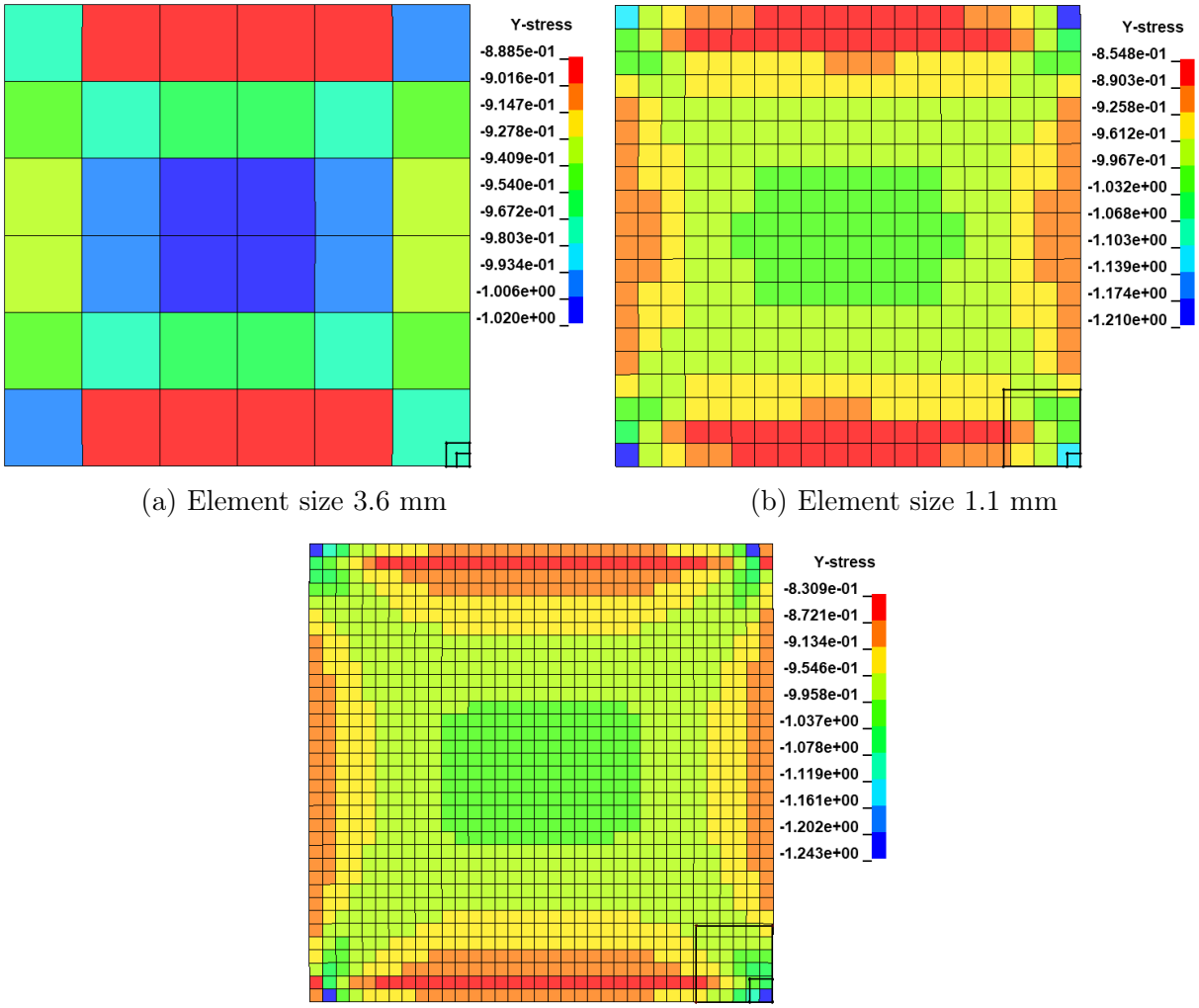
Figure 3.14: Stress-strain curves for different element sizes

Table 3.4: Computing time for different element sizes

Mesh size [mm]	Computing time
3.6	1 min 10 s
2.2	1 min 52 s
1.467	3 min 29 s
1.1	5 min 33 s
0.88	5 min 47 s
0.733	8 min 7 s
0.629	8 min 12 s

The stress-strain curves obtained from the simulations indicate that the simulation model has not yet reached full convergence, as the point of failure decreases with a smaller element size. Furthermore, the shape of the curve after failure is highly dependent on the element size. The strain value for the maximum stress remains relatively constant across an element size of 1.1 mm and smaller. Similarly, the behaviour of the model at the beginning is identical regardless of element size. These findings suggest that the element size and material model MAT_54 are interdependent, making it impossible to optimize one independently of the other.

One explanation for the non-convergence is that stress peaks in elements at the edges which become more pronounced as the element size decreases. Figure 3.15 shows the resultant stress values in y-direction for each element in ply 16 at $t = 1.0$ ms for different



(a) Element size 3.6 mm

(b) Element size 1.1 mm

(c) Element size 0.63 mm

Figure 3.15: Y-stress for different meshes at $t = 1.0$ ms

element sizes. In the right corner of each mesh, the size of one element from the other element sizes is displayed. The stress plots reveal that finer meshes have higher stress concentrations at the corners. For example, the mesh with an element size of 1.1 mm has lower stress values (1.21 kN/mm^2) at its corner compared to the mesh with an element size of 0.63 mm (1.24 kN/mm^2). However, this also means that these elements fail first, triggering a chain reaction that can cause the entire row to fail. In contrast, larger elements lead to an averaging of stresses over larger regions and present lower stresses. Consequently, samples with finer meshes tend to fail earlier due to the earlier failure of individual elements.

Due to time constraints for this work, the current element size of 1.1 mm is retained, as optimizing a new material model for a smaller mesh size would be too time-consuming. Additionally, generating data for the ML model would take significantly longer for simulations with a finer mesh, resulting in a substantial decrease in the amount test data that can be generated.

4 Generating the Data

In this section, the process for generating data for the ML models based on simulation results is described. First, multiple samples are created using a Python script and then simulated with LS-DYNA. From the simulation results, the target variables are extracted, and together with the sample configurations, are used to train the RF and SVR models to predict mechanical properties of tape-layered specimens. The workflow for creating and evaluating the data, which will be described in the following sections, is shown in Figure 4.1.

The samples are set up with the reduced mesh and its boundaries Γ_D, Γ_N from Section 3.3.3. One side of the mesh is fixed, while the other side experiences a motion, resulting in the Dirichlet boundary condition. The CFRP is modelled with three-dimensional TSHELL elements with a size of 1.1 mm $\times$ 1.1 mm $\times$ 0.115 mm as the plies have a thickness of 0.115 mm. The cohesive layers are 0.01 mm thick, resulting in an element size for the three-dimensional solids of 1.1 mm $\times$ 1.1 mm $\times$ 0.01 mm. The optimized material model MAT_54 from Section 3.4 is employed for the CFRP plies and the cohesive layers are modelled with MAT_240 as in the existing model from Section 3.2. A Mortar contact is used between the TSHELL elements of the CFRP plies to prevent penetrations in case the cohesive layers fail. The simulation will utilize an auto-switch from implicit to explicit time integration with $\Delta t_{\max} = 0.005$ ms for implicit and $\Delta t = 5.6 \cdot 10^{-5}$ ms for explicit. The simulation time is set to $T = 2.5$ ms.

The goal for generating samples is to introduce randomness and diversity in their configurations, effectively covering a wide range of defects. The samples will have either 8, 12, or 16 plies, resulting in a total sample thickness of 1.0 mm, 1.5 mm, or 2.0 mm. The stacking sequence will be random but symmetric and balanced, meaning that for every 0 degree ply, there exists a 90 degree ply in the stacking sequence. The same applies to $+45$ and -45 plies. Additionally, each sample can have either one or two defect, but only one per ply. Defects can be triangular or rectangular in shape. The rectangular defect can represent either a macroscopic pore (1.1 mm $\times$ 1.1 mm) or extend through the entire sample, representing a defect between tapes. Furthermore, a defect can be either a gap or an overlap between tapes. For gaps, the CFRP ply thickness is reduced to 0.015 mm, while for a overlaps, it increases to 0.24 mm.

4.1 Generating the Samples for the Simulations

The first step towards generating the data for ML models involves creating many different samples with various defects and their associated input files for simulations, from which the data will be extracted. Since all simulation settings are identical except for the mesh, only a single simulation file for the mesh needs to be generated for each mesh configuration. These files for LS-DYNA are created using the Python script *output_files_generator.py*, which aggregates many individual scripts. The entire process for generating simulation files for 1000 samples took approximately 44 minutes and 42 seconds.

The script *simple_model.py* begins by creating nodes and elements for a random stack with a random number of plies without any defects selected from predefined configurations. Subsequently, it randomly determines the properties of defects and their location within samples. Two additional scripts, *add_defect_triangle.py* and *add_defect_rectangle*, ap-

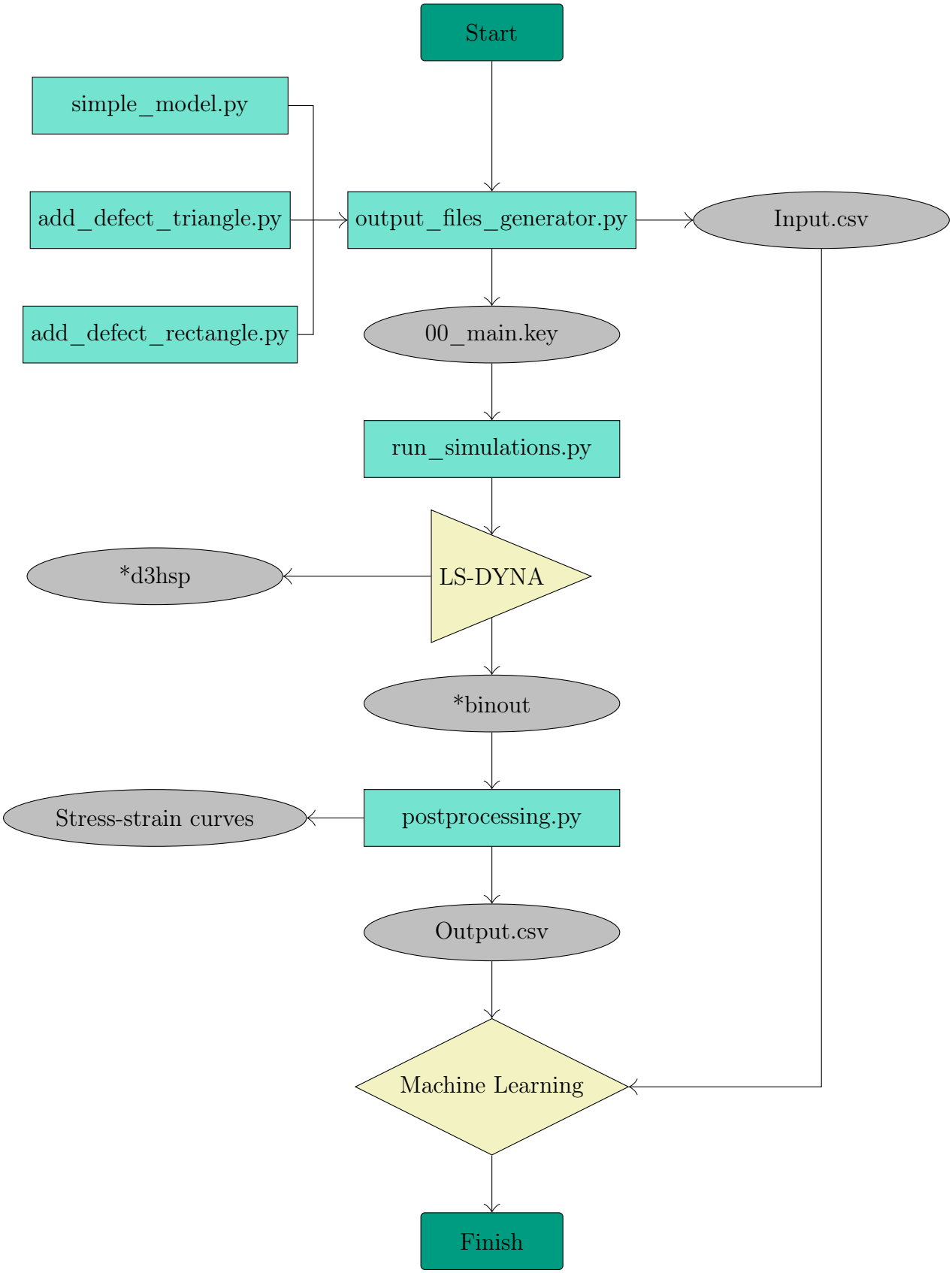


Figure 4.1: Workflow for the generation of the data

ply the random defect to the sample by modifying the z-coordinates of the affected nodes. For gap defects, the z-coordinate is decreased, while for an overlap defect, it is increased by the thickness of a ply. Additionally, the fibre orientation must be adjusted for triangular defects. In this work, potential changes in local fibre volume content near defects are not considered.

Every sample is identified by an ID, which can be used to access the model creation settings. Subsequently, element and node information for the completed mesh are written into a simulation file named *00_main.key*. This file is then saved in a separate folder named after the ID of each sample. Furthermore, a file called *Input.csv* is created during the mesh generation process, containing all relevant information about every single sample. This file will later be used as input for ML models. Table 4.1 shows an excerpt from this file. The first column contains the ID, followed by columns *1* to *8* listing the stacking sequence. Since the stacking sequence is symmetrical, it is sufficient to list only half of it. The column *ply_defect1* specifies the ply number where the first defect occurs, while *type_of_defect1* indicates whether it is a gap (0) or an overlap (1). The subsequent columns, specify the size and location of the defect within a ply. The columns *x_1* and *y_1* define a point p positioning the defect in the 2D plane. If the defect is triangular, the opening angle is listed in *angle1*, while if it is rectangular in shape, the columns *width1* and *length1* give the dimensions. By specifying one point and the dimensions, the defect is clearly determined.

The columns *c1.1* to *c1.7* indicate the corner points of the defect. With this information, the mesh can be manually rebuilt if files are lost. For triangular defects, the six corner points occupy positions *c1* to *c3* and *c5* to *c7*. The four corner points for a rectangular defect are listed in columns from *c1* to *c4*. Figure 4.2 illustrates the characterization of a defect based on its corner points. The point p corresponds to positions *c2* and *c7* for the triangular defect. The subsequent columns list the properties for a second defect. Empty cells are left blank.

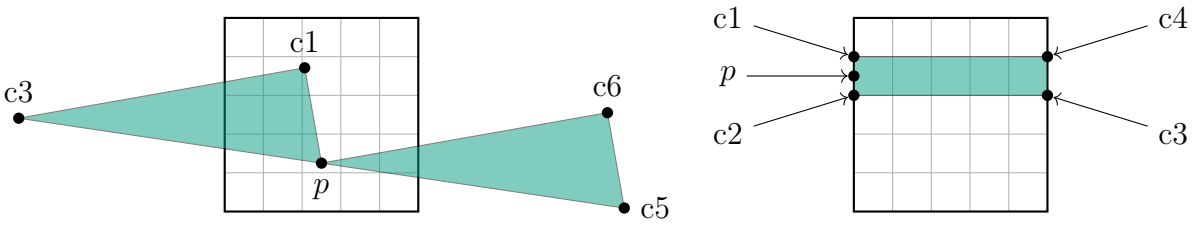


Figure 4.2: Characterization of the defects

Table 4.1: The first six IDs of the file *Input.csv*

ID	1	2	3	4	5	6	7	8	ply_defect1	type_of_defect1	x_1
1	-45	0	45	90					6	1	-0.6
2	90	0	90	0	45	-45			11	0	18.4
3	90	0	90	0					2	1	11.1
4	-45	90	45	0					2	1	11
5	45	-45	-45	90	45	0			4	0	93
6	0	45	-45	90	90	0	45	-45	16	0	10.3

y_1	angle1	width1	length1	c1.1x	c1.1y	c1.2x	c1.2y	c1.3x	c1.3y	c1.4x
-0.3		1.7	32.81	-1.20	0.30	0.00	-0.90	23.20	22.30	22.00
17.6		1.1	1.1	17.85	17.60	18.95	17.60	18.95	18.70	17.85
16.8		1.1	1.1	10.55	16.80	11.65	16.80	11.65	17.90	10.55
3.2		1.1	1.1	11.00	3.75	11.00	2.65	12.10	2.65	12.10
7.2	5			92.39	21.19	93.00	7.20	-67.48	14.21	
0		1.1	32.21	9.75	0.00	10.85	0.00	10.85	32.21	9.75

c1.4y	c1.5x	c1.5y	c1.6x	c1.6y	c1.7x	c1.7y	ply_defect2	type_of_defect2
23.50								
18.70							1	0
17.90							7	0
3.75								
	252.87	14.18	253.48	0.19	93.00	7.20		
32.21							9	1

x_2	y_2	angle2	width2	length2	c2.1x	c2.1y	c2.2x	c2.2y	c2.3x
214.00	7.80	2.2			213.73	21.80	214.0	7.80	-150.6
7.10	-32.40	9.9			-6.85	-33.61	7.10	-32.40	0.07
143.00	-103.80	4.1			153.25	-94.26	143.00	-103.8	286.3

c2.3y	c2.4x	c2.4y	c2.5x	c2.5y	c2.6x	c2.6y	c2.7x	c2.7y
14.80			578.4	14.80	578.6	0.80	214.0	7.80
-113.5			0.18	47.52	14.13	48.73	7.10	-32.40
-237.2			9.92	39.16	-0.32	29.62	143.0	-103.8

4.2 Running the Simulations and Postprocessing

The next step involves running simulations for each sample and extracting necessary data from the results. The script *run_simulation.py* sends the simulation files *00_main.key* one after another to the workstation, which runs LS-DYNA. During the simulation, LS-DYNA generates multiple files, but most of them are automatically deleted by the Python script due to the workstation's maximum file limit. However, certain critical files remain, including **d3hsp* and **binout*. The **d3hsp* file contains information about the simulation process, such as computational time, time step details, or whether the simulation completed successfully. From the 1000 simulations conducted, 17 terminated with an error and 25 ran fully implicit without a switch to explicit. The reason for the error termination or the full implicit run could not be derived from the properties of the samples. The average computational time per simulation was 3 minutes and 22 seconds. The simulation with the lowest runtime of 1 minutes and 47 seconds experienced an early switch to explicit, whereas the one with longest time of 19 minutes 10 seconds was a full implicit run.

The **binout* file contains all the simulation results for the nodes and elements, which allows the script *postprocessing.py* to extract relevant data. This includes the course of the force of the cross-section in y-direction, the displacement in y-direction of one boundary node, and the area of the cross-section. With these outputs, stress and strain for each time step can be calculated using equations (3.1) and (3.2) in Section 3.2. After performing these calculations, all target parameters for the ML models can now be extracted from the simulation data.

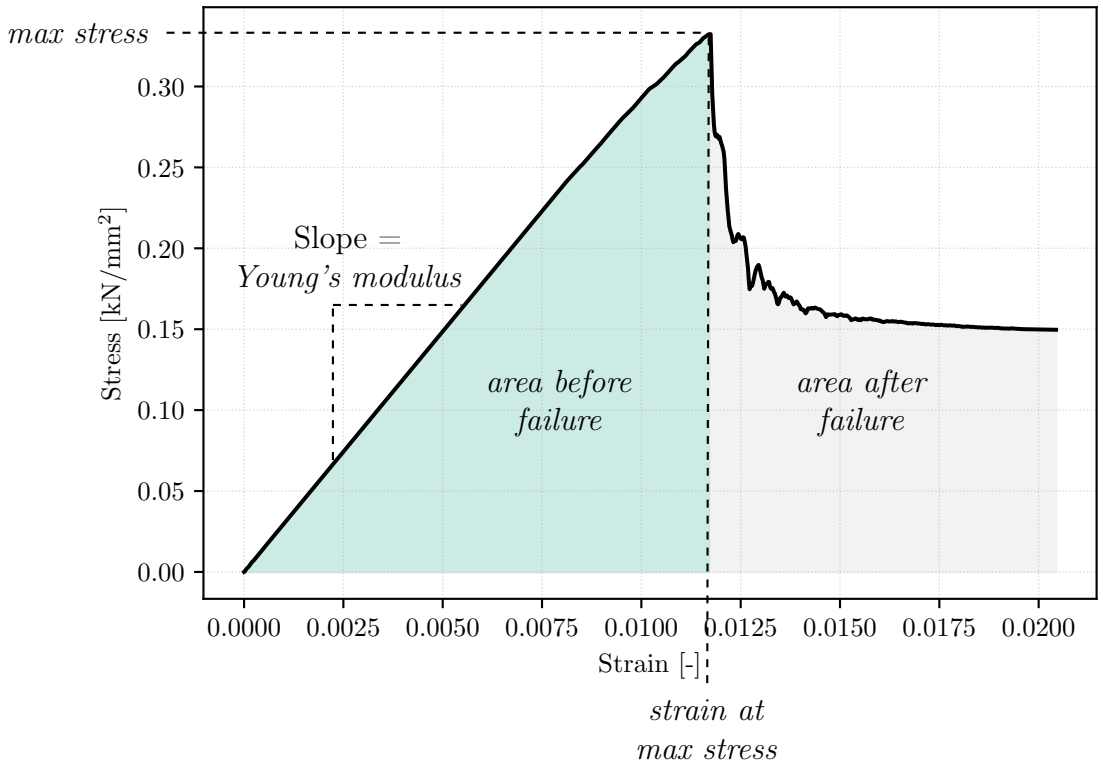


Figure 4.3: Relationship between the target variables and a typical stress-strain curve

The relationship between the five target variables and the stress-strain curve is shown in Figure 4.3. The variables *max stress* and *strain at max stress* are chosen as targets because they represent the strength of a sample and how much strain it can take before it failing. The target *Young's modulus* is an important value, as it measures the stiffness and is needed to describe the law of elasticity (see Section 2.3). It can be calculated by determining the slope during the elastic part of the stress-strain curve until failure. The variables *area before failure* and *area after failure* are used as targets to extract some information about the course of the stress-strain curve. For example, a low value of *area after failure* indicates that the plateau in the stress-strain curve after failure is very low or non-existent. The variable *area after failure* is only calculated until the stress drops below 0.05, as some samples have strong oscillations or negative stresses after failure. Both *area before failure* and *area after failure* are calculated with the integral function from the Python scipy library [45]. Additionally, the stress-strain curve for each sample is plotted, which will be later used in Section 4.3.1 to assess the quality of the simulation results. The values of the target parameters are saved in the file *Output.csv*, as shown in Table 4.2.

Table 4.2: The first six IDs of the file *Output.csv*

ID	max stress	strain at max stress	Young's modulus	area before failure	area after failure
1	0.3724	0.009607	39.06	0.0018032	0.0002437
2	0.3986	0.009607	42.54	0.0019536	0.0010232
3	0.2877	0.009352	48.82	0.0015434	0.0003704
4	0.2102	0.005516	39.26	0.0005964	0.0013951
5	0.2884	0.008789	34.05	0.0013060	0.0010146
6	0.4243	0.011384	38.64	0.0024980	0.0026363

4.3 Analysing the Data

The next step involves checking the simulation results from the samples for outliers, unusual behaviour, unsuitable results, or numerical difficulties that could potentially negatively impact the accuracy of the ML models.

4.3.1 Stress-Strain Curves

The first step in assessing the simulation results involved manually checking all stress-strain curves for unusual behaviour. It is observed that simulations for almost all samples with 8 plies and a thickness of 1 mm exhibit numerical difficulties, manifesting as high oscillations or erratic behaviour in the stress-strain curves compared to samples with 12 or 16 plies. Examples are shown in the Appendix C. This behaviour is explained by the thinness of the samples with 8 plies which can lead to failure through buckling under compression, a mode of failure distinct from the samples with 12 or 16 plies, which experience failure through fibre rupture. The behaviour of the 8 plies is physically explainable but LS-DYNA has difficulties in solving this problem which is shown through the oscillations distorting the target variables such as *max stress* and *area after failure*. To ensure uniformity in failure modes across the dataset, all 8-ply samples are removed, leaving 662 samples.

Additionally, it was noticed that 23 stress-strain curves exhibit very low maximum stresses with high plateaus after failure for both 12 and 16 plies samples. Upon reviewing the settings for these samples, it is observed that all of them have a stacking featuring only 45 degree plies. In these samples, the fibres take almost no load during compression, with forces primarily absorbed by the matrix. As a result, these samples have low strength and fail due to matrix failure at the point of failure. Given that this behaviour is physically plausible, the samples with only 45-degree plies in their stacking are left in the dataset.

4.3.2 Extreme Values

In the following, the samples with the highest and lowest value for each target variable are checked to detect unusual behaviour. For the extreme values of *max stress*, *strain at max stress*, *Young's modulus* and *area before failure*, a clear pattern in the settings for the samples is recognisable. The highest values for these variables are found in samples with only 0 and 90 degree plies. In contrast, the lowest of the above values is taken by samples with only 45 degrees in the plies. An exception are the extreme values for *area after failure*. The highest values has a sample with a balanced stacking of 12 plies and a gap in a 45 degrees ply. The lowest value is taken by a sample with 16 plies, a fairly balanced stacking and an overlap in a 45 degrees ply. The reason for the extreme values can not be derived from the settings of the samples. Since all the samples with extreme values do not exhibit unusual behaviour, they can remain in the data set.

4.3.3 Outlier Detection

In this section, an outlier detection is performed to identify potential outliers and their causes. The outlier detection is executed in Python using IF [46] and LOF [27] (see Outlier Detection in Section 2.5.1) from the scikit-learn library with the default settings.

Out of 662 samples, IF identifies 92 as outliers, which contain extreme values due to samples with only 45 or 0/90 degree plies in their stacking sequence. These stacking sequences are relatively rare due to the random generation process and symmetry requirement. Additionally, IF also declares almost all samples that underwent a pure implicit run without a switch to explicit as outliers. However, it is unclear which configuration of a sample leads to a pure implicit run.

In contrast, the LOF method delivers different results compared to IF. It does not classify either the stacking sequences with only 45 or 0/90 degree plies, nor the purely implicit simulations, as outliers. This is because the LOF method defines an outlier by how isolated it is (see Section 2.5.1). Since there exist about 20 samples with a pure 45 or 0/90 degree stacking, these observations are not isolated and therefore are not considered outliers. Instead, LOF seems to have detected outliers that have unique combinations of features and cannot be recognized as part of a pattern. Furthermore, LOF detected 114 outliers, which represents about 17% of the 662 samples.

One reason for LOF and IF declaring around 14 to 17 percent as outlier might be the randomness in the data. This can result in many unique combinations, especially with a smaller amount of data points. A larger dataset would likely result in fewer outliers being declared. A total of 19 samples are identified as outliers by both IF and LOF. A closer look at their stress-strain curves shows that most of these samples exhibit some numerical problems during the simulations, such as oscillations in the stress-strain curves. Despite

the apparent anomalies, no clear correlation between specific configurations and these issues could be determined. Moreover, only a small fraction of the samples are affected by this problem meaning that all the samples are left in the dataset. This results in 662 individual samples being available for model design.

5 Development of the Machine Learning Models

In this section, RF and SVR models are developed and assessed according to their performance. This involves finding a suitable format for the dataset through feature engineering. The models will use a training test split of 80:20 and their performance is evaluated with the metrics MAE (equation (2.42)) and MAPE (equation (2.43)) for all five target variables *max stress*, *strain at max stress*, *Young's modulus*, *area before failure*, *area after failure* and the overall model. The performance of the model is always assessed on the unknown test dataset. The models are build in Python with the scikit-learn modules for RF [47] and SVR [48].

5.1 Data Preparation

Before training the RF and SVR models, it is essential to prepare and alter the data. Currently, the *Input.csv* file contains numerous missing values and unnecessary attributes, making it unsuitable for ML. The first step in addressing this issue is feature selection, which involves identifying and removing unnecessary columns. The file *Input.csv* contains all the information of the samples, which are partially dependent on each other and therefore redundant. As mentioned in Section 4.1, the columns of the corner points are only listed in the dataset to manually rebuild the simulation files if they are lost. Consequently, these columns only contain information that can be extracted from other attributes. As a result, all the columns of the corner points are removed, leading to a great dimensionality reduction from 51 attributes to 23. Additionally, the column containing the sample ID is also removed, leaving 22 relevant attributes. The data in *Output.csv* is already in a suitable format after removing the ID column.

However, the *Input.csv* file still contains many missing values because some samples do not have two defects or 16 plies. To address this issue, feature improvement is necessary to fill in these missing values. For columns describing the defects, such as *ply_ defect, x, y, angle, width* and *length*, the missing values are set to 0. This also makes sense from a geometric perspective, as for example, if there is no defect, the width or length would be 0. The 0 in the *type_ of_ defect* column has a specific meaning, with 0 indicating a gap and 1 indicating an overlap. Therefore, the missing values in these columns are filled with -1. The same applies to the columns that specify the fibre direction of the plies. Here, the value 0 represents a fibre direction and for this reason the missing values are filled in with -999 to clearly indicate missing plies. The current status of the table is shown in Table 5.1.

With this preprocessing step complete, the dataset is now ready for training and evaluating the RF model. However, since the SVR is a distance-based model, it requires scaling the data to avoid the domination of large-scaled attributes. Before scaling, the columns with fibre directions for plies have to be converted because the -999 values for missing plies would distort the scaling. Therefore, the fibre directions are converted to labels as only four fibre directions are used in this work. The conversion is as follows

$$-999 \to 0, \qquad -45 \to 1, \qquad 0 \to 2, \qquad 45 \to 3, \qquad 90 \to 4.$$

These labels also consider the angle difference between the fibre directions, which is a 45 degree jump from label to label. With these labels, all the columns can now be scaled using the z-score standardization from equation (2.44) from Section 2.5.1. When scaling, it is essential to only scale the training dataset and then apply the same scaler to the test

Table 5.1: The first six rows of the altered *Input.csv* file

1	2	3	4	5	6	7	8	ply_defect1	type_of_defect1	x_1
-45	0	45	90	-999	-999	-999	-999	6	1	-0.6
90	0	90	0	45	-45	-999	-999	11	0	18.4
90	0	90	0	-999	-999	-999	-999	2	1	11.1
-45	90	45	0	-999	-999	-999	-999	2	1	11
45	-45	-45	90	45	0	-999	-999	4	0	93
0	45	-45	90	90	0	45	-45	16	0	10.3

y_1	angle1	width1	length1	ply_defect2	type_of_defect2	x_2	y_2
-0.3	0	1.7	32.81	0	0	0	0
17.6	0	1.1	1.1	1	0	214.00	7.80
16.8	0	1.1	1.1	7	0	7.10	-32.40
3.2	0	1.1	1.1	0	0	0	0
7.2	5	0	0	0	0	0	0
0	0	1.1	32.21	9	1	143.00	-103.80

angle2	width2	length2
0	0	0
2.2	0	0
9.9	0	0
0	0	0
0	0	0
4.1	0	0

dataset to avoid data leakage from the training to the test data. With these preprocessing steps completed, the input is now ready for training the SVR model.

5.2 Performance on Different Data Sets

In this section, RF and SVR are applied to different datasets that have been altered by various feature engineering techniques. The goal is to find a format for the ML models achieving the best performance. Both RF and SVR are trained with the default settings from scikit-learn. For RF, this means that 100 trees are fitted, with all the features considered for each tree. For the default SVR with a RBF kernel in scikit-learn, the parameters are set to $C = 1.0$ and $\epsilon = 0.1$. The value for γ is scaled according to the input data resulting in $\gamma = 0.05$.

5.2.1 Baseline Performance

In the first step, RF and SVR are trained on the dataset from Section 5.1. These models serve as a baseline performance for comparison with other models that have a modified dataset. The dataset consists of 662 observations, with 22 attributes in total, out of which 530 observations are used for training. The metrics obtained by these models are presented in Table 5.2. RF trains in about 2.42 seconds, while SVR requires only 0.095 seconds.

Table 5.2: Metrics of the baseline performance

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0458	12.26	0.0432	11.69
strain at max stress	0.000 548	5.28	0.000 578	5.69
Young's modulus	3.30	9.44	2.87	8.20
area before failure	0.000 317	16.19	0.000 316	16.42
area after failure	0.000 507	67.33	0.000 563	73.21
overall model	0.669	22.10	0.583	23.04

The metrics reveal that RF and SVR have similar performance. The models performed well in predicting *strain at max stress* and *Young's modulus*, with a MAPE below 10%. However, for *max stress* and *area before failure*, the MAPE is around 12% and 16%, respectively, indicating room for improvement. The target variable *area after failure* exhibits poor performance from both models. It is also the only variable that is extracted during the explicit part of the simulation.
It seems that the ML models have difficulties identifying clear correlations between the sample characteristics and its behaviour after failure. A potential explanation could be that the explicit time integration introduces noise into simulation, distorting the values for *area after failure*. Another possibility is that the highly non-linear behaviour of the model after failure may be difficult for the ML models to predict accurately.

Overall, RF performs better than SVR on three out of five target variables, and it has a better MAPE on the overall model. However, SVR has a better MAE overall, which is due to its better performance on *Young's modulus*. This variable has high absolute values compared to the other targets. Therefore, the overall performance is measured

with MAPE as it is independent of scaling.

Figure 5.1 illustrates the first two upper levels of a decision tree used in the RF model to predict *max stress*. Each node in the decision tree displays the current feature and its threshold, followed by the value of the squared error loss function (equation (2.49)), which is used to assess the quality of the split. Additionally, it shows the number of samples in each subset and the mean value of *max stress* for each subset. The tree structure reveals that this decision tree first sorts the samples based on their stacking. The samples sorted to the far left have lower *max stress* values compared to those in other nodes. This is because all these sample have a fibre direction of -45 degrees in ply 1, 3 and 5. The symmetry of the stacking sequence implies that there must to be an equal amount of +45 and -45 degree plies, which likely leads to a pure 45 degree stacking with very low *max stress* compared to other stacking sequences.

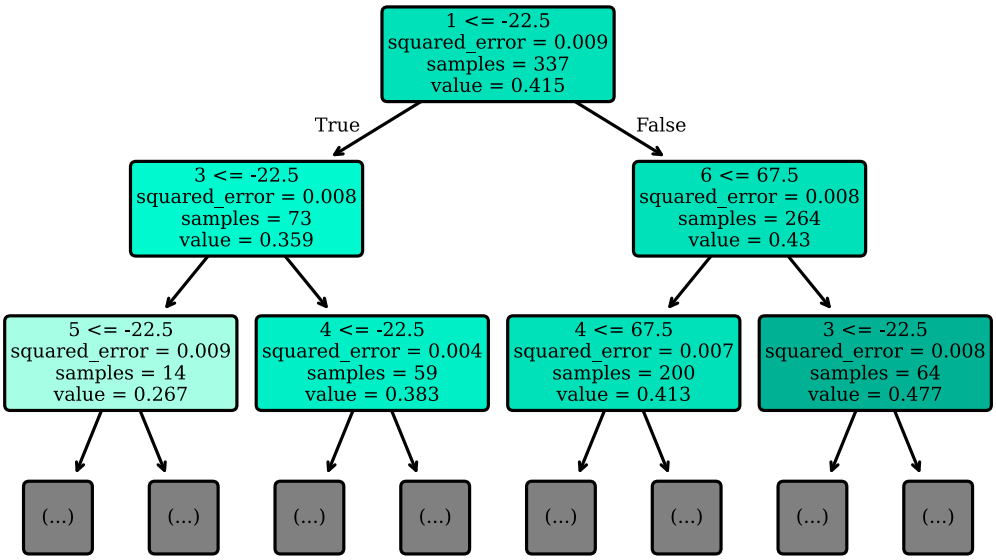


Figure 5.1: First two upper levels of a decision tree for max stress

5.2.2 Doubling of the Dataset

In the current input format, if a sample has one defect, its information is stored in the columns corresponding to *1*. If it has a second defect, its settings are stored in the columns corresponding to *2*. As a result, the columns with *1* are always filled, while the columns with *2* are only filled if a sample has two defects. The sequence of storing defects for samples with two defects is random and lacks any meaning. The idea now is to swap the items of columns *1* and *2*, as shown in Figure 5.2 on the basis of two examples. In the first example, the sample has only one rectangular defect in ply 6. The second sample has a rectangular defect in ply 11 and a triangular defect in ply 1. Before swapping the columns, the first defect would be stored in the columns corresponding to *1*. However, after swapping the columns, its characteristics are now written down in the columns corresponding to *2*. This swapping process has no effect on the properties of the samples. However, it makes a difference for the ML models because they perceive these samples as new observations. To take advantage of this new information, the old

dataset and the new one with the column swap are combined, resulting in twice as large dataset than before. This leads to 1324 observations from which 1059 are used for training the RF and SVR models. Table 5.3 shows the performance for both models on the new dataset. RF needs now 3.82 to train the model, while SVR only needs about 0.28 seconds.

Before:

ply_defect1	...	width1	length1	ply_defect2	...	angle2	width2	length2
6	...	1.7	32.81	0	...	0	0	0
11	...	1.1	1.1	1	...	2.0	0	0

After switching the columns:

ply_defect1	...	angle1	width1	length1	ply_defect2	...	width2	length2
0	...	0	0	0	6	...	1.7	32.81
1	...	2.0	0	0	11	...	1.1	1.1

Figure 5.2: Process of swapping the columns for the two defects

The table reveals that the performance for both RF and SVR has increased significantly across all target variables. The MAPE for *max stress* is now below 10%, while *strain at max stress* and *Young's modulus* are even under 5% for the RF model. The target variable *area before failure* experiences a decrease in MAPE, with 4% improvement for RF and only 1% for SVR. Notably, RF appears to have benefited more from the larger dataset than SVR. The performance for *area after failure* improved significantly, but it still lags behind the other targets in terms of accuracy. However, this improvement indicates that the models can now determine a clearer relationship between the samples and their behaviour after failure with more training data. Given the significant improvements in performance with the larger dataset, it will be retained for future studies.

Table 5.3: Metrics for doubling of the dataset

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0297	7.68	0.0356	9.70
strain at max stress	0.000 496	4.86	0.000 545	5.45
Young's modulus	1.75	4.95	2.25	6.10
area before failure	0.000 238	12.70	0.000 275	15.13
area after failure	0.000 419	50.88	0.000 482	57.62
overall model	0.356	16.21	0.458	18.80

5.2.3 Separate Models for Samples with 12 and 16 Plies

In the following section, two separate models are trained for samples with 12 and 16 plies using both RF and SVR. The assumption behind this approach is that by separating samples based on the number of plies, they will become more similar to each other, which could lead to better predictions. In the input table for the samples with 12 plies, the columns for ply 7 and 8 are removed, as they are not necessary for samples with 12 plies

and would contain only the dummy value -999. All four models are trained using the larger dataset. This results in 718 observations for the 16 ply samples out of which 574 are used for training. The 12 ply samples have 607 observations, resulting in 486 training samples. The performance metrics for both RF and SVR on these separate datasets are presented in Tables 5.4 and 5.5.

Table 5.4: Metrics for samples with 16 plies

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0257	6.66	0.0193	5.09
strain at max stress	0.000 331	3.08	0.000 290	2.73
Young's modulus	2.08	5.95	1.47	4.25
area before failure	0.000 174	8.42	0.000 153	7.44
area after failure	0.000 403	44.24	0.000 424	50.90
overall model	0.421	13.71	0.298	14.08

Table 5.5: Metrics for samples with 12 plies

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0363	12.00	0.0396	12.40
strain at max stress	0.000 766	7.89	0.000 927	9.59
Young's modulus	1.73	5.69	1.92	5.66
area before failure	0.000 295	19.97	0.000 368	25.01
area after failure	0.000 421	58.11	0.000 497	66.01
overall model	0.353	20.73	0.393	23.74

The metrics show that the models trained on the dataset with only 16 ply samples have improved their performance compared to the models from Section 5.2.2. For all target variables except *area after failure*, the MAPE has dropped below 10%. The MAPE for *strain at max stress* is even lower, at around 3%. SVR now shows better performance than RF on all target variables except for *area after failure*. However, the overall MAPE of SVR is slightly worse than that of RF due to its higher MAPE value for *area after failure*. Despite this, the overall MAE of SVR is significantly better than that of RF. This is because the absolute values of *area after failure* have a low scaling and therefore do not have a great impact on the overall MAE.

In contrast, the models trained on samples with 12 plies have poor performance. Particularly, the target *area before failure* worsened in terms of accuracy. One reason for the bad performance on the 12 ply samples might be that fewer observations are available to train the model than with the 16 ply samples. Another reason might be that the samples with 12 plies experience some kind of buckling similar to those with 8 plies, making their properties less predictable. Furthermore, the influence of the defects increases for the 12 plies sample because the material is thinner and more prone to instability. For these samples, the defects represent a large proportion of the total material volume. In

the following, the separate models for 12 and 16 plies are not used because the 12 plies models have worse performance.

5.2.4 Combining the Attributes *angle* and *length*

Currently, depending on the defect, either the column *angle* or the columns *width* and *length* are empty or filled with a 0. The idea is to combine the attributes *angle* and *length* into a single column, *angle_or_length*, for both defect 1 and 2. The column *width* still indicates the shape of the defect because it contains non-zero values only for rectangular defects. By combining the columns *angle* and *length*, a dimensionality reduction from 22 to 20 attributes is achieved, which might lead to better performance for the ML models. Table 5.6 lists the performance for RF and SVR on the new dataset.

Table 5.6: Metrics for *angle* and *length* combined

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0288	7.38	0.0339	9.17
strain at max stress	0.000 479	4.70	0.000 536	5.36
Young's modulus	1.73	4.89	2.04	5.52
area before failure	0.000 238	12.55	0.000 265	14.42
area after failure	0.000 420	51.08	0.000 473	55.87
overall model	0.352	16.12	0.415	18.07

The performances show a slight improvement compared to the current best model from Section 5.2.2. Both RF and SVR have improved slightly across all variables and the overall model, indicating that the dimensionality reduction has led to better performance. Overall, RF has better predictions on the target variables than SVR.

5.2.5 Adding the Attribute *area*

As in the previous section, another attempt is made to transform the attributes to find a dataset format with a better performance than before. Instead of quantifying the size of the defect using columns *angle*, *length* and *width*, a new column *area* is introduced. This columns defines the area of a defect which it occupies in its ply. The idea behind this column is to provide the ML models with a better understanding of the size of the defect, as larger defects may have a higher influence on sample behaviour. By replacing these attributes with *area*, some information about the shape of the defect (rectangular or triangular) is lost. To prevent this loss of information, a new column called *tria_or_rect* is introduced to label the shape of the defect. A rectangular defect has the label 1, a triangular the label 2, and if there is no defect this is indicated by a 0. The introduction of these two columns and the removal of three attributes per defect leads to 20 attributes in the dataset. Table 5.7 displays the metrics for RF and SVR on this altered dataset.

Compared to the current best dataset from Section 5.2.4, both RF and SVR perform slightly worse on target variable *strain at max stress*, with a 0.04% and 0.07% increase in MAPE. Additionally, the MAPE for *area after failure* increased slightly. However, the overall model and all the other target variables experience improved predictions. Therefore, this dataset format is found to be the best so far and will be used for future

investigations of RF and SVR.

Table 5.7: Metrics for adding attribute *area*

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0281	7.24	0.0330	8.84
strain at max stress	0.000 484	4.74	0.000 529	5.29
Young's modulus	1.71	4.83	2.09	5.59
area before failure	0.000 226	11.94	0.000 256	14.02
area after failure	0.000 422	51.14	0.000 475	56.23
overall model	0.347	15.98	0.424	17.99

Compared to the baseline performance from Section 5.2.1, the altered dataset leads to a decrease in MAPE for the overall model from around 22% to 16% for RF and 23% to 18% for SVR. The largest improvement is seen in the target variable *area after failure* with a MAPE decrease from 16% for RF and 17% for SVR. Except for the already well-performing *strain at max stress*, the other variables improved their MAPE for RF of 5% and 3% for SVR.

5.3 Feature Importance for Random Forest

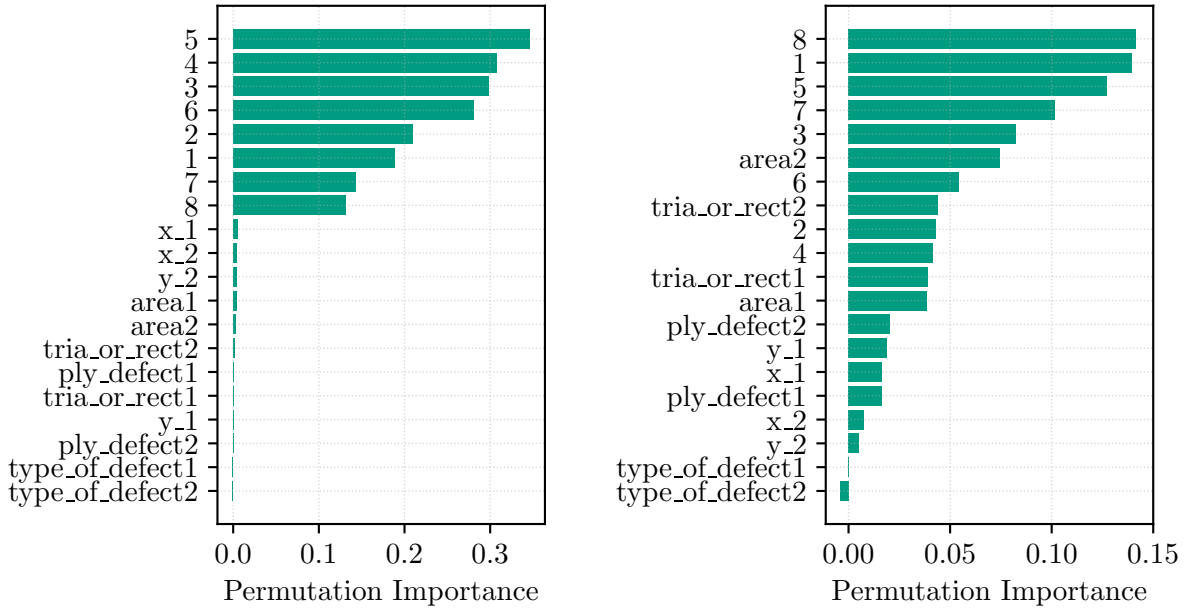
In this section, Feature Importance (FI) is performed to assess the influence and importance of the attributes on the five target variables. This analysis helps in better understanding the ML models and determines what parameters of a defect have the most impact on the behaviour of a sample. The FI is examined using Permutation Feature Importance for the RF model, which is trained on the dataset from Section 5.2.5 using the default settings from scikit-learn library. Figure 5.3 displays the Permutation Importance values for all five target variables.

The Permutation Importance results show that, for the target variables *max stress* and *Young's modulus*, only the fibre direction of the plies has an influence. The defects are largely irrelevant for these two variables. The fibre direction of ply 5 is found to be the most important feature for both targets. The least important plies for *max stress* and *Young's modulus* are ply 7 and 8, which the samples with 12 plies do not have. This finding might suggest that it does not make a big difference if the columns of ply 7 and 8 are filled with fibre directions for the 16 ply samples or dummy values for the 12 ply samples. This could indicate that the samples with 12 and 16 ply have similar values for *max stress* and *Young's modulus*.

For the targets *area before failure* and *area after failure*, the plies also have the most influence. After that come the attributes *area* and the coordinates of a point x and y defining the position of the defect in a ply. However, their permutation importance is still relatively low. The only target variable where the defects seem to have an impact is *strain at max stress*. The first few important features are some plies, but the area of the defects and whether it is a triangular or rectangular have significant influence on the model. Notably, the fibre direction of ply 8 is found to be the most important feature for this variable, sug-

gesting that the samples with 12 and 16 plies have different values for *strain at max stress*.

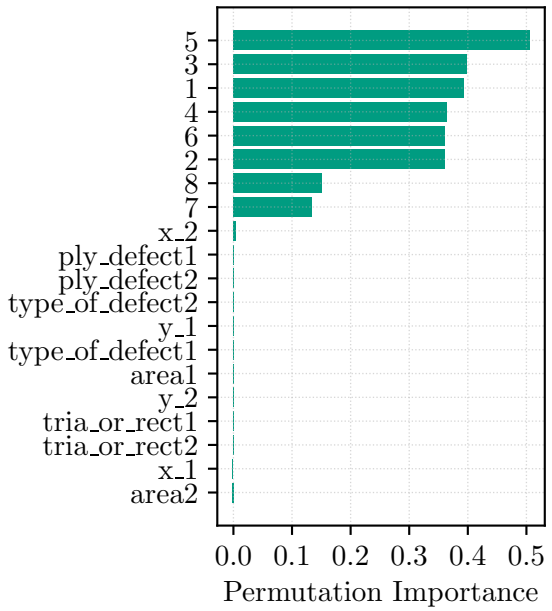
Overall, for all targets, the least important features seems to be *type_ of_ defect*, which defines if the defect is a gap or an overlap. Another surprising finding is that in what ply the defect has low importance overall. When a defect is located in one of the bottom plies, the plies above it also exhibit the defect because they fill it in or cause it to through the layers. However, this does not appear to have significant impact on the target variables.



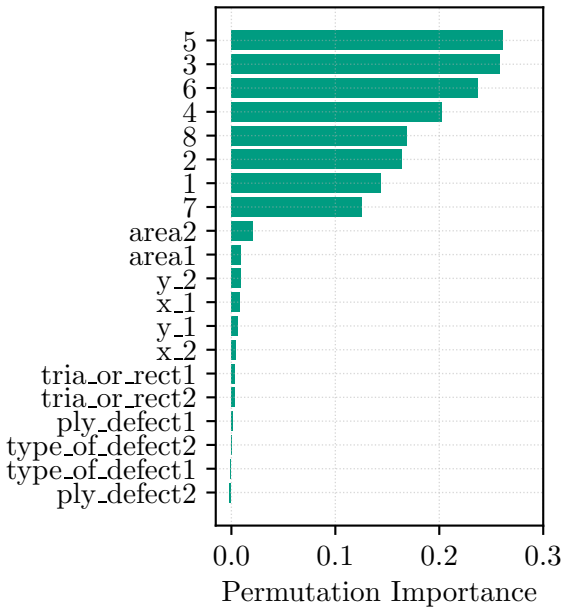
(a) Maximum stress

(b) Strain at maximum stress

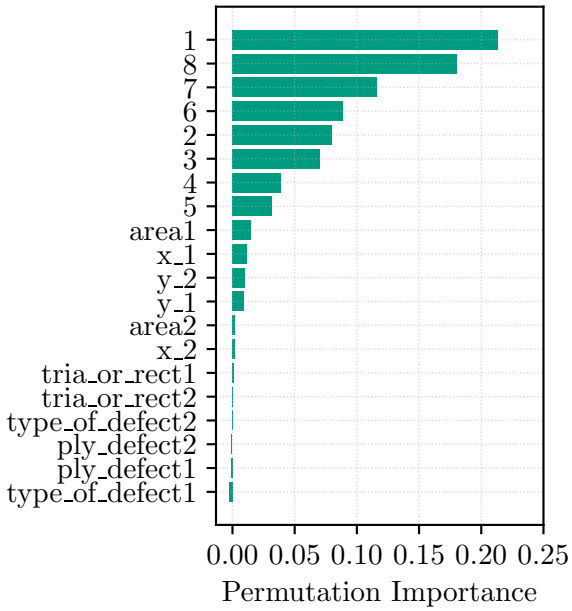
Figure 5.3: FI for different target variables (1)



(c) Young's modulus



(d) Area before failure



(e) Area after failure

Figure 5.3: FI for different target variables (2)

5.4 Hyperparameter Optimization

In this section a hyperparameter optimization for both RF and SVR is performed. The goal is to improve the performance on the dataset of Section 5.2.5. The grid search is executed with the module from scikit-learn [49] and the Bayes search is adopted from scikit-optimize [50]. Both have the k-fold CV integrated and MAE is used for scoring.

5.4.1 Random Forest

As discussed in Section 2.5.2, the number of trees and the number of features considered for each tree are important hyperparameters. Therefore, they are now analysed separately to determine their influence on the model performance.

In the previous Section 5.2, the RF was trained with 100 trees. In the following, several different numbers of trees are tested to evaluate their performance. Figure 5.4 displays the relationship between the number of trees and MAPE, while Figure 5.5 shows the training time for each model. The development of MAPE across various number of trees reveals that the performance does not improve beyond 10 to 15 trees, at which point it converges The training time increases linearly with the number of trees. However, this is not a concern when selecting the optimal number of trees for this application, as even 140 trees can be trained in under two seconds. Every target variable has its own optimal number of trees. The best overall performance achieves the model with 120 trees, as shown in Table 5.8. Compared to the current best model from Section 5.2.5, all target variables increase their performance slightly except for *Young's modulus* for which the MAPE only increases by 0.02% and the MAE by 0.01%. Therefore, RF with 120 trees is established as the current best model.

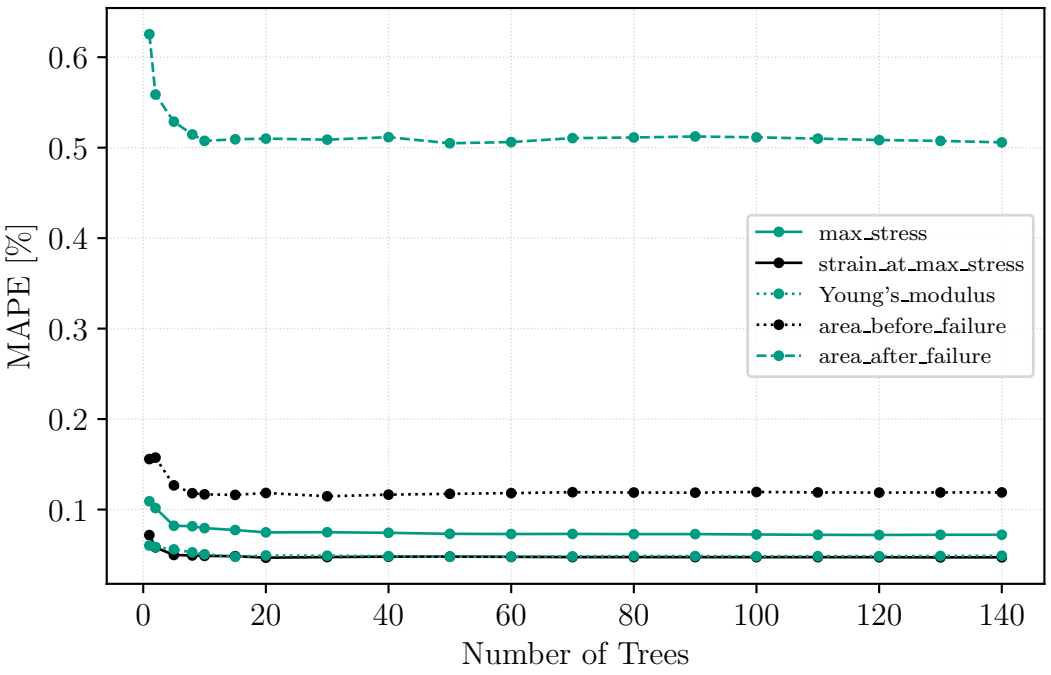


Figure 5.4: MAPE per target variable depending on number of trees

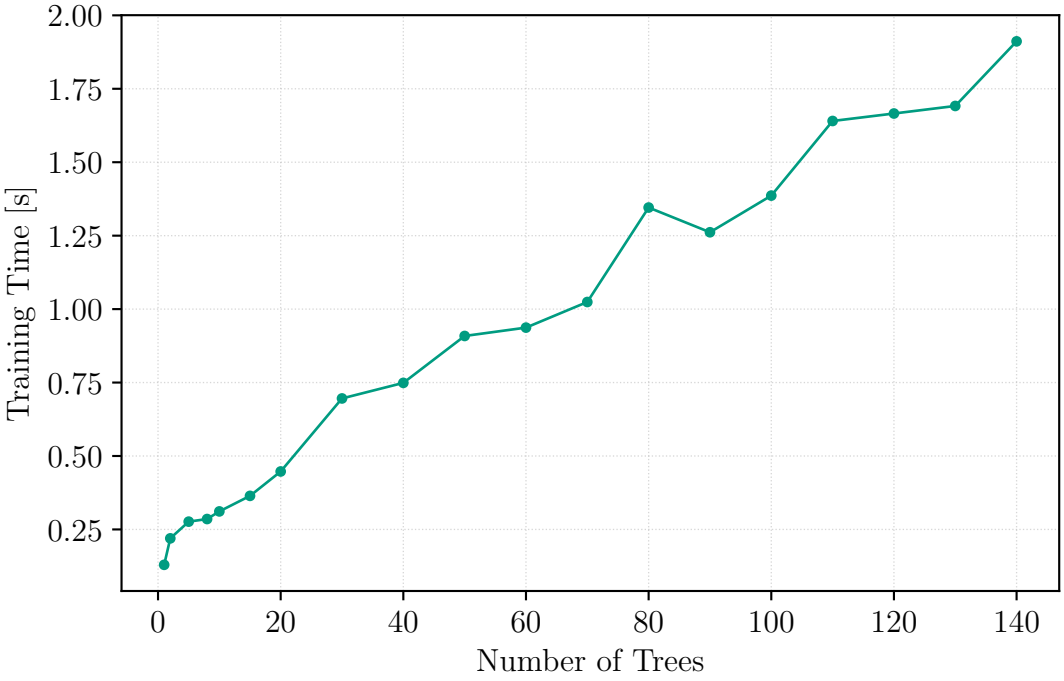


Figure 5.5: Training time for number of trees

Table 5.8: Metrics for RF with 120 desicion trees

Target	MAE	MAPE [%]
max stress	0.0280	7.19
strain at max stress	0.000 483	4.73
Young's modulus	1.72	4.85
area before failure	0.000 224	11.87
area after failure	0.000 422	50.84
overall model	0.349	15.90

In the default setting of scikit-learn, all features are considered when building the trees. Figure 5.6 displays the course of the MAPE for different number of features across all targets. The decimal number represents the fraction of the overall number of features used at a given value (e. g., 1.0 means using all features). The plot shows that the MAPE decreases with an increasing number of features, with the minimum being at 1.0 for all target variables. However, the improvements in MAPE are not as significant as those seen with the number of trees, as the MAPE decrease is more flat. Figure 5.7 displays the influence of the number of features on the training time of RF. Like the plot for the number of trees, it shows an increase in training time with a growing number of features, but the relationship is not monotonic. Given that using all features was already found to be the optimal value, it remains as such.

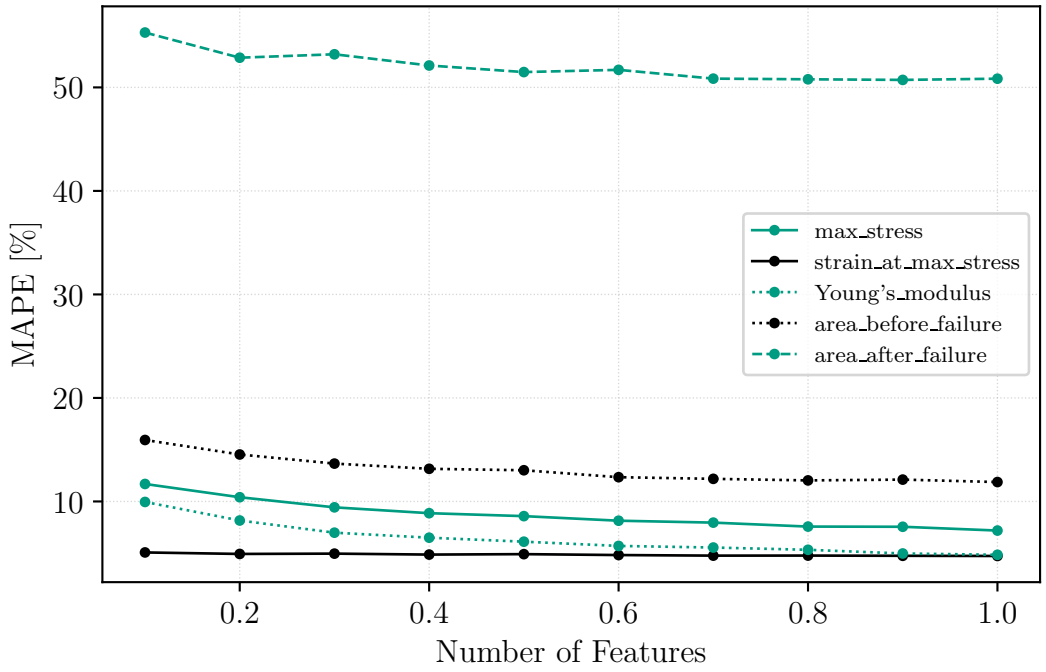


Figure 5.6: MAPE per target variable depending on max_features

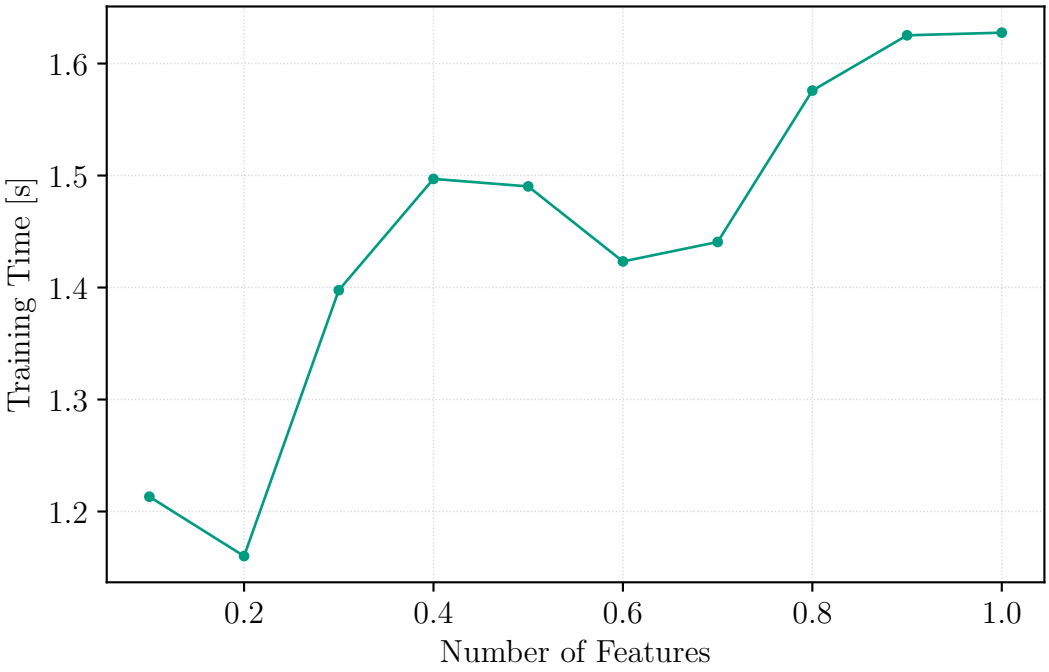


Figure 5.7: Computing time per max_features

With a number of trees of 120 and using all features as the optimal value, three other parameters of RF are optimized using grid search and Bayes Search to determine optimal settings. The parameters optimized are the maximum depth of a decision tree called *max_depth* in scikit-learn, the minimum number of samples required to split a node denoted as *min_samples_split* and the minimum number of samples required to be at a leaf node called *min_samples_leaf* in scikit-learn [47]. The parameter spaces for grid search is

$$\begin{aligned} \text{max\_depth} &: [\mathbf{None}, 10, 20, 30, 40, 50], \\ \text{min\_samples\_split} &: [\mathbf{2}, 5, 7, 10, 12, 15, 20], \\ \text{min\_samples\_leaf} &: [\mathbf{1}, 2, 4, 8] \end{aligned}$$

with the bold numbers indicating the default settings which were used until now. The parameter space for the Bayes search is

$$\begin{aligned} \text{max\_depth} &: \text{Integer}(5, 100), \\ \text{min\_samples\_split} &: \text{Integer}(2, 20), \\ \text{min\_samples\_leaf} &: \text{Integer}(1, 8). \end{aligned}$$

The upper limit *max_depth* is set to 100 to account for the setting None in grid search. Both grid search and Bayes search resulted in the default settings of these three parameters providing the best model performance.

5.4.2 Support Vector Regression

In this section, a hyperparameter optimization for SVR is performed. SVR is generally more sensitive to its hyperparameters than RF. The parameters ϵ, γ and C are optimized using a grid search and a Bayes search. First, a grid search is performed with the default settings and a 5 fold CV and the MAE metric as scoring function. This results in 336 combinations with 1680 fits over the 5 folds. Scikit-learn recommends that for C and γ, the values should be spaced exponentially far apart [35]. The parameter space for ϵ, γ and C is

$$\begin{aligned} \epsilon &: [0.001, 0.01, 0.05, \mathbf{0.1}, 0.15, 0.2] \\ \gamma &: [1\text{e}{-4}, 1\text{e}{-3}, 1\text{e}{-2}, 1\text{e}{-1}, 1, 10, \mathbf{\text{scale}}, \text{auto}] \\ C &: [1\text{e}{-3}, 1\text{e}{-2}, 1\text{e}{-1}, \mathbf{1}, 10, 100, 1000] \end{aligned}$$

with the bold values being the default settings. It takes 468 seconds to perform the grid search. The best found parameters are

$$\epsilon = 0.01, \quad \gamma = 0.01, \quad C = 10.$$

The metrics for this new model are presented in Table 5.9. Compared to the previous model from Section 5.2.5, the MAPE for the overall model and all the target variables decreased by approximately 1%. The target variable that benefited most from the optimization is *Young's modulus*, with an improvement in MAPE from 5.59% to 3.87% and MAE from 2.09 to 1.44. The new values for ϵ, γ and C clearly improved the model performance.

Table 5.9: Metrics for SVR after grid search

Target	MAE	MAPE [%]
max stress	0.0275	7.56
strain at max stress	0.000 532	5.30
Young's modulus	1.44	3.87
area before failure	0.000 239	13.10
area after failure	0.000 474	54.85
overall model	0.294	16.94

In the following, a Bayes search is conducted to compare the results with the grid search and to evaluate the computational times of both optimization methods. The Bayes search from scikit-optimize is used with a 3 fold CV and a parameter space of

$$\begin{aligned} \epsilon &: \text{Real}(0.001, 1.0) \\ \gamma &: \text{Real}(1\text{e}{-4}, 10) \\ C &: \text{Real}(1\text{e}{-3}, 1\text{e}3) \end{aligned}$$

with 'Real' indicating that the search is performed over a real-valued interval. The optimization process takes approximately 55 seconds which is significantly faster than the grid search. The results of the Bayes search are

$$\epsilon = 0.001, \quad \gamma = 0.010, \quad C = 24.88.$$

Notably, both the grid search and Bayes search set γ to 0.01, but the Bayes search determines a higher value for C and a lower value for ϵ than the grid search. Figures 5.8, 5.9 and 5.10 display the combinations assessed by the Bayes search and their MAE for ϵ, γ and C. Figure 5.8 shows that the values for C and γ appear to have a minimum point in the parameter space, which is identified by the Bayes search as the region with the lowest MAE value. However, Figures 5.9 and 5.10 indicate that ϵ does not have a clear minimum. The values for ϵ all have relatively high MAE values across the range of ϵ, indicating a lack of a densely regions with low MAE values. The optimized value for ϵ is the lower limit of the given range. As a result, the lower limit of the parameter space of ϵ is set to 1e$-$10. A second Bayes search is performed with this new range for ϵ, but the Bayes search still chooses this very low limit for ϵ with the values for γ and C staying the same. Furthermore, the model performance does not change. As a result, the value $\epsilon = 0.001$ is kept for the Bayes search.

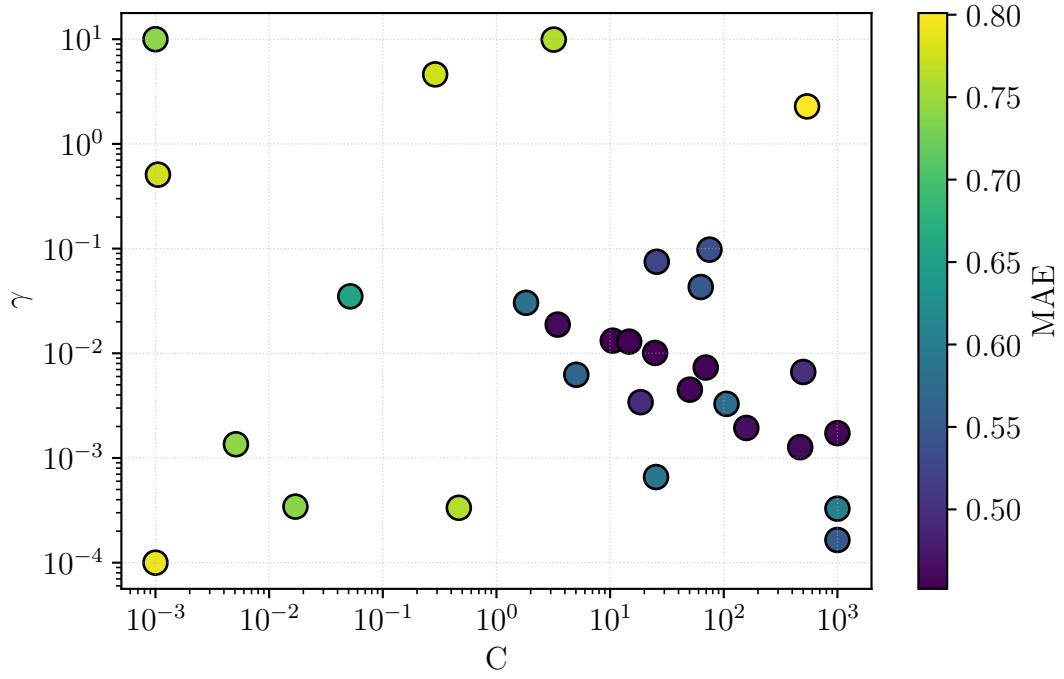


Figure 5.8: Distribution of C and γ with MAE

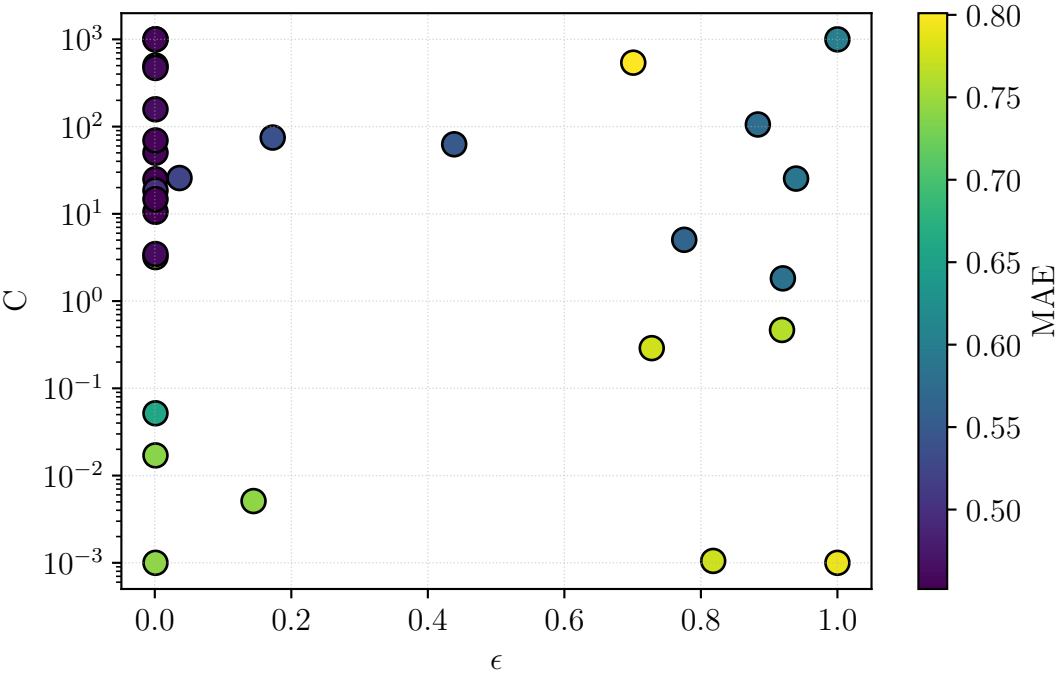


Figure 5.9: Distribution of ϵ and C with MAE

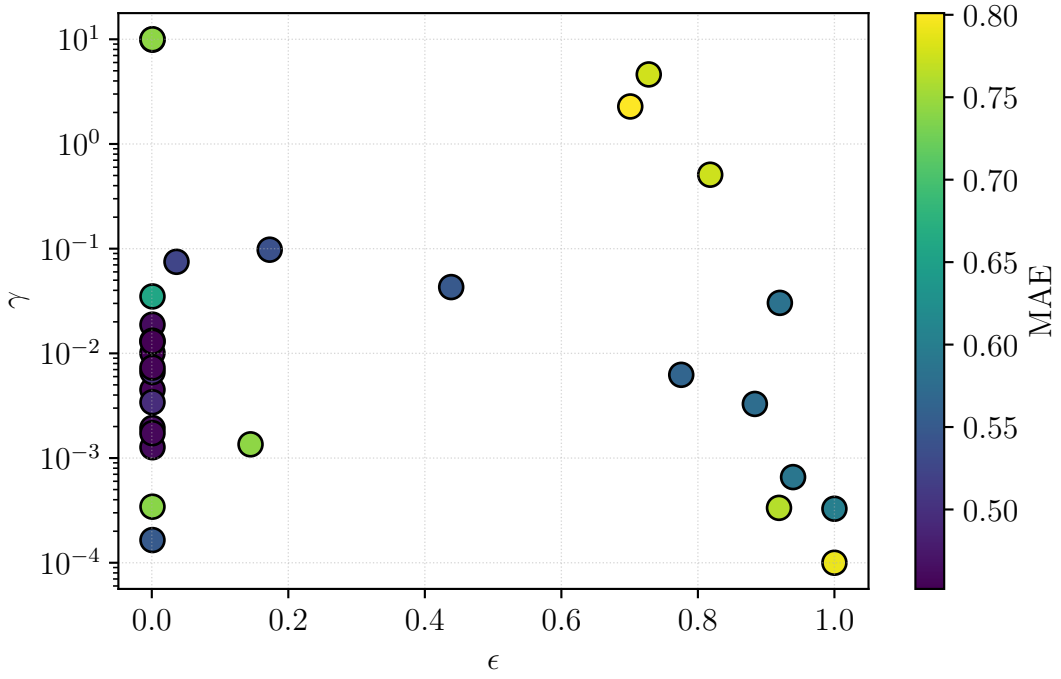


Figure 5.10: Distribution of ϵ and γ with MAE

Table 5.10 presents the metrics for SVR using the optimized parameters from the Bayes search. The overall model performance is equivalent to the grid search model according to the MAPE but better in terms of the MAE. Furthermore, the target variables *max stress*, *Young's modulus* and *area before failure* show improved prediction performance compared to the grid search model. Only for *strain at max stress* and *area after failure* does the Bayes search model make worse predictions than the grid search model. Despite this, the Bayes search model has a better overall MAE and performs better on three out of five target variables, making it the best SVR model according to these metrics.

Table 5.10: Metrics for SVR after Bayes search

Target	MAE	MAPE [%]
max stress	0.0258	6.89
strain at max stress	0.000 553	5.47
Young's modulus	1.18	3.11
area before failure	0.000 244	12.85
area after failure	0.000 489	56.40
overall model	0.242	16.94

5.5 Comparison of the final Random Forest and Support Vector Regression models

In this section, the final models of RF and SVR are compared to the baseline performance from Section 5.2.1. The final RF model has 120 trees and requires approximately 4.8 seconds to train on the dataset from Section 5.2.5. The settings for the final SVR are $\epsilon = 0.001$, $\gamma = 0.010$ and $C = 24.88$, and it takes about 1.5 seconds to train. Table 5.11 displays the baseline performance, while Table 5.12 presents the performance of the final models.

Table 5.11: Metrics of the baseline performance

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0458	12.26	0.0432	11.69
strain at max stress	0.000 548	5.28	0.000 578	5.69
Young's modulus	3.30	9.44	2.87	8.20
area before failure	0.000 317	16.19	0.000 316	16.42
area after failure	0.000 507	67.33	0.000 563	73.21
overall model	0.669	22.10	0.583	23.04

The target variable that showed the least improvement is *strain at max stress*, but it already had good performance in the baseline models. However, the target variables *max stress*, *Young's modulus* and *area before failure* all saw significant improvements in their MAPE values, with a decrease of about 5%. The weakest target variable, *area after failure*, improved its MAPE by 17%. Overall, both RF and SVR seen an improvement in performance of around 7%, thanks to the alterations made to the dataset and the optimization of their hyperparameters.

Table 5.12: Metrics for the final models

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0280	7.19	0.0258	6.89
strain at max stress	0.000 483	4.73	0.000 553	5.47
Young's modulus	1.72	4.85	1.18	3.11
area before failure	0.000 224	11.87	0.000 244	12.85
area after failure	0.000 422	50.84	0.000 489	56.40
overall model	0.349	15.90	0.242	16.94

In the final model, RF has a better overall MAPE, but SVR performs better in terms of MAE. This is because RF excels at predicting *area after failure*, while SVR performs well on *Young's modulus*. As mentioned earlier, these variables have high MAPE and MAE values, respectively. When examining the metrics for both models across all target variables, it becomes clear that they are generally comparable in performance. However, RF performs better on *strain at max stress*, *area before failure* and *area after failure*. Except for *strain at max stress*, these variables have poor performance, which makes it challenging to determine which model is more suitable for predicting mechanical properties. are these the variables with a comparatively bad performance. RF appears to have a slight advantage over SVR, but both models have their strengths and weaknesses.

6 Application of the Machine Learning Models

In this section, the final trained ML models from Sections 5.4.1 and 5.4.2 are evaluated on various datasets to assess their performance in producing realistic results for different defect configurations.

6.1 Performance on Unknown Datasets

The final trained models for RF and SVR are evaluated on two completely unknown datasets, meaning the training set does not contain any similar samples. First, the models are tested on 10 samples with only 8 plies. These samples are chosen from the initial dataset of 1000 samples. Care is taken to choose samples which failed with fibre rupture and did not have numerical problems with large oscillations during simulation. Table 6.1 shows the performance of these models on these 10 samples with 8 plies. All the target variables have poor performance with most targets having a MAPE value of over 20%. This shows that the samples with 8 plies exhibit significantly different behaviour compared to thicker samples, validating the decision of removing them from the training dataset in Section 4.3.1.

Table 6.1: Metrics for samples with 8 plies

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.0571	18.00	0.141	39.25
strain at max stress	0.001 91	24.70	0.001 76	22.57
Young's modulus	7.80	18.68	22.9	56.82
area before failure	0.000 606	51.88	0.000 647	41.62
area after failure	0.001 10	379.50	0.000 408	140.70
whole model	1.57	98.56	4.62	60.19

Table 6.2 presents the prediction performance on 10 randomly chosen samples with 12 and 16 plies, symmetric stacking and no defects. For RF, all target variables showed a decline in performance, except for *strain at max stress* and Young's modulus, which still performed well, with MAPE values under 6%. However, the performance of the target *area after failure* is significantly worse, with a MAPE of around 132%, making this prediction unreliable. The other two targets showed fine performance, but their MAPE increased by around 5% compared to previous results. In contrast, SVR performed similarly on samples without defects, except for *area after failure*, which showed a worsening MAPE of under 1%. This finding shows that SVR can predict the mechanical properties of sample iwthout defects with good accuracy.

Table 6.2: Metrics for samples without defects

Target	RF		SVR	
	MAE	MAPE [%]	MAE	MAPE [%]
max stress	0.060 657	11.94	0.033 527	6.73
strain at max stress	0.000 746	5.91	0.000 727	5.78
Young's modulus	2.113 720	5.72	1.261 055	3.33
area before failure	0.000 525	16.00	0.000 422	13.05
area after failure	0.000 960	132.69	0.000 773	67.76
whole model	0.435 300	34.45	0.259 300	19.33

6.2 Effects of Defects on the Samples

The ML models are now used to identify the influence of different defects on a sample's behaviour. The target variables are predicted using both the final RF and SVR models. For the samples without any defects, the SVR model is used as it has shown excellent performance on these datasets in the previous Section 6.1. For samples with a defect, the targets *max stress* and *Young's modulus* are estimated with the SVR and *strain at max stress*, *area before failure*, and *area after failure* are predicted with RF because the models showed the best performances on these targets respectively.

6.2.1 Defects in Different Plies

First, it is examined whether the ply in which the defect occurs has a significant influence. If a defect is in one of bottom plies, more plies are stacked above it and are also affected, as they must fill in the gap or stack on the overlap. To investigate this effect, different samples with a stacking sequence of $[90°/0°/90°/0°]_{2s}$ are created, each containing a rectangular gap in the upper half of one of the 90 degree plies, with a with of 1.9 mm. The defect is placed in ply 1,7, or 16, and an additional sample without a defect is included as reference. Table 6.3 shows the predicted values for the target variables.

Table 6.3: Mechanical properties for a defect travelling through the plies

Defected Ply	max stress	strain at max stress	Young's modulus	area before failure	area after failure
No Defect	0.6099	0.011 62	49.88	0.003 513	0.002 040
1	0.5849	0.011 42	50.06	0.003 303	0.001 089
7	0.5959	0.011 59	50.29	0.003 309	0.001 098
16	0.6062	0.011 65	50.30	0.003 394	0.001 252

It can be observed that the further down the defect is located, the weaker the sample becomes. However, the differences in strength values are relatively small. The maximum stress decreases by only 3.5% when comparing a defect in ply 16 to one in ply 1. The target *strain at max stress* also decreases when the defect is at the bottom of the sample, but again, the differences are minimal. Furthermore, there is no noticeable difference between a defect-free sample and one with a defect in ply 16. These results are consistent with the FI analysis from Section 5.3, where the defected ply was identified as a less

significant feature. While the defected ply does have some influence, its effect is minor compared to other, more important features.

6.2.2 Size of the Defects

In the following, the influence of a rectangular gap on the behaviour of a sample is examined. For this, different samples with defect sizes between 1.1 mm and 1.9 mm are tested, which corresponds to the defect range used for training the ML models The stacking sequence is $[0°/-45°/90°/+45°]_{2s}$, with the defect located in ply 7, which is a 90 degrees ply. Table 6.4 presents the mechanical properties of the samples with varying gap sizes.

Table 6.4: Mechanical properties for different gap sizes

Size of Gap	max stress	strain at max stress	Young's modulus	area before failure	area after failure
No Defect	0.4548	0.012 06	39.83	0.002 881	0.001 656
1.1 mm	0.4354	0.011 69	38.71	0.002 621	0.002 000
1.5 mm	0.4355	0.011 63	38.72	0.002 613	0.001 998
1.9 mm	0.4355	0.011 62	38.74	0.002 620	0.002 005

The results show no significant differences between the gap sizes. This is due to mesh size of 1.1mm $\times$ 1.1mm. Consequently, rectangular defects with sizes of 1.1 mm and 1.9 mm are discretised into the same element width, leading to identical results. This indicates that the model is not suitable for comparing different rectangular defect sizes. Therefore, a finer mesh an an adjusted defect size range should be considered in future work. Overall, the samples containing a gap show a lower maximum strength and strain at maximum strength compared to the sample without a defect.

6.2.3 Differences between Gap and Overlap Defects

According to the FI analysis, the attribute indicating whether the defect is a gap or an overlap is overall the least important feature for all target variables. This, however, contrasts with findings from previous studies, where overlaps were shown to have little influence on strength and stiffness, while gaps did have a significant impact (see Section 2.1) [4], [5], [8]. To investigate this further, samples with a stacking sequence of $[0°/-45°/90°/+45°]_{2s}$ and a defect in ply 7, which is a 90 degrees ply, are analysed. Table 6.5 compares the results for a sample with a gap, an overlap, and no defect.

Table 6.5: Mechanical properties for different gap sizes

	max stress	strain at max stress	Young's modulus	area before failure	area after failure
No Defect	0.4548	0.012 06	39.83	0.002 881	0.001 656
Gap	0.4355	0.011 62	38.74	0.002 620	0.002 005
Overlap	0.4330	0.011 62	38.54	0.002 622	0.002 024

The results show that *max stress* and *Young's modulus* vary only minimally between gap and overlap defects, while the values for *strain at max stress* are even identical. In line

with the FI analysis, this suggests that, according to the ML models, there is no significant difference in behaviour between samples with rectangular gaps or overlaps.

7 Conclusion

In this work, the goal was to develop a RF and a SVR model to predict the mechanical properties of tape-layered specimen with defects which are manufactured with the AFP process. The ML models were trained on results from simulation data with LS-DYNA representing a compression test. The used simulation model employed an implicit time integration method that switched to explicit when the implicit method failed to reach convergence after sample failure. In contrast to the explicit method, the implicit part of the simulations showed no oscillations, providing good quality results. However, an auto-switch formulation was used as a good compromise between quality of the results and robustness of the model. With an optimization of the parameters of the material model MAT_54, the simulation results were matched close to experimental results. The computational time was reduced by simulating only the part of the sample not affected by clamping.

Using this model, 1000 samples with different properties and defect configurations were generated, out of which only 17 terminated in error, demonstrating the robustness of the model. The samples with 8 plies experienced failure through buckling, leading to numerical problems during the simulation with high oscillations. Additionally, this failure mode is very different from the fibre ruptured observed in 12- and 16-ply samples. As a result, all specimens with 8 plies were removed from the dataset to ensure mostly uniform failure modes across the dataset.

After training RF and SVR on the dataset using the default settings from scikit-learn to obtain a baseline performance, the performances of the models were improved through feature engineering techniques involving the generation of a new attribute and the doubling of the dataset. Further optimization was performed on RF and SVR with hyperparameter tuning, which improved their performances. The final models for RF and SVR achieved excellent prediction performance for *max stress*, *strain at max stress* and *Young's modulus*, with MAPE under 7% and some values reaching under 5%. The target *area before failure* had a good performance with a MAPE of around 12%. However, the target variable *area after failure* had very poor prediction results with a MAPE between 50% and 56%, which may be attributed to its extraction from the explicit part of the simulation. Overall, SVR excelled on *max stress* and *Young's modulus* while RF had a better performance on the other three targets. In conclusion, the mechanical properties for a compression test were successfully predicted using both RF and SVR.

The application of the models revealed that SVR can predict the mechanical properties of samples without defects with the same precision without being trained on them. The properties for samples with defects at different plies were predicted with realistic results showing that defects at the bottom layers decrease the strength of a sample compared to defects in the upper layers. However, a comparison between gap and overlap defects showed no difference in the mechanical properties.

8 Outlook

The aim of this thesis was to establish the foundation capable of predicting all the mechanical properties of components with defects during the AFP manufacturing process. To achieve this, several improvements to both the simulation and the machine learning models are required.

The simulation model could be refined by using a smaller element size to represent smaller defects and to obtain more accurate simulation results. Furthermore, the material model could be more closely aligned with experimental data, particularly to better represent the behaviour after failure. For this, an extensive material test campaign should be performed to provide adequate calibration data for the simulation model. Another important step is to implement samples with a larger variation to cover a wider range of defect types. This includes samples with a higher number of defects, additional plies, or fibre orientations other than those used in this work. Defects themselves could also be modelled more realistically, for example by incorporating the reduced fibre volume content in gap defect. Additionally, to obtain all mechanical properties of a component, bending and tension tests should be evaluated alongside compression tests. For thinner samples with eight plies, the ML should be able to handle different failure modes, such as buckling, and predict the corresponding mechanical properties. Moreover, it would be useful if the ML model can output the whole stress strain curve rather than just individual values from it.

Another topic is the integration of additional parameters from the AFP process, such as the temperature and layup speed. Additionally, it would be advantageous to incorporate data from the robot arm used for tape placement. During production, the robot generates a point cloud of the laid tapes from which the defect geometry could be estimated. Converting this point cloud into a suitable format would enable real-time predictions without manually inserting the sample properties.

References

[1] L. Raps, F. Atzler, A. R. Chadwick, and H. Voggenreiter, "In-situ automated fiber placement gap defects filled by fused granular fabrication," *Manufacturing Letters*, vol. 40, pp. 125–128, 2024. DOI: 10.1016/j.mfglet.2024.03.007.

[2] F. Kibrete, T. Trzepieciński, H. S. Gebremedhen, and D. E. Woldemichael, "Artificial Intelligence in Predicting Mechanical Properties of Composite Materials," *Journal of Composites Science*, vol. 7, no. 9, p. 364, 2023. DOI: 10.3390/jcs7090364.

[3] G. A. Lyngdoh, N.-K. Kelter, S. Doner, N. A. Krishnan, and S. Das, "Elucidating the auxetic behavior of cementitious cellular composites using finite element analysis and interpretable machine learning," *Materials & Design*, vol. 213, 2022. DOI: 10.1016/j.matdes.2021.110341.

[4] S. Yoo et al., "Effect of automated fibre placement (AFP) process induced defects: Dynamic compression properties at intermediate strain rates," *Journal of Composite Materials*, 2025. DOI: 10.1177/00219983251357662.

[5] F. Diemar et al., "X-ray micro-computed tomography for mechanical behaviour analysis of automated fiber placement (afp) laminates with integrated gaps and overlaps," *Composite Structures*, vol. 351, 2025, ISSN: 02638223. DOI: 10.1016/j.compstruct.2024.118601.

[6] M. Vinot, L. Raps, and N. Toso, "High-Fidelity Modelling of Composite Specimens Manufactured with the Automated Fibre Placement Technique," 2024. [Online]. Available: https://www.ansys.com/content/dam/events/event-pdfs/german-innovation-conference-2024/02-dlr-vinot.pdf.

[7] C. Sacco, A. B. Radwan, A. Anderson, R. Harik, and E. Gregory, "Machine learning in composites manufacturing: A case study of Automated Fiber Placement inspection," *Composite Structures 250*, vol. 250, 2020. DOI: 10.1016/j.compstruct.2020.112514.

[8] L. Raps, I. Schiel, and A. R. Chadwick, "Effect of gap defects on in-situ AFP-manufactured structures," *Composites Meet Sustainability - Proceedings of the 20th European Conference on Composite Materials*, 2022. DOI: 10.5075/epfl-298799_978-2-9701614-0-0. [Online]. Available: https://infoscience.epfl.ch/entities/publication/ceba6a39-6b21-4fb1-9e9c-9c51c63b9a97.

[9] P. Wriggers, *Nonlinear Finite Element Methods*. Berlin Heidelberg: Springer-Verlag, 2008, ISBN: 978-3540710004.

[10] J. E. Marsden and T. J. R. Hughes, *Mathematical Foundations of Elasticity*. New York: Dover Publications, 1994, ISBN: 978-0486678658.

[11] P. G. Ciarlet, *Mathematical Elasticity: Volume I: Three-Dimensional Elasticity*. Amsterdam: Elsevier Science Publishers B. V., 1988, ISBN: 978-0-444-70259-3.

[12] ANSYS, Ed., *LS-DYNA Theory Manual*, 2024. [Online]. Available: https://ftp.lstc.com/anonymous/outgoing/web/ls-dyna_manuals/R15/LS-DYNA_Manual_Theory_R15.pdf.

[13] H. Schürmann, *Konstruieren mit Faser-Kunststoff-Verbunden*, 2nd ed. Berlin and Heidelberg: Springer, 2007, ISBN: 978-3-540-72189-5.

[14] T. Mánik, "A natural vector/matrix notation applied in an efficient and robust return-mapping algorithm for advanced yield functions," *European Journal of Mechanics - A/Solids*, vol. 90, 2021, ISSN: 09977538. DOI: 10.1016/j.euromechsol.2021.104357.

[15] ANSYS, Ed., *LS-DYNA Keyword User's Manual: Volume I*, 2023. [Online]. Available: https://ftp.lstc.com/anonymous/outgoing/web/ls-dyna_manuals/R14/LS-DYNA_Manual_Volume_I_R14.pdf.

[16] ANSYS, Ed., *Guideline for Implicit Analyses Using Ansys LS-DYNA Software*, 2024. [Online]. Available: https://lsdyna.ansys.com/wp-content/uploads/2024/09/Ansys_LS-DYNA_implicit_guideline_2024R2.pdf.

[17] ANSYS, Ed., *LS-DYNA Keyword User's Manual: Volume II Material Models*, 2023. [Online]. Available: https://ftp.lstc.com/anonymous/outgoing/web/ls-dyna_manuals/R14/LS-DYNA_Manual_Volume_II_R14.pdf.

[18] P. Feraboli and B. Wade, "Simulating Laminated Composite Materials Using LS-DYNA Material Model MAT54: Single-Element Investigation," *Federal Aviation Administration Technical Report*, 2015. [Online]. Available: https://www.researchgate.net/publication/322152587_Simulating_Laminated_Composite_Materials_Using_LS-DYNA_Material_Model_MAT54_Single-Element_Investigation.

[19] A. Cherniaev, J. Montesano, and C. Butcher, "Modelling the Axial Crush Response of CFRP Tubes using MAT054, MAT058 and MAT262 in LS-DYNA," 2018. [Online]. Available: https://www.dynalook.com/conferences/15th-international-ls-dyna-conference/composites/modeling-the-axial-crush-response-of-cfrp-tubes-using-mat054-mat058-and-mat262-in-ls-dyna-r.

[20] S. Dridi, "Supervised Learning - A Systematic Literature Review," 2021. DOI: 10.31219/osf.io/qtmcs.

[21] G. Cerulli, *Fundamentals of Supervised Machine Learning*. Cham: Springer International Publishing, 2023, ISBN: 978-3-031-41336-0.

[22] M. A. El Mrabet, K. El Makkaoui, and A. Faize, "Supervised Machine Learning: A Survey," in *2021 4th International Conference on Advanced Communication Technologies and Networking (CommNet)*, IEEE, 2021, pp. 1–10, ISBN: 978-1-6654-0306-1. DOI: 10.1109/CommNet52204.2021.9641998.

[23] scikit-learn developers, Ed., *3.4. Metrics and scoring: quantifying the quality of predictions*. Accessed: Jul. 25, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html.

[24] S. Ozdemir and D. Susarla, *Feature Engineering Made Easy: Identify unique features from your dataset in order to build powerful machine learning systems*, 1st ed. Birmingham: PACKT Publishing, 2018, ISBN: 9781787286474.

[25] scikit-learn developers, Ed., *5.2. Permutation feature importance*. Accessed: Jul. 25, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/permutation_importance.html.

[26] scikit-learn developers, Ed., *2.7. Novelty and Outlier Detection*. Accessed: Jul. 25, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/outlier_detection.html.

[27] scikit-learn developers, Ed., *LocalOutlierFactor*. Accessed: Jul. 21, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html.

[28] L. Breimann, "Random Forests," *Machine Learning*, no. 45, pp. 5–32, 2001. DOI: 10.1023/A:1010933404324.

[29] scikit-learn developers, Ed., *1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking*. Accessed: Jul. 28, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/ensemble.html.

[30] H. Wang, C. Zhang, B. Zhou, S. Xue, P. Jia, and X. Zhu, "Prediction of triaxial mechanical properties of rocks based on mesoscopic finite element numerical simulation and multi-objective machine learning," *Journal of King Saud University - Science*, vol. 35, no. 7, p. 102 846, 2023, ISSN: 10183647. DOI: 10.1016/j.jksus.2023.102846.

[31] M. Alrsai, A. Alsahalen, H. Karampour, M. Alhawamdeh, and O. Alajarmeh, "Integrated finite element analysis and machine learning approach for propagation pressure prediction in hybrid Steel-CFRP subsea pipelines," *Ocean Engineering*, vol. 311, p. 118 808, 2024, ISSN: 00298018. DOI: 10.1016/j.oceaneng.2024.118808.

[32] scikit-learn developers, Ed., *1.10. Decision Trees*. Accessed: Jul. 28, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html.

[33] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, pp. 273–297, 1995. DOI: 10.1007/BF00994018.

[34] A. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004. DOI: 10.1023/B:STCO.0000035301.49549.88.

[35] scikit-learn developers, Ed., *1.4. Support Vector Machines*. Accessed: Jul. 29, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html#id15.

[36] scikit-learn developers, Ed., *RBF SVM parameters*. Accessed: Jul. 29, 2025. [Online]. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html.

[37] scikit-learn developers, Ed., *3.2. Tuning the hyper-parameters of an estimator*. Accessed: Aug. 4, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/grid_search.html#grid-search.

[38] scikit-learn developers, Ed., *3.1. Cross-validation: evaluating estimator performance*. Accessed: Aug. 5, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation.

[39] S. Hanifi, A. Cammarono, and H. Zare-Behtash, "Advanced hyperparameter optimization of deep learning models for wind power prediction," *Renewable Energy*, vol. 221, 2024. DOI: 10.1016/j.renene.2023.119700.

[40] W. Wang, H. Wang, J. Zhou, H. Fan, and X. Liu, "Machine learning prediction of mechanical properties of braided-textile reinforced tubular structures," *Materials & Design*, vol. 212, 2021. DOI: 10.1016/j.matdes.2021.110181.

[41] C. Zhang, Y. Li, B. Jiang, R. Wang, Y. Liu, and L. Jia, "Mechanical properties prediction of composite laminate with FEA and machine learning coupled method," *Composite Structures*, vol. 299, 2022. DOI: 10.1016/j.compstruct.2022.116086.

[42] C. Schmied, T. Erhart, T. Borrvall, N. Karajan, and M. Schenke, *Implicit Analysis using LS-DYNA: Tips & Tricks for successful implicit analyses*, 2020. Accessed: Jul. 9, 2025. [Online]. Available: https://www.dynamore.de/de/download/presentation/2020/implicit-2020.pdf.

[43] DYNAmore GmbH, Ed., *Welcome to LS-OPT Support Site*. Accessed: Aug. 7, 2025. [Online]. Available: https://www.lsoptsupport.com/.

[44] R. A. Cutting, A. J. Favaloro, and J. E. Goodsell, "A novel post-processing method for progressive failure analysis of brittle composite compression," *Journal of Composite Materials*, vol. 56, no. 19, pp. 3029–3047, 2022, ISSN: 0021-9983. DOI: 10.1177/00219983221101985.

[45] SciPy community, Ed., *integral: scipy.interpolate.UnivariateSpline.* 27.07.2025. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.UnivariateSpline.integral.html.

[46] scikit-learn developers, Ed., *IsolationForest*. Accessed: Jul. 21, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html#sklearn.ensemble.IsolationForest.

[47] scikit-learn developers, Ed., *RandomForestRegressor*. Accessed: Aug. 5, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html#sklearn.ensemble.RandomForestRegressor.

[48] scikit-learn developers, Ed., *SVR*, 25.08.2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html.

[49] scikit-learn developers, Ed., *Gridsearchcv*, 26.08.2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[50] scikit-optimize contributors, Ed., *Skopt.bayessearchcv*, 12.10.2021. [Online]. Available: https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html.

A Appendix: Control Cards Implicit

```
*CONTROL_IMPLICIT_AUTO
$#   iauto     iteopt     itewin      dtmin      dtmax      dtexp      kfail
         1         11          5                 0.005        2.0          0
$#   kcycle
         0
$
*CONTROL_IMPLICIT_GENERAL
$#  imflag         dt0     imform       nsbs        igs      cnstn       form
         4     0.0003          2          1          2          0          0
$#  zero_v
         0
$
*CONTROL_IMPLICIT_DYNAMICS
$#   imass      gamma       beta     tdybir     tdydth     tdybur      irate
         1        0.6       0.38        0.0 1.00000E281.00000E28          0
$#   alpha
       0.0
$
*CONTROL_ACCURACY
$#     osu        inn     pidosu       iacc
         1          2          0          1
$
*CONTROL_IMPLICIT_SOLUTION
$#  nsolvr     ilimit     maxref      dctol      ectol      rctol      lstol
        12         11         15      0.001       0.011.00000E10        0.9
$#  abstol
1.0000E-10
$#   dnorm     diverg      istif    nlprint     nlnorm    d3itctl      cpchk
         1          1          1          1          2          0          0
$#  arcctl     arcdir     arclen     arcmth     arcdmp     arcpsi     arcalf
         0          0        0.0          1          2          0          0
$#  arctim
         0
$#   lsmtd      lsdir       irad       srad       awgt       sred
         4          2        0.0        0.0        0.0        0.0
$
$---------------------------------------------------------------------
$
*CONTACT_AUTOMATIC_SINGLE_SURFACE_MORTAR_ID
$#     cid
         1
$#    ssid       msid      sstyp      mstyp     sboxid     mboxid        spr
         1          0          2          0          0          0          0
$#     mpr
         0
$#      fs         fd         dc         vc        vdc     penchk         bt
      0.25        0.0        0.0        0.0        0.0          0        0.0
$#      dt
1.00000E20
$#     sfs        sfm        sst        mst       sfst       sfmt       fsf
       1.0        1.0        0.0        0.0        1.0        1.0        1.0
$#     vsf
       1.0
$#    soft     sofscl     lcidab     maxpar      sbopt      depth      bsort
         0        0.1          0       1.25        3.0          5          0
$#  frcfrq
         1
```

```
$#   penmax     thkopt     shlthk      snlog       isym      i2d3d     sldthk
         0.0          0          1          1          0          0        0.0
$#   sldstf
         0.0
$#     igap     ignore     dprfac     dtstif    unused     unused     flangl
           1          1        0.0        0.0                             0.0
$# cid_rcf
           0
```

B Appendix: Material Cards

Table B.1: Parameters for MAT_54 ([17], [18], [19])

Parameter	Definition	Type	Unit	Used Value
MID	Material identification number	Computational	—	1
RO	Mass density	Constitutive properties	kg/mm^3	$1.95 \cdot 10^{-6}$
EA	Longitudinal Young's modulus E_1	Constitutive properties	GPa	100.867
EB	Transverse Young's modulus E_2	Constitutive properties	GPa	11.8
EC	Through-thickness Young's modulus	Constitutive properties	GPa	11.8
PRBA	Minor Poisson's ratio ν_{21}	Constitutive properties	—	0.022
PRCA	Minor Poisson's ratio ν_{31}	Constitutive properties	—	0.022
PRCB	Major Poisson's ratio ν_{12}	Constitutive properties	—	0.606
GAB	Shear modulus G_{12}	Constitutive properties	GPa	4.5
GBC	Shear modulus G_{23}	Constitutive properties	GPa	3.7
GCA	Shear modulus G_{31}	Constitutive properties	GPa	4.5
ALPH	Elastic shear stress nonlinear factor α	Shear weighing factor	—	0.1
SOFT	Material strength factor after crushing failure	Damage factor	—	1.0
FBRT	Longitudinal tensile strength factor after 2-direction failure	Damage factor	—	0.5
YCFAC	Longitudinal compressive strength factor after 2-direction failure	Damage factor	—	1.2
EFS	Effictive failure strain	Deletion parameter	mm/mm	0.6
XC	Longitudinal compressive strength	Material strength	GPa	1.45
XT	Longitudinal tensile strength	Material strength	GPa	1.98
YC	Transverse compressive strength	Material strength	GPa	0.234
YT	Transverse tensile strength	Material strength	GPa	0.062
SC	Shear strength	Material strength	GPa	0.091

Continued on the next page

Parameter	Definition	Type	Unit	Used Value
CRIT	Specification of failure criterion	Computational	—	54
BETA	Shear factor in tensile failure criterion	Shear weighing factor	—	0.5
SLIMT1	Factor for minimum stress limit after stress maximum (fibre tension)	Minimum stress factor	—	0.5
SLIMC1	Factor for minimum stress limit after stress maximum (fibre compression)	Minimum stress factor	—	0.605
SLIMT2	Factor for minimum stress limit after stress maximum (matrix tension)	Minimum stress factor	—	0.5
SLIMC2	Factor for minimum stress limit after stress maximum (matrix compression)	Minimum stress factor	—	0.5
SLIMS	Factor for minimum stress limit after stress maximum (shear)	Minimum stress factor	—	0.8
SOFTG	Transverse shear moduli factor after crushing failure	Damage factor	—	1

C Appendix: Stress-strain curves examples for samples with 8 plies

This section shows a few stress-strain curves of samples with 8 plies. Due to the buckling, the samples with 8 plies behave very differently than the samples with 12 or 16 plies. Therefore, the 8 plies samples are removed from the dataset. The settings of the simulations are described in the beginning of Section 4.

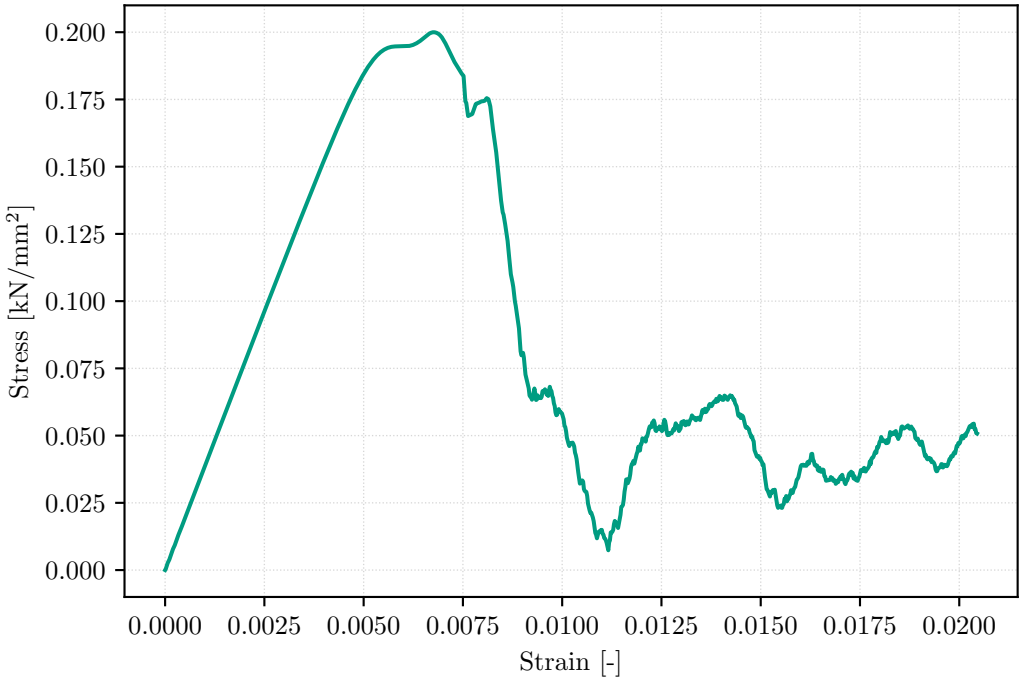


Figure C.1: Stacking $[-45°, 45°, 0°, 90°]_s$, triangular gap in ply 5 at $p = (-18.8, 7.8)$, angle $= 9.4$

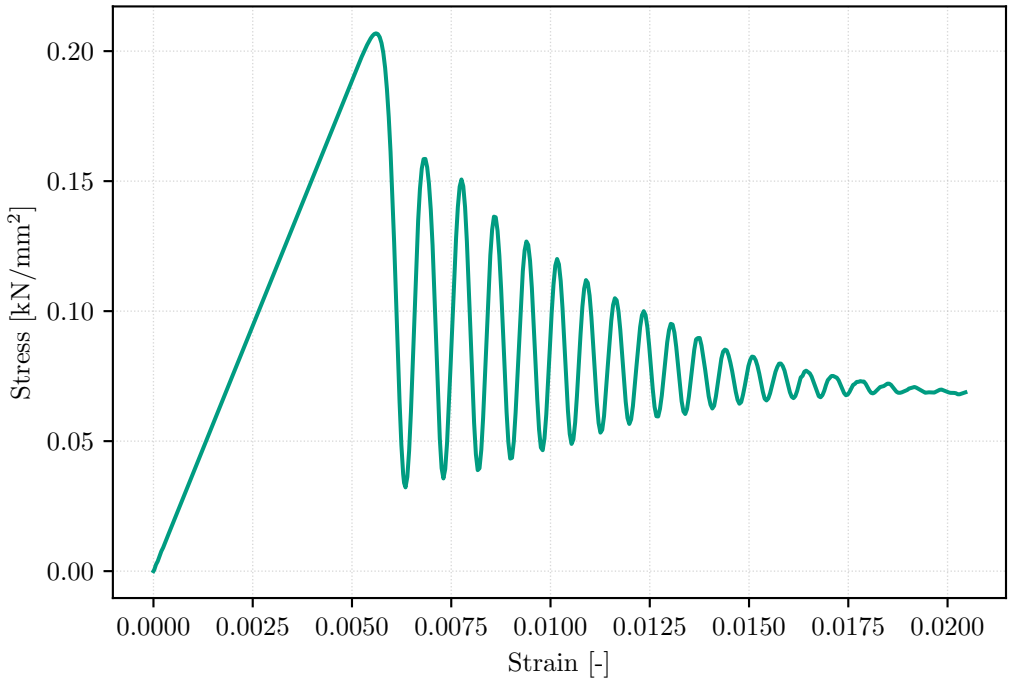


Figure C.2: Stacking $[90°, -45°, 45°, 0°]_s$, triangular gap in ply 5 at $p = (14.1, -12.8)$, angle = 9.5

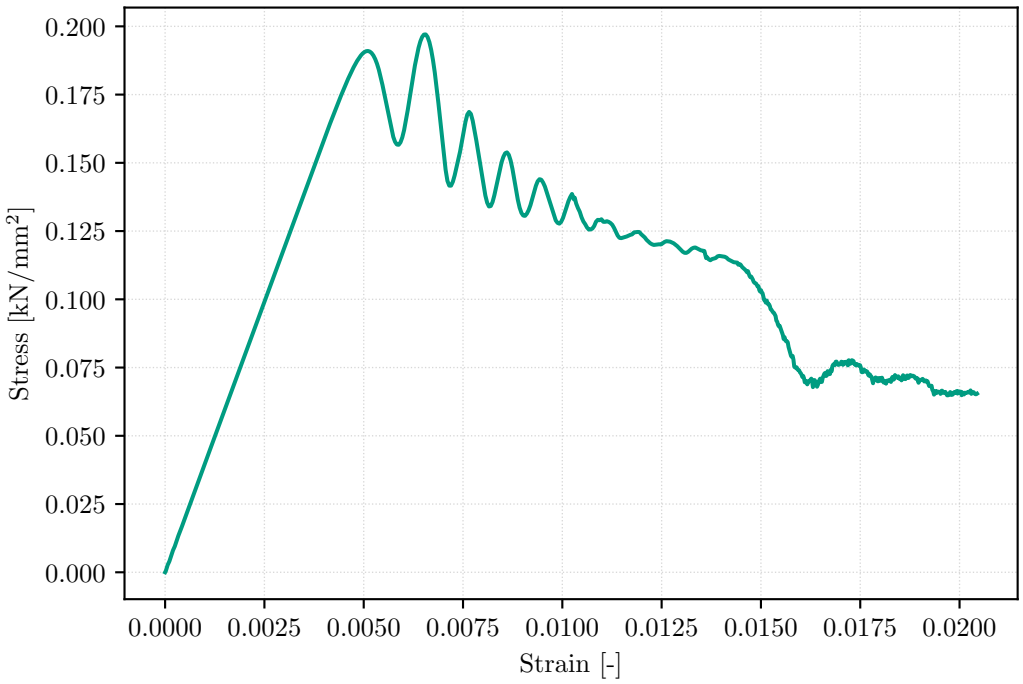


Figure C.3: Stacking $[45°, -45°, 90°, 0°]_s$, triangular gap in ply 8 at $p = (-45.3, -61.4)$, angle = 6.5

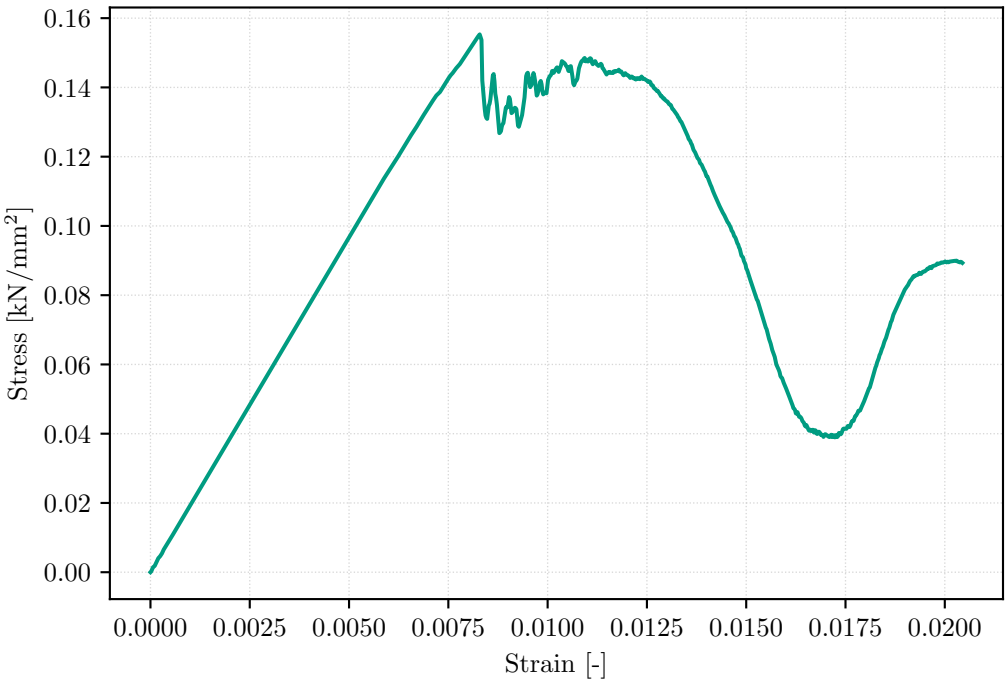


Figure C.4: Stacking $[45°, -45°, -45°, 45°]_s$, rectangular overlap in ply 4 at $p = (-0.5, 5)$, width $= 1.3$

Erklärung

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Carolin Lupprian, 01.09.2025