Bachelor Project

Czech Technical University in Prague

F6

Faculty of Transportation Sciences
Department of Mechanics and Materials

FE Simulations for Assessment of Material Response to Quasi-Static Loading

Bohumil Hora

Supervisor:
Ing. Petr Koudelka, Ph.D.,
Ing. Jan Šleichrt
Field of study: Technology in transportation and telecommunications
Subfield: Transportation Systems and Technology
November 2021

CZECH TECHNICAL UNIVERSITY IN PRAGUE
Faculty of Transportation Sciences
Dean's office
Konviktská 20, 110 00 Prague 1, Czech Republic

K618 .... Department of Mechanics and Materials

BACHELOR'S THESIS ASSIGNMENT

(PROJECT, WORK OF ART)

Student's name and surname (including degrees):

Bohumil Hora

Study programme (field/specialization) of the student:

bachelor's degree – DOS – Transportation Systems and Technology

Theme title (in Czech):	**Numerické simulace pro analýzu odezvy materiálů na kvazi-statické zatížení**
Theme title (in English):	FE simulations for assessment of material response to quasi-static loading

Guidelines for elaboration

During the elaboration of the bachelor's thesis follow the outline below:

- Prediction of deformation response of materials to uni-axial quasi-static loading using experimentally verified numerical simulations is an important part of smart materials' development.
- Using the finite element method (FEM), perform numerical simulations of uni-axial compression loading scenarios while considering both the static and dynamic formulation of the problem with the load-rate corresponding to the reference quasi-static experiment.
- On the basis of the parametric simulations, evaluate the possibility to use the dynamic formulation for prediction of the deformation response during quasi-static loading.
- Consider the following aspects in the parametric simulations: mesh density, order of shape functions, elastic, and elasto-plastic material model.
- Demonstrate the methods using simulations of both a homogeneous material and a selected 3D printed porous structure.

CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Graphical work range:	not specified
Accompanying report length:	minimum of 35 pages (including figures, graphs, and tables)
Bibliography:	Madenci E., Guven I.: The Finite Element Method and Applications in Engineering Using ANSYS, Springer-Verlag New York Inc., 2015 Shen R. W., Lei G., Introduction to the Explicit Finite Element Method for Nonlinear Transient Dynamics, John Wiley & Sons, 2012

Bachelor's thesis supervisor: **Ing. Jan Šleichrt**
Ing. Petr Koudelka, Ph.D.

Date of bachelor's thesis assignment: **October 9, 2020**
(date of the first assignment of this work, that has be minimum of 10 months before the deadline of the theses submission based on the standard duration of the study)

Date of bachelor's thesis submission: **December 1, 2021**
a) date of first anticipated submission of the thesis based on the standard study duration and the recommended study time schedule
b) in case of postponing the submission of the thesis, next submission date results from the recommended time schedule

Jiroušek

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
L. S.
- 3 -

[illegible]

prof. Ing. Ondřej Jiroušek, Ph.D.
head of the Department of Mechanics and Materials

doc. Ing. Pavel Hrubeš, Ph.D.
dean of the faculty

I confirm assumption of bachelor's thesis assignment. Hora

Bohumil Hora
Student's name and signature

Prague ....August 23, 2021

Acknowledgements

At this point, I would like to thank my thesis supervisors Ing. Petr Koudelka, Ph.D. and Ing. Jan Šleichrt for guiding, help and compliance with the writing of this bachelor's thesis. Special thanks also go to doc. dr. Nejc Novak, mag. inž. str., who kindly assisted me during my ERASMUS+ mobility.

I would also like to thank my whole family and friends for their emotional support and my parents and grandparents for their materialistic support as well, which was provided during the entirety of my studies.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

Bohumil Hora
Maribor, 29. November 2021

Abstract

The work deals with an analysis of the response of 3D-reentrant honeycomb lattice subjected to quasi-static uni-axial loading using finite element method-based parametric simulations. The problem is defined and solved both as static using the implicit algorithm in Ansys APDL and as dynamic with a strain-rate equivalent to the strain-rate of the quasi-static experiment using the explicit algorithm in LS-DYNA. Two models of the structure are presented, and the results of the simulations are then compared with the results of the real experiments.

Keywords: Finite Element Method, Quasi-Static Uni-Axial Compression, Implicit and Explicit Formulation, Homogeneous and Porous Solid, Auxetic Lattice

Supervisor:
Ing. Petr Koudelka, Ph.D.,
Ing. Jan Šleichrt

Abstrakt

Práce se zabývá analýzou odezvy 3D reentrant honeycomb struktury na kvazi-statické zatížení jednoosým tlakem s využitím konečně prvkových parametrických simulací. Simulace jsou řešeny jako statický problém implicitním algoritmem v programu ANSYS APDL a zároveň jako dynamický problém s rychlostí deformace odpovídající kvazi-statickému experimentu explicitním algoritmem v programu LS-DYNA. V práci jsou sestaveny dva modely struktury a výsledky simulací jsou porovnávány s daty ze skutečného experimentu.

Klíčová slova: metoda konečných prvků, kvazi-statický jednoosý tlak, implicitní a explicitní formulace, homogenní porézní materiál, auxetická struktura

Překlad názvu: Numerické simulace pro analýzu odezvy materiálů na kvazi-statické zatížení

Contents

Figures

Tables

Abbreviations

AM	additive manufacturing.
ASTM	American Society for Testing and Materials.
CS	cross section.
DoF	degree of freedom.
EF	element formulation.
FEA	finite element analysis.
FEM	finite element method.
LS	load-step.
MD	mesh density.
PBF	powder bed fusion.

Chapter 1

Introduction

With recent updates in the metal 3D printing technology and rising safety standards in the transportation industry, research in the high energy absorption materials becomes important. Lattice structures offer not only high energy absorption capabilities, but are also ultra-light, have higher specific stiffness and high impact impedance [1]. Because the geometry of such structures might be very complex, numerical simulations are often used to analyse their mechanical properties.

The finite element method represents a major breakthrough in the field of computational mechanics [2]. With origins in 1940s, the finite element analysis is a preferred analysis tool in the design of products and systems these days. It enables to design or examine products with complex geometries, which used to be impossible to perform. By using computer simulations rather than physical objects, it is possible to save both time and money and obtain much broader results without additional equipment [3].

This work uses FEA for parametric simulations of a quasi-static loading. The objective is to analyze the response of a homogeneous cube and a 3D reentrant honeycomb lattice structure to a quasi-static uni-axial pressure loading using finite element parametric simulations. Simulations are going to be first defined as static and solved by an implicit algorithm in ANSYS APDL, then as dynamic with strain-rate corresponding to a quasi-static experiment and solved by an explicit algorithm in LS-DYNA. In the simulations, parameters like mesh density, used elements, strain-rate, or material model are going to be varied. To perform these tasks, universal scripts for executing the simulations with changeable input parameters are going to be first created for each algorithm and each specimen. Then, different methods for automating the simulation processes are going to be developed. In the end, simulation results are going to be compared with the results of the already conducted experiments on the same structure.

In chapter 2, fundamentals of the FEM are given together with a brief introduction to additive manufacturing. Chapter 3 is devoted to the methodology of simulations, and examined specimens are more closely introduced. Implicit numerical simulations in ANSYS APDL are described in chapter 4 and explicit numerical simulations in LS-DYNA are described in chapter 5. Both chapters are divided into two subchapters, each for one specimen. The final chapter 6 is devoted to discussion of obtained results, and their applicability is evaluated.

Chapter 2

Selected theoretical topics

This chapter serves to present the reader with the topics that this work deals with. It is closely focused on the specific subjects of the work and provides an understandable overview of the discussed topics. Basic knowledge of the principles described in this chapter is necessary for understanding the following parts of the work.

2.1 Finite element method

The finite element method is a numerical method used for calculating approximate solutions of problems in a wide range of fields, including structural mechanics, heat conduction, fluid dynamics, and electric and magnetic fields [4]. It is a very universal and robust method and currently holds a dominant position among other numerical methods in the field of solid mechanics [5].

2.1.1 Analytical and numerical solution methods

Physics phenomena can be described by differential equations or by systems of differential equations together with boundary conditions. For solving these equations, there are two basic approaches - analytical solution or numerical solution.

Analytical solution

When calculating the analytical solution, we look for the result in the form of continuous functions, for the calculation of which mathematical analysis is used. This approach is advantageous because the result is a general functional dependency between the input and output variables. Such method can solve similar types of problems by simply substituting new input parameters into the already solved formula and expressing the output parameters. However, the analytical solution is often limited to the simpler geometry of the problem, the idealization of the material model, and a number of other theoretical assumptions. In engineering practice, however, the geometry is often very complex, the load is diverse, the material does not behave linearly, etc. In such cases, the analytical solution might not even exist.

2.1.2 Numerical solution

Numerical calculations generally return approximate solutions. The basis of numerical methods is the discretization process. The search for continuous functions at all points in the system switches to the search for a finite number of results in a finite number of steps. The numerical simulation result is no longer a general functional dependency, but only a finite number of results, the dependence of which on the input parameters is unknown. Whenever the input parameters change, it is necessary to recalculate the entire solution. Numerical methods often include iterative algorithms, which are hardly possible to calculate without using a computational device. On the other hand, these methods are adapted for efficient computer calculations. The advantage of numerical methods is the possibility of solving a problem that would be difficult or impossible to calculate analytically. The numerical model can have very complex geometry, nonlinear material behaviour, and other attributes for which analytical solutions do not have to exist [4].

2.1.3 Principles of FEM

The main idea of the FEM is the division of the model into a finite number of smaller parts (elements) with finite dimensions. Because the mathematical description of the behaviour of individual elements is relatively simple, behaviour of the whole model formed by these elements can also be described. The simulation procedure can be divided into the following steps:

- Discretization
- Formulation of elements behaviour
- Assembling of the equations for the whole model
- Setting of boundary conditions
- Solution - getting primary variables
- Postprocessing - calculating secondary variables [6]

The discretization process is called meshing, and as a result, a finite element mesh is created. Elements make up the whole body of the model and are interconnected through nodes. When densifying the mesh, the numerical solution approaches the analytical solution corresponding to the continuous problem. However, with a rising number of elements, the computation cost rises too, so it is always necessary to find a reasonable balance.

The goal of the formulation of elements behaviour is to get equations of motion

$$F_e = \mathbf{K}_e x_e + \mathbf{M}_e x_e'' \ , \tag{2.1}$$

for dynamic analysis, or

$$F_e = \mathbf{K}_e x_e \tag{2.2}$$

for static analysis, which describe each element. Equation (2.1) is derived from equation (2.2) by incorporating inertial forces according to the Newton's II. law and neglecting damping effects.

By assembling all equations for each element, we get the global equation of motion

$$F = \mathbf{K}x + \mathbf{M}x'' \ , \tag{2.3}$$

for dynamic analysis, or

$$F = \mathbf{K}x \tag{2.4}$$

for static analysis, which describes the whole model. By applying boundary conditions to the equation, it can be solved and the primary variables (vector of displacement) are obtained. Because many engineering tasks also require knowledge of other (secondary) variables (stress, strain, etc.), these can be calculated in the postprocessing part from obtained displacements.

2.1.4 Implicit and explicit algorithm

FEM simulations can be solved using two different algorithms - implicit and explicit. Choice of the proper algorithm is mostly determined by the duration of the problem. For slow-movement long-duration problems (e.g., material creep), the implicit method is preferred, while for fast-movement short-duration problems (e.g., impact involved analysis), the explicit method is recommended [7]. Reasons for this are going to be clarified in this subsection.

While the implicit algorithm can solve both static and dynamic problems as they are, the explicit algorithm can directly solve only dynamic problems. This drawback, however, might be bypassed by rephrasing the static problem to a quasi-static one. Because this work deals with implicit simulations of a static problem and explicit simulations of a dynamic problem, these instances will be described.

Implicit

Implicit solution of a linear static problem would be prescribed by the equation

$$F = \mathbf{K}x \ , \tag{2.5}$$

which is relatively easy to solve. Because nonlinearities are present in the simulations throughout this work (such as nonlinear material model, large deformations, and large rotations), stiffness matrix **K** is actually dependent on the vector of displacement x as in equation 2.6.

$$F = \mathbf{K}(x)x \tag{2.6}$$

For the solution of such a system of equations, Ansys APDL uses the Newton-Rhapson algorithm, which is a numerical root-finding iterative algorithm. It produces successively better approximations of the roots until APDL convergence criteria are met [8, 9].

This algorithm is unconditionally stable, which means that no matter how long time-step is chosen (note that, as the analysis is static, time corresponds only to a measure of deformation and has actually nothing to do with a real time), the solution will not diverge. Because at every load-step, the computationally demanding equation 2.6 has to be solved, choosing the time-step as long as possible is recommended [4].

Explicit

Explicit solution of a dynamic problem would be prescribed by the equation

$$F = \mathbf{K}x + \mathbf{M}x'' \; . \tag{2.7}$$

For the explicit algorithm, LS-DYNA uses the finite difference method [10]. From the differential formulas, relations for velocity and acceleration can be expressed

$$x'_{n+1} = \frac{x_{n+1} - x_n}{\Delta t} \; , \tag{2.8}$$

$$x''_{n+1} = \frac{x'_{n+1} - x'_n}{\Delta t} \; . \tag{2.9}$$

By combining equations (2.8) and (2.9), vector of accelerations can be approximated by the vectors of deformations

$$x''_{n+1} = \frac{x_{n+1} - 2x_n + x_{n-1}}{\Delta t^2} \; . \tag{2.10}$$

By putting together equations (2.7) and (2.10), we obtain a formula for calculating the vector of displacements at time t_{n+1} from the vectors of displacements at two previous time-steps

$$\frac{\mathbf{M}}{\Delta t^2} x_{n+1} = F_n - \mathbf{K}x_n + \mathbf{M}\frac{2x_n - x_{n-1}}{\Delta t^2} \; . \tag{2.11}$$

As LD-DYNA uses a diagonal mass matrix for computing, the vector of displacements x_{n+1} can be relatively easily obtained from this equation [10].

The solution of an explicit algorithm is conditionally stable. The stable solution can be obtained only by keeping time-step sufficiently small, i.e., smaller than the critical time-step t_c, which for one element is calculated as

$$\Delta t_c = \frac{l_c}{\sqrt{\frac{E}{\rho}}} \; , \tag{2.12}$$

where l_c is the characteristic dimension of the element. As a characteristic dimension, LS-DYNA uses the volume of the element V_e over the area of its largest side $A_{e,max}$ for hexahedrons and the element's minimal altitude for tetrahedrons [11]. Critical time-step for the whole model is then determined by the smallest time-step required by all elements. In LS-DYNA simulations, time-step $0.9 \cdot t_c$ is used by default [12].

Time-step in explicit algorithms is usually $100 - 1000\times$ smaller than the one used in implicit algorithms. However, the solution of every iteration is much less computational demanding than in an implicit algorithm. Because there are so many time-steps and because after every time-step the stiffness matrix is updated according to the current state, there is no need for iterative algorithms in the root-finding procedure as in explicit algorithm [6].

Computation time acceleration

Sometimes the available hardware might not be powerful enough to accommodate simulations that one would like to perform. Therefore, it is important to know what techniques might be used to shorten their computation time. Even though getting more powerful hardware would work, it is not always feasible. Some other simulation settings-based techniques used in this work will be described.

For both implicit and explicit simulations, the computation time can be reduced by using less dense mesh and simpler element formulations. Despite both of these cases generally leading to lower accuracy of results, a reasonable balance between accuracy and low computation time always needs to be found.

For implicit simulations, a longer time-step can be used to speed up simulations. The drawback of prolonging the time-step is obtaining fewer points from which the results can be read. Even though the results can be acquired from sub-steps, which are smaller steps in each time-step when Newton-Rhapson's algorithm is performed, to get results from evenly spaced data points, appropriate time-step to get the desired data from the simulation must be chosen.

For explicit simulations, two main approaches are used. The first one is simply shortening the simulation time, which was done either by speeding up the deformation or by deforming the object to a smaller extent. The other approach is to prolong the time-step by mass scaling. Mass scaling is a process in which a nonphysical mass is added to the elements so that their critical time-step t_c is longer (according to equation 2.12). This is done in a selective manner where the density of smaller elements is increased so that their critical time-step gets longer [12, 13].

2.1.5 Material models

A material model is a mathematical description of material behaviour under loading. When performing FEM simulations, the selection of a correct material model is crucial. Even though material properties are predetermined by the material microstructure, the material model is usually defined according to the results of experiments with macro samples. In the figure 2.1 real deformation curve from a compression test is shown.

In this work, two material models are considered - linear and bilinear with isotropic hardening. Despite the fact that a more detailed material model would yield more accurate results, it is not always preferred. A more detailed material model comes with a higher computational cost. Moreover, it is not

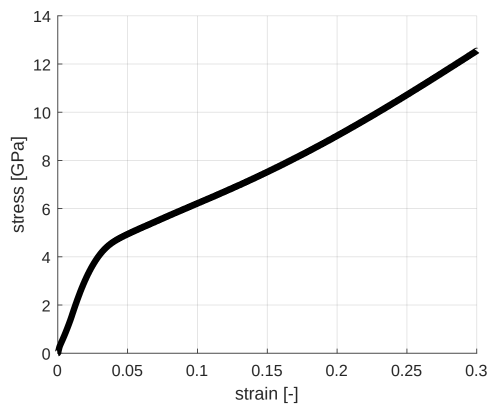


Figure 2.1: Engineering stress-strain curve from an experimental compression test [14].

always possible to determine the material model in sufficient detail due to a lack of experimental data for the given material.

Linear elastic material model

For a linear material model, Hooke's law is assumed to be in effect in the whole spectrum of deformations. Although this is accurate only to the yield point, the main advantage is low computational complexity. Even though the real material does not behave in this way, since plastic deformations are not expected for most manufactured products, linear elastic material model is very suitable in certain cases. Parameters characterizing this model are Young's modulus and Poisson's ratio.

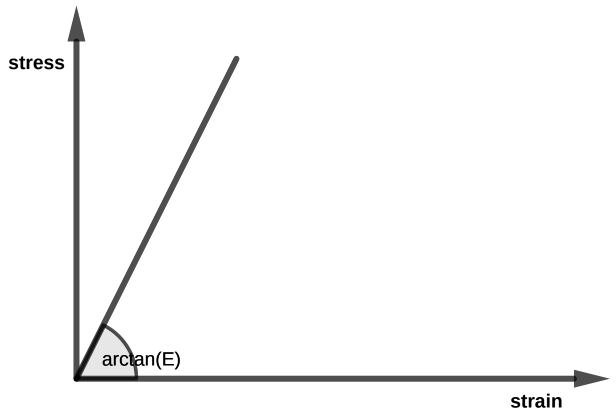


Figure 2.2: Linear material model.

Bilinear material model with isotropic hardening

This model is approximated by two linear functions, each corresponding to the elastic and plastic deformation, respectively. The deformation curve starts as the elastic material model, but after reaching the yield point (Re), the slope of the curve is changed from E to E_{tan}. As can be seen by comparing the graphs in figures 2.1 and 2.3, this bilinear approximation actually captures the experimental data very well.

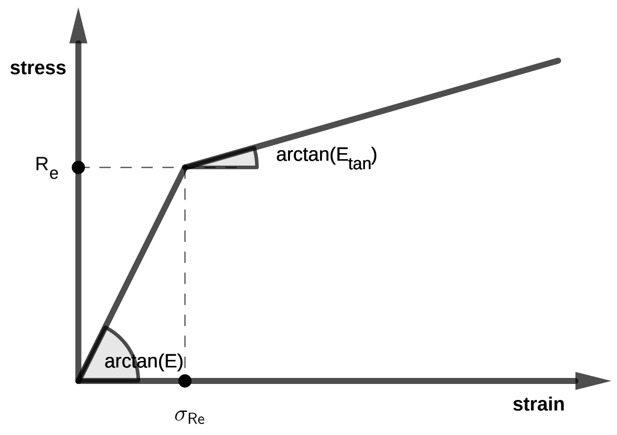


Figure 2.3: Bilinear material model.

This deformation curve would look the same for a bilinear model with kinematic hardening. The difference between these types is the transformation of the yield surface. It remains the same shape but expands with increasing stress for isotropic hardening and remains the same shape and size but translates in stress space for kinematic hardening [15]. Such difference would become visible under unloading. However, since only monotonic loads are considered in this thesis, the difference is insignificant, and a model with isotropic hardening will be used. Such a material model is characterized by four constants - Young's modulus, Poisson's ratio, Yield strength, and tangential modulus [16, 17].

2.2 Additive manufacturing

Additive manufacturing (AM), popularly known as 3D printing, is a revolutionary industrial technology. It is defined as the "process of joining materials to make parts from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing and formative manufacturing methodologies" [18]. Such manufactured components can be fabricated with an unprecedented degree of freedom and can be printed from a wide range of materials, including metals, ceramics, and polymers [19]. Currently, there are 7 AM methods - binder jetting, directed energy deposition, material extrusion, material jetting, powder bed fusion, sheet lamination and vat photopolymerization [18]. As the examined structure was printed by the powder bed fusion (PBF) method, attention will be focused on this method.

2.2.1 Powder bed fusion

PBF is defined as the "process in which thermal energy selectively fuses regions of a powder bed" [18]. This method can be further divided into two subcategories according to the energy source—laser-based PBF for laser and electron beam-based PBF for a beam of electrons.

The whole manufacturing process takes place in a closed chamber filled with vacuum or inert gas (see fig. 2.4). The object in production is submerged in the powder with an additional thin layer sitting on top. The heat source selectively melts regions of the powder, which bond with the object, contributing to its final shape. After the whole layer is sintered, the platform moves down by a thickness of the layer, and a new powder is dispersed by the powder roller on the top again. This process repeats until the final object is done [20].

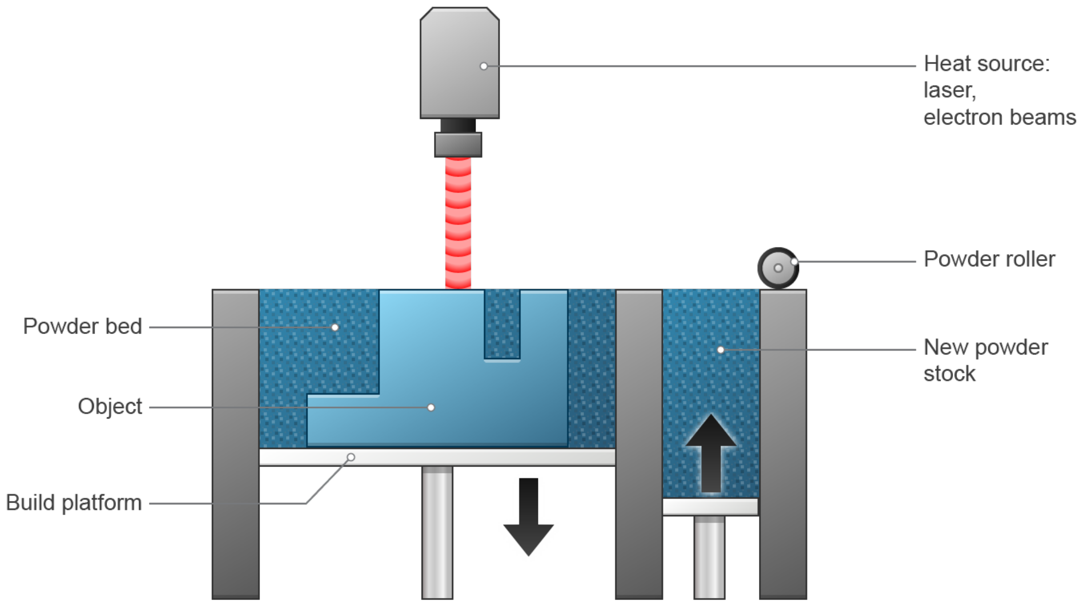


Figure 2.4: Scheme of powder bed fusion manufacturing process [21].

Chapter 3

Methodology of simulations and examined specimens

To achieve the objectives of the work, the simulations will first be defined as static and solved by an implicit algorithm in Ansys APDL. In the next part as dynamic with a strain-rate corresponding to a quasi-static experiment and solved by an explicit algorithm in LS-DYNA. For this, parametric simulations will be used, in which mesh density, mesh type and used elements will be varied. Additionally, different models of the structure will be explored for implicit solution, as well as two deformation speed modes and strain-rates for the explicit one.

The number of simulations on the structure needed to capture all combinations of input parameters is too high. In order to limit the number of simulations to a reasonable amount, a set of similar simulations will be first tested on an imaginary specimen - a homogeneous cube. By doing so, it will be possible to determine how the input parameters contribute to the results, and inappropriate input parameters will then be omitted in simulations on the structure.

As a partial measure of the effectiveness of individual simulations, the time needed to solve them will be used. Therefore, all calculations were performed on the same computer with a 12-core Intel Xeon W-2265 processor, each core with a base frequency of 3.5 GHz and two threads [22]. The amount of available RAM was 256 GB but this value has never been reached. The graphics card was not used for calculation acceleration. Licenses of the software did not allow to reach the maximal CPU potential. During the simulations in APDL, the typical CPU utilization was about 48 %, and only one simulation at a time could be solved. For the simulations in LS-DYNA, the utilization was about 4.5 % per simulation, and up to 20 simulations could be solved simultaneously.

3.1 Homogeneous cube

The homogeneous cube sample was chosen with an edge length of 10 mm, as it approximately corresponds to the edge size of the structure. Model of the cube is shown in figure 3.1.

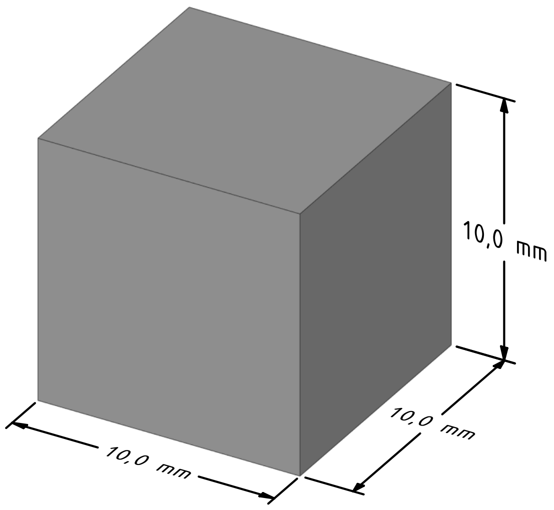


Figure 3.1: Model of the cube

3.2 3D printed lattice structure

The model of the 3D printed lattice structure is shown in the figure 3.2. It is a 3D re-entrant honeycomb lattice structure, which, thanks to the unique internal arrangement of the beams, has a negative Poisson's ratio in all three directions [23]. The structure was chosen for this work thanks to its interesting mechanical properties and also performed experiments, which can be used to validate the results of the simulations.

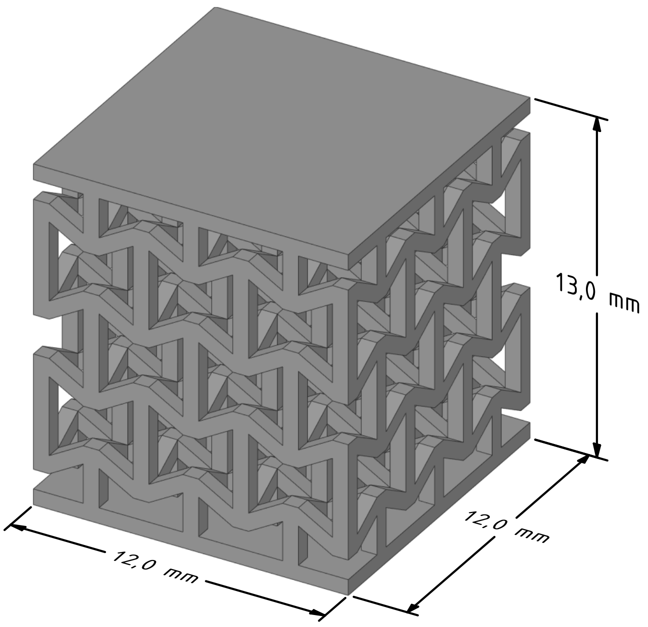


Figure 3.2: Model of the structure.

The actual samples were printed on an AM 250 device (Renishaw, UK) using the L-PBF (Laser Powder Bed Fusion) method by laser sintering of austenitic 316L-0407 stainless steel powder composed of iron with 16-18 % chromium, 10-14 % nickel, 2-3 % molybdenum. With a maximum carbon content of up to 0.03 %, this is a modification of the standard SS316L alloy [24, 25].

The nominal dimensions of the sample are $12.0 \times 12.0 \times 13.0$ mm (width $\times$ depth $\times$ height) with nominal porosity of 72.3 %, the number of cells in the structure is $2 \times 2 \times 3$ (along the width $\times$ depth $\times$ height). The cross section of the beams in the structure is square-shaped with an edge length of 0.6 mm. The dimensional parameters of the structure cells are shown in figure 3.3 and described in table 3.1.

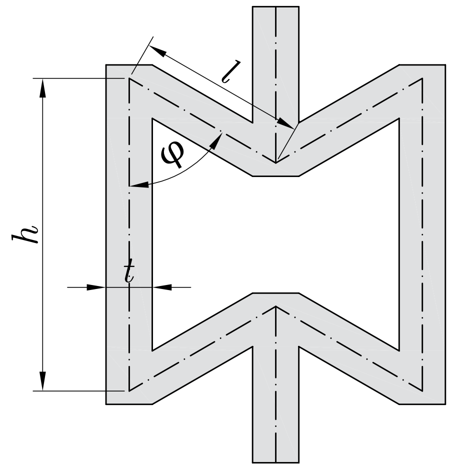


Figure 3.3: Scheme of dimensional parameters of the structure cells (in proportion).

label	value	unit
h	4.054	mm
l	2.194	mm
t	0.6	mm
φ	60	°

Table 3.1: Dimensional parameters of the structure cells.

3.3 Material model

Both specimens were assigned the same material model, which is based on already performed experiments on printed material samples. The source of data for the material model was the material data sheet provided by the manufacturer [26], the article [14], in which the yield strength and the tangential modulus of elasticity were established, and dissertation [27], which dealt with structures printed by the same method from the same material, while the parameters used in it are based on the previous two sources.

The manufacturer's material data sheet shows Young's modulus of elasticity and the yield strength for two angles (0° and 90°) between the direction of printing and the direction of deformation. The results were carried out from a quasi-static tensile test according to the ASTM E8 standard.

The outcomes of the article are the yield strength and the tangential modulus of elasticity for three angles (0°, 45° and 90°) between the direction of printing and the direction of deformation determined from the quasi-static pressure test. Since the deformation was determined from the displacement of the cross-head using an extensometer, it is not possible to reliably determine Young's modulus of elasticity from these data, and other conclusions of this work must also be taken with caution.

Material constants from these three sources are listed in table 3.2.

source	angle [°]	E [GPa]	R_e [MPa]	E_{tan} [GPa]	ρ [g/cm^3]
material data sheet [26]	0	190 $\pm$ 10	494 $\pm$ 14	-	7.99
	90	197 $\pm$ 4	547 $\pm$ 3	-	
article [14]	0	-	415 $\pm$ 34	2.48 $\pm$ 0.13	7.52 $\pm$ 0.17
	45	-	443 $\pm$ 4	2.32 $\pm$ 0.12	
	90	-	496 $\pm$ 33	2.23 $\pm$ 0.11	
dissertation [27]	-	206	520	2.32	7.52

Table 3.2: Material model parameters from different sources (values used for determining the parameters of the material model are highlighted).

Although the material showed signs of anisotropy due to the manufacturing process, the material model was considered isotropic in this work. There were not enough experiments carried out to validate each other's results, the parameters from the performed experiments do not really match, and the measurement errors are significant. This decision will also help to maintain the simplicity and clarity of the simulations.

For the purposes of this thesis, Young's modulus of elasticity was considered as the average of the average of the moduli of elasticity from the material data sheet and the modulus of elasticity used in the cited dissertation. Yield strength was considered as the average yield strength from the material data sheet. Tangential modulus was considered as the average of tangential moduli from the article. Density was considered as the average of the densities from the material data sheet and the article. The Poisson's ratio of the material was considered 0.3. Material parameters used in the simulations are summarised in the table 3.3.

material property	label	value	unit
Young's modulus	E	200	GPa
Poisson's ratio	μ	0.3	–
Yield strength	R_e	520	MPa
Tangential modulus	E_{tan}	2.34	GPa
Density	ρ	7.76	$\mathrm{g/cm^3}$

Table 3.3: Parameters of the material model used in the simulations

Since one material model, which will be used in the chapter 5, requires as input the shear modulus G and the bulk modulus K, these two moduli were calculated from the determined constants according to formulas (3.1) and (3.2).

$$G = \frac{E}{2(1+\nu)} = 79\,\mathrm{GPa} \tag{3.1}$$

$$K = \frac{E}{3(1-2\nu)} = 172\,\mathrm{GPa} \tag{3.2}$$

To define the contact between elements in the subchapter 5.3, the static coefficient of friction μ_s and the dynamic coefficient of friction μ_d were determined [10, 28].

$$\mu_s = 0.78 \tag{3.3}$$

$$\mu_d = 0.42 \tag{3.4}$$

3.4 Validation of numerical results with experimental results

An essential part of numerical simulations is the validation of the obtained results. Although no actual experiment has been performed on the cube, the verification of the obtained results will be relatively straightforward. Because it is a very simple shape, the deformation curve obtained from the cube simulations should correspond to the deformation curve of the chosen material model.

Results from the simulations of the structure are going to be validated differently. There was an experiment performed with the structure - quasi-static uni-axial pressure test using Instron 3382 device with a cross-head speed of 0.5 mm/min to a maximum overall deformation of 50 %. Using digital image correlation, the displacements were read from 6 points marked in the figure 3.4 and the deformation was taken as the average of the approaches of these three pairs of points. The value 8.106 mm marked in the figure was taken as the initial length l_0 both in the experiment and in this work [27].

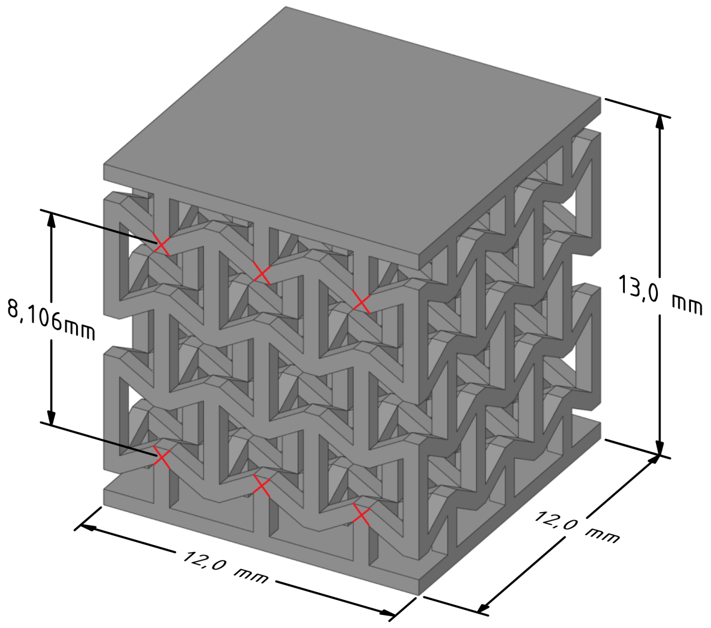


Figure 3.4: Points from which the displacements were read in the experiment.

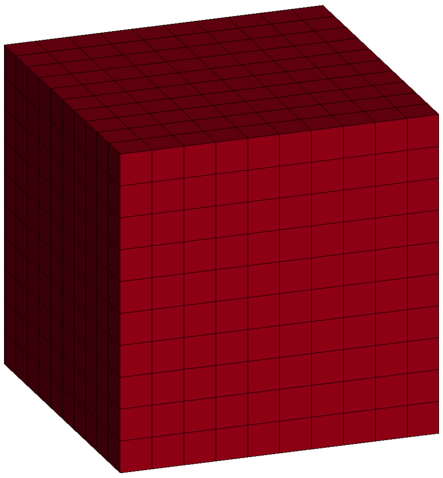
Chapter 4

Implicit numerical simulations in ANSYS APDL

The objective of this chapter is to perform several simulations with different input parameters in the APDL programming language separately for both specimens, from which the necessary results will be obtained and processed. Although it would be possible to write only one script in APDL for each specimen to perform all simulations, with a lack of advanced functions, this code would be long and confusing. For easier results processing and plots production, it would still be necessary to use other programming tools too. Therefore, the entire core of automation was written only in the Matlab programming environment, where it is possible to define simulation parameters, generate scripts for APDL, load the calculated results, and process them, with one script for each specimen.

4.1 Cube

This subchapter is dedicated to parametric implicit simulations of the quasi-static deformation of the homogeneous cube presented in the subchapter 3.1.


(a) : Mapped mesh.

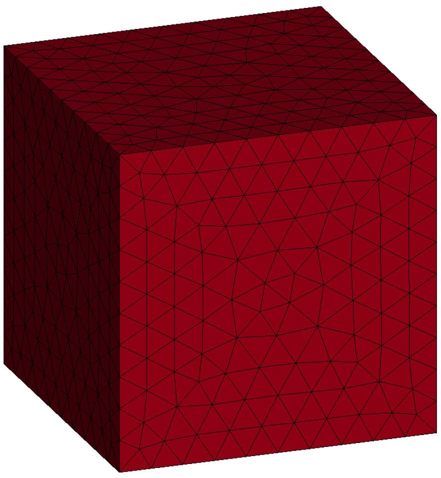

(b) : Free mesh.

Figure 4.1: Example of mapped and free mesh with 10 elements along the edge.

4.1.1 Investigated simulation parameters

Simulations were mainly focused on the parameters presented in this section.

Mesh density

The mesh density was controlled by the number of elements along all edges of the cube. Simulations were created for the number of elements along the edge in the range of 1-20.

Mesh type

Two mesh types were used in the simulations, which are referred to in APDL as mapped and free. The mapped mesh is formed by regularly arranged hexagonal elements, while the free mesh consists of tetrahedral elements formed by the APDL meshing algorithm. An example of these two types of mesh with 10 elements along the edge is shown in figure 4.1.

Elements

A hexagonal linear element SOLID185 and a hexagonal quadratic element SOLID186 were used for the mapped mesh. For the free mesh, the same elements were used as for the mapped mesh (both elements allow merging of several nodes together to create a tetrahedral element), together with the tetrahedral linear element SOLID285 and the tetrahedral quadratic element SOLID187. The nodes of all used elements have 3 DoFs - translations in the direction of the axes x, y, and z. All elements support plasticity and large deformations [29]. Schemes of all used elements are shown in figure 4.2.

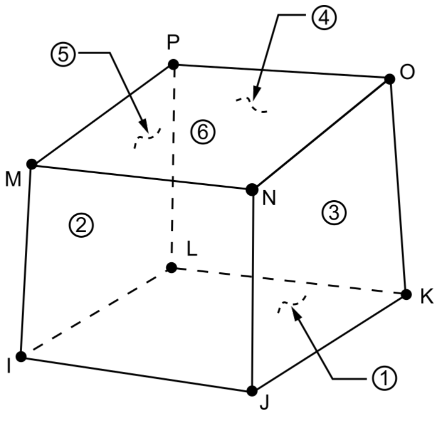
4.1.2 Principles of simulations

The *APDL_cube_deformation* script first stores the current time on the computer for later determination of the calculation time. The command *TB, BISO* defines bilinear material model and the command *ET* the element type.

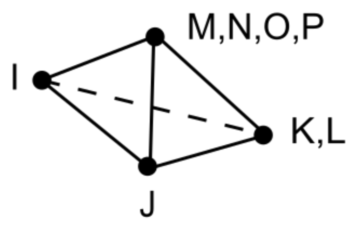
The geometry of the model is created by first defining the eight keypoints that make up the corners of the cube. Between the respective pairs of keypoints, 12 lines are created, which are divided by the command *LESIZE* into the number of parts corresponding to the selected mesh density. From the respective four lines, six surfaces are then formed, which form the walls of the cube, by which the volume of the cube is finally created.

Then, depending on the simulation parameters, the *MSHKEY* command defines the mesh type (mapped or free), and the *MSHAPE* command defines the element type (tetrahedral or hexahedral). The *VMESH* command then meshes the volume of the cube.

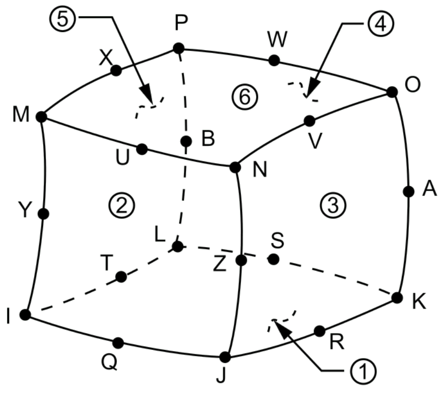
Subsequently, the bottom wall of the cube is flexibly supported so that a zero displacement in the z-axis direction (i.e., in the deformation direction) is introduced to all nodes lying thereon. Zero displacements in the plane of the base and perpendicular to the edge are also introduced to the nodes at



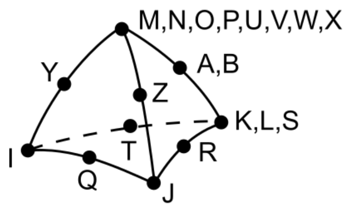
(a) : Default element SOLID185.



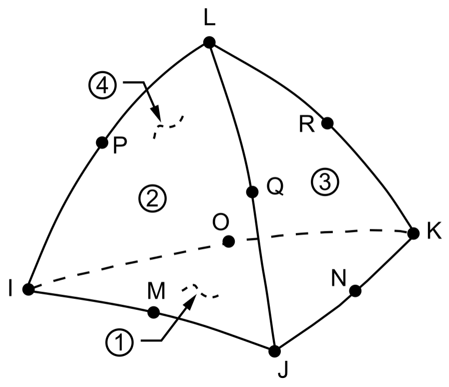
(b) : Modification of element SOLID185 into a tetrahedral element.



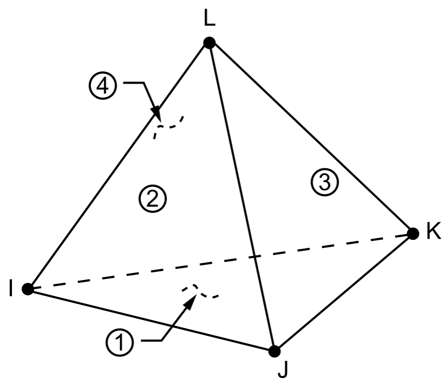
(c) : Default element SOLID186.



(d) : Modification of element SOLID186 into a tetrahedral element.



(e) : Tetrahedral element SOLID187.



(f) : Tetrahedral element SOLID285.

Figure 4.2: Schemes of SOLID type elements used in simulations with the cube [29].

two adjacent edges of the base. The support is shown in figure 4.3. The cube is supported in such a way that when deformed, the cube is free to expand yet held in its place.

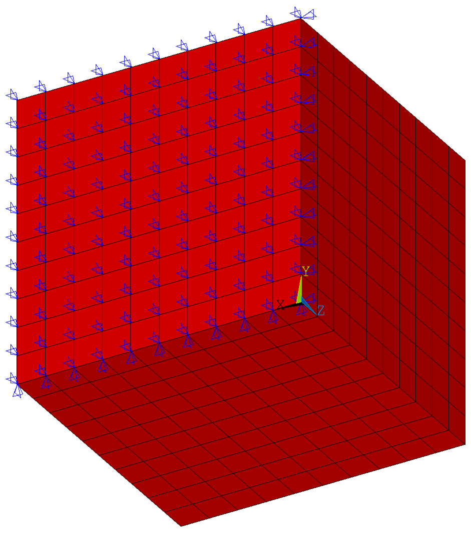


Figure 4.3: Illustration of supporting the base of the cube (blue "pyramids" indicate translation constrains of the nodes at their tips in the direction in which they point).

Furthermore, in the individual load-steps, the introduction of forced displacement to the nodes of the upper wall and the solution of such loaded model repeats iteratively. After calculating all load-steps, the reaction forces in the upper wall at each load-step are together with the relative deformation and the solution time stored in the *results* folder named *results_id.txt*, where *id* is the identification number of the simulation.

4.1.3 Automation

The whole process of creating and calculating simulations and loading and processing results was automated into a few steps. An overview of the automation process is in figure 4.4.

Prerequisites

The following folders and files must be stored in the same directory for proper functioning (see figure 4.5):

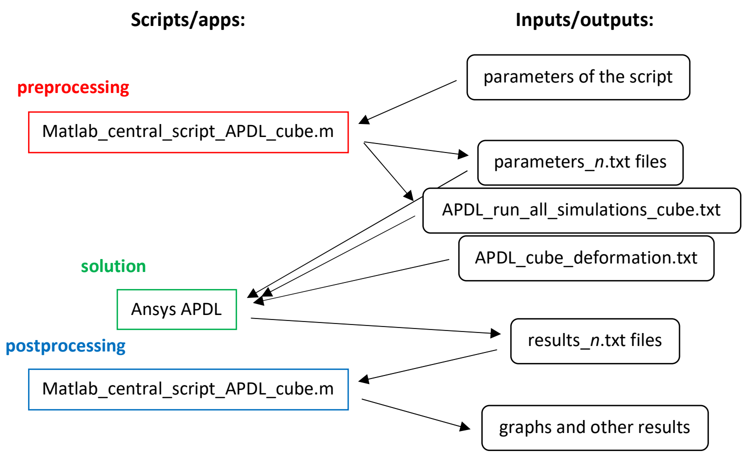
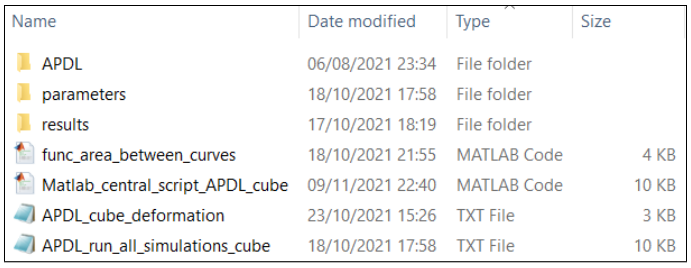


Figure 4.4: Block diagram of the simulations performation process in APDL with the cube.

- folder *APDL* - working directory for APDL
- folder *parameters* - parameters of the individual simulations will be generated here
- folder *results* - results of the individual simulations will be saved here
- file *func_area_between_curves.m* - function which calculates the area between 2 curves
- file *Matlab_central_script_APDL_cube.m* - script which is the foundation of the automation process
- file *APDL_cube_deformation.txt* - script for the cube deformation simulation
- file *APDL_run_all_simulations_cube.txt* - script, which runs all generated simulations

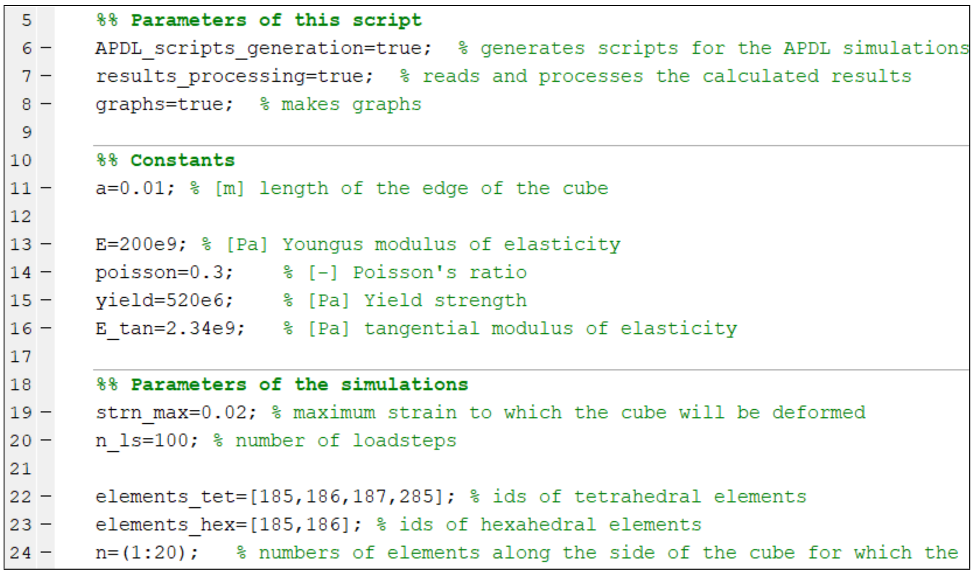


Name	Date modified	Type	Size
APDL	06/08/2021 23:34	File folder	
parameters	18/10/2021 17:58	File folder	
results	17/10/2021 18:19	File folder	
func_area_between_curves	18/10/2021 21:55	MATLAB Code	4 KB
Matlab_central_script_APDL_cube	09/11/2021 22:40	MATLAB Code	10 KB
APDL_cube_deformation	23/10/2021 15:26	TXT File	3 KB
APDL_run_all_simulations_cube	18/10/2021 17:58	TXT File	10 KB

Figure 4.5: Necessary items in the directory.

Conduction of parametric simulations

The generation of simulation parameters was done using the script *Matlab_central_script_APDL_cube.m* in the Matlab environment. In the section *Parameters of this script*, parameters of the script can be defined. It is first necessary to select that the simulation parameters will be generated, select desired parameters of the simulations in sections *Constants* and *Simulation parameters* and run the script. An illustration of the script set to generate parameter files for APDL with the parameters that were used in subsection 4.1.4 is shown in the figure 4.6.



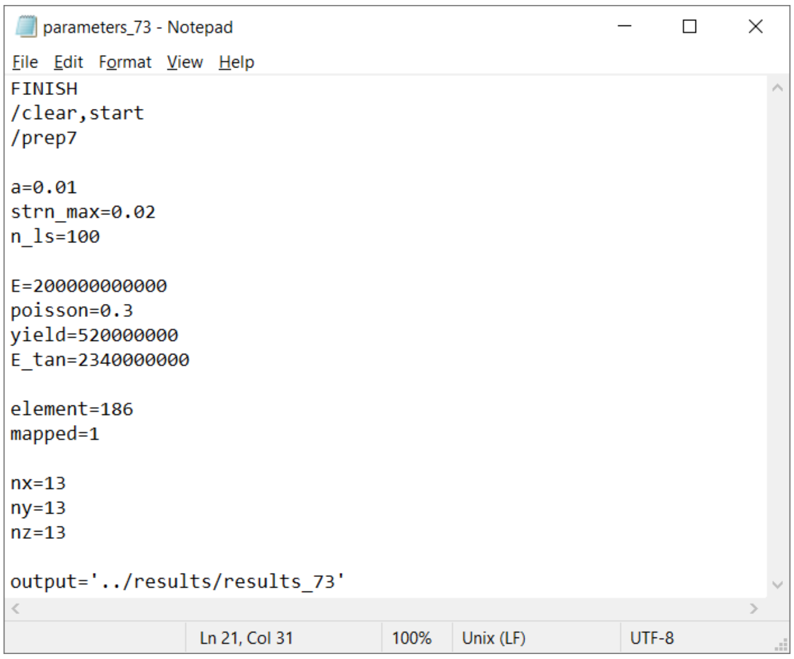
```
%% Parameters of this script
APDL_scripts_generation=true;  % generates scripts for the APDL simulations
results_processing=true;  % reads and processes the calculated results
graphs=true;  % makes graphs

%% Constants
a=0.01; % [m] length of the edge of the cube

E=200e9; % [Pa] Youngus modulus of elasticity
poisson=0.3;    % [-] Poisson's ratio
yield=520e6;    % [Pa] Yield strength
E_tan=2.34e9;   % [Pa] tangential modulus of elasticity

%% Parameters of the simulations
strn_max=0.02; % maximum strain to which the cube will be deformed
n_ls=100; % number of loadsteps

elements_tet=[185,186,187,285]; % ids of tetrahedral elements
elements_hex=[185,186]; % ids of hexahedral elements
n=(1:20);   % numbers of elements along the side of the cube for which the
```

Figure 4.6: Illustration of the defined constants and parameters in Matlab.

By running the script set in this way, files named *parameters_id.txt* are created in the *parameters* folder for all possible combinations of the input parameters. An example of one parameter file is in figure 4.7.

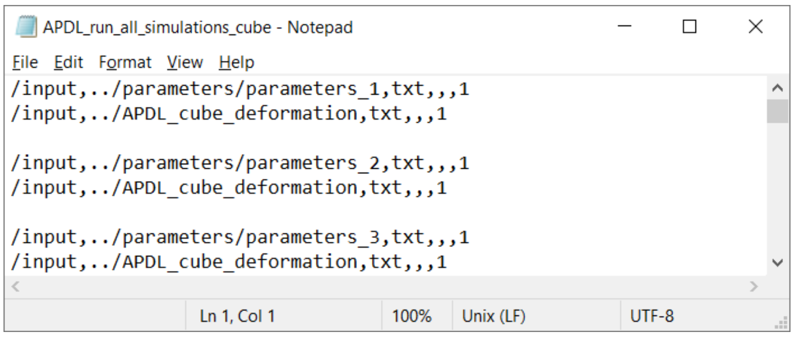
At the same time, a script named *APDL_run_all_simulations_cube.txt* is created (or updated) in the directory of the script (example in figure 4.8).

This script has to be loaded into the APDL, where it alternately loads the parameter files and the script for the cube deformation simulation. After each simulation, the *results_id.txt* file is saved in the *results* folder, which



```
FINISH
/clear,start
/prep7

a=0.01
strn_max=0.02
n_ls=100

E=200000000000
poisson=0.3
yield=520000000
E_tan=2340000000

element=186
mapped=1

nx=13
ny=13
nz=13

output='../results/results_73'
```

Figure 4.7: Illustration of a *.txt* file with parameters.

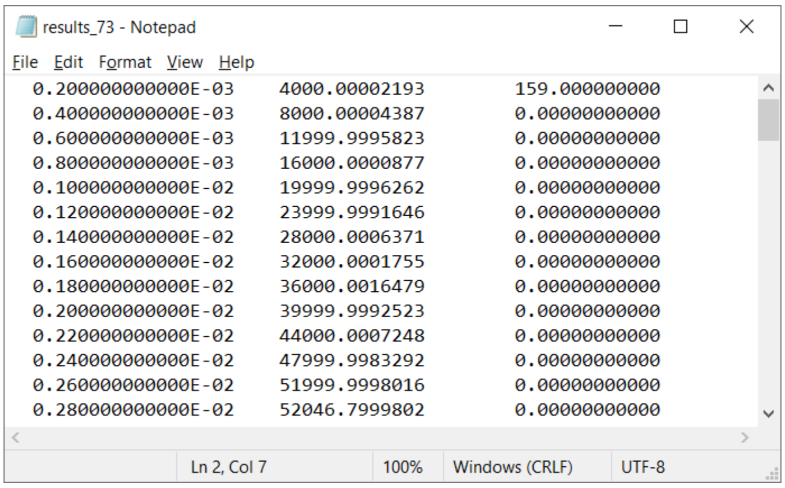

APDL_run_all_simulations_cube - Notepad
File Edit Format View Help
Ln 1, Col 1
100%
Unix (LF)
UTF-8

```
/input,../parameters/parameters_1,txt,,,1
/input,../APDL_cube_deformation,txt,,,1

/input,../parameters/parameters_2,txt,,,1
/input,../APDL_cube_deformation,txt,,,1

/input,../parameters/parameters_3,txt,,,1
/input,../APDL_cube_deformation,txt,,,1
```

Figure 4.8: Illustration of the APDL script which runs all the simulations.

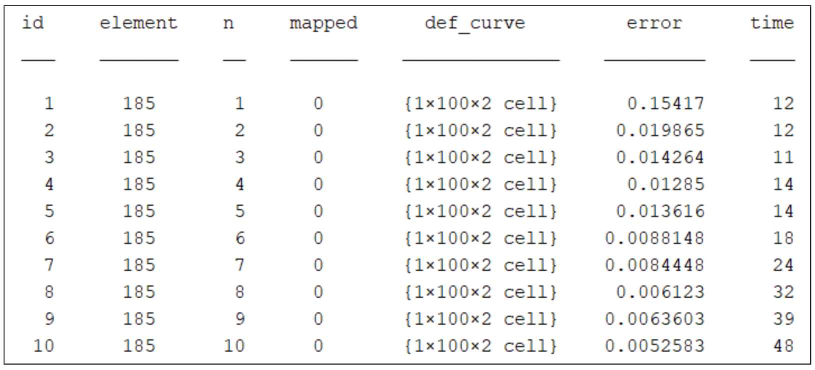
contains data of the deformation curve and the computation time. An example is in figure 4.9.

After completing the computation of all simulations, the result files can be automatically loaded into one complete table by running the script *Matlab_central_script_APDL_cube.m* again (this time set to process results). Example can be seen in figure 4.10. The data sorted in this way are then easy to work with and are ready for further processing.



```
0.200000000000E-03     4000.00002193        159.000000000
0.400000000000E-03     8000.00004387        0.00000000000
0.600000000000E-03     11999.9995823        0.00000000000
0.800000000000E-03     16000.0000877        0.00000000000
0.100000000000E-02     19999.9996262        0.00000000000
0.120000000000E-02     23999.9991646        0.00000000000
0.140000000000E-02     28000.0006371        0.00000000000
0.160000000000E-02     32000.0001755        0.00000000000
0.180000000000E-02     36000.0016479        0.00000000000
0.200000000000E-02     39999.9992523        0.00000000000
0.220000000000E-02     44000.0007248        0.00000000000
0.240000000000E-02     47999.9983292        0.00000000000
0.260000000000E-02     51999.9998016        0.00000000000
0.280000000000E-02     52046.7999802        0.00000000000
```

Figure 4.9: Illustration of a *.txt* file with results (left to right: strain [-], reaction force [N], computation time [s]).



```
id    element    n    mapped      def_curve           error      time
___   _______   __    ______    _______________    _________    ____

 1      185      1      0       {1×100×2 cell}       0.15417     12
 2      185      2      0       {1×100×2 cell}      0.019865     12
 3      185      3      0       {1×100×2 cell}      0.014264     11
 4      185      4      0       {1×100×2 cell}       0.01285     14
 5      185      5      0       {1×100×2 cell}      0.013616     14
 6      185      6      0       {1×100×2 cell}     0.0088148     18
 7      185      7      0       {1×100×2 cell}     0.0084448     24
 8      185      8      0       {1×100×2 cell}      0.006123     32
 9      185      9      0       {1×100×2 cell}     0.0063603     39
10      185     10      0       {1×100×2 cell}     0.0052583     48
```

Figure 4.10: Illustration of the database of the parameters and results.

4.1.4 Simulations set 1

This set of simulations is focused on determining the usability of different mesh types, mesh densities, and elements.

Used parameters

A total of 120 simulations were performed, in which the mesh density, mesh type, and used elements were varied. For both mesh types and their corresponding elements, 20 simulations were created with numbers of elements along the edge in the range of 1-20. Used parameters are summarised in figure 4.11.

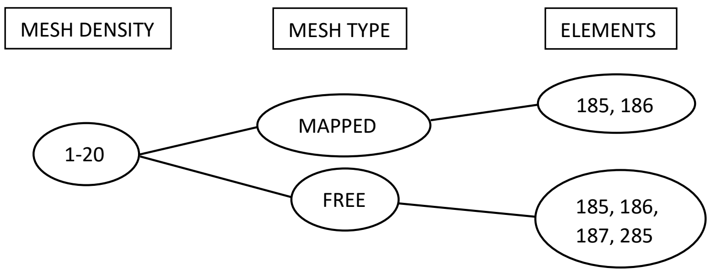


Figure 4.11: Scheme of used parameters for simulations set 1 with the cube in APDL.

The cube was deformed to a maximum relative deformation of 2 % at 100 LS. The maximum relative deformation of 2 % was chosen because in the simulations that preceded the appropriate choice of parameters, it was found that after exceeding the yield strength, the deformation curves only oscillate around the prescribed deformation curve. Increasing the maximum deformation would therefore not bring any new data. 100 LS were chosen to obtain a relatively detailed stress-strain curve with acceptable computation needs.

Results

The area between the prescribed deformation curve given by the material model and the calculated deformation curve was used as a measure to quantify the correctness or incorrectness of the presented results. The graph of the error dependence on the number of elements for different elements is shown in the figure 4.12.

From the graph in the figure 4.12 it can be seen that tetrahedral elements are inaccurate in a sparse mesh and gradually get better as the mesh gets denser, while the linear and quadratic elements give very similar results, and it is not possible to determine which one is more accurate. At the same time, it can be seen that the free mesh results of elements 185 and 285 overlap, as do the free mesh results of elements 186 and 187. This suggests that the formulation of these two pairs is identical. For a mapped mesh, the solution

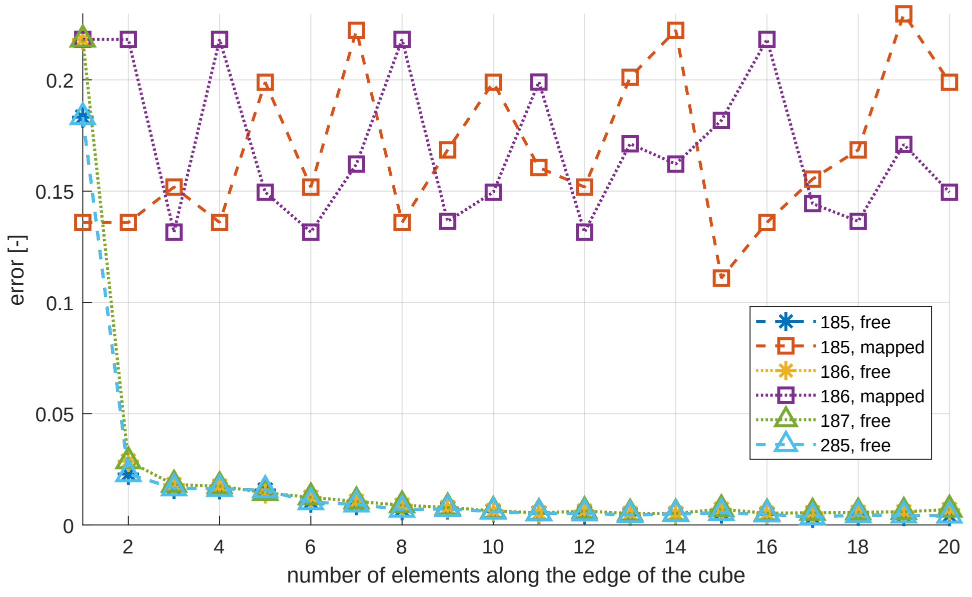
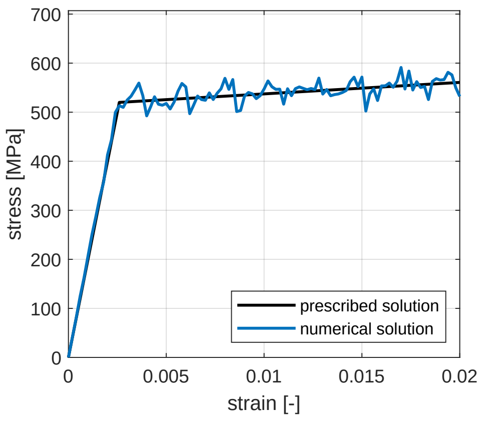


Figure 4.12: Error of the solution plotted against the mesh density for multiple elements (dashed line - linear element, dotted line - quadratic element; cube - hexahedral elements, triangle - tetrahedral elements by default, star - hexahedral elements degenerated to tetrahedral elements).

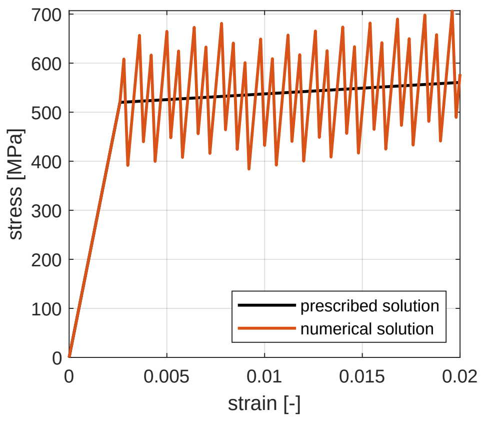
error is intuitively higher than for a free mesh. The error in the mapped mesh varies significantly with the mesh density, but it is not possible to determine whether an increasing or decreasing trend prevails. The global difference in solution errors between linear and quadratic elements is not apparent.

It is necessary to clarify that although errors in the solutions were found, in all cases, the errors are so small that every numerical simulation can be considered almost correct. The largest relative error between the calculated and prescribed stress was found in the mapped mesh with five SOLID185 elements along the edge in the 46th load-step and was only $6 \cdot 10^{-6}\,\%$, which might be just a computational error.

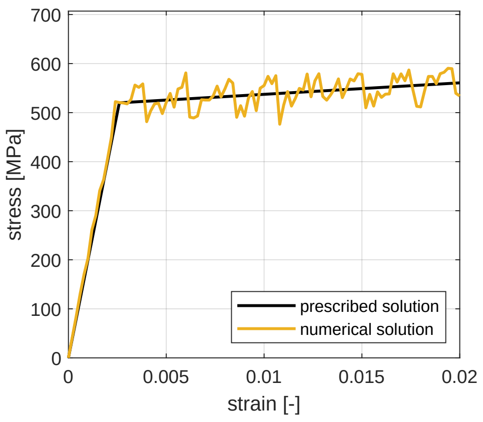
Figure 4.13 shows the deformation curves for all elements and mesh types with MD of 20. Here, it can also be seen that the graphs of the free mesh simulations for both the default and degenerated tetrahedral elements are identical.



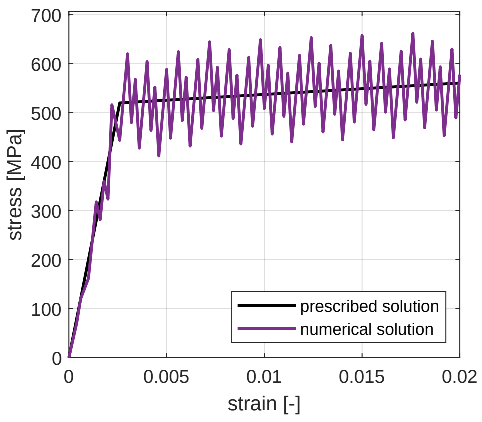
(a) : free mesh, element 185



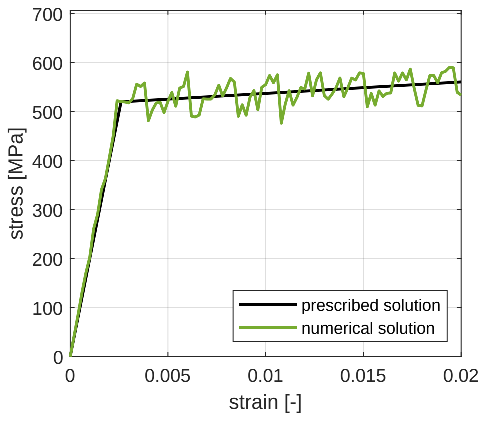
(b) : mapped mesh, element 185



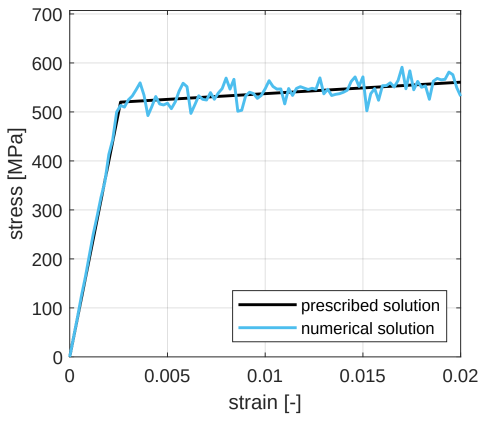
(c) : free mesh, element 186



(d) : mapped mesh, element 186



(e) : free mesh, element 187



(f) : free mesh, element 285

Figure 4.13: Comparison of prescribed and numerical deformation curves for mesh with density of 20 elements along the side of the cube for different elements and mesh types (difference between prescribed and numerical curves is scaled by a factor of $5 \cdot 10^6$ for mapped mesh and by a factor of $5 \cdot 10^7$ for free mesh).

The figure 4.14 shows the computation time dependency on the mesh density. The simulations with the mapped mesh using element 185 were the most time-saving, while the free mesh simulations with element 186 were the most time-consuming, where the difference was more than an order of magnitude. In all cases (excluding simulations lasting dozens of seconds), the simulations with linear elements were more time-efficient than the simulations with their quadratic counterparts. There can also be seen a significant difference in computation time for the free mesh between the default tetrahedron and the default hexahedron elements. The longest simulation lasted 44 minutes.

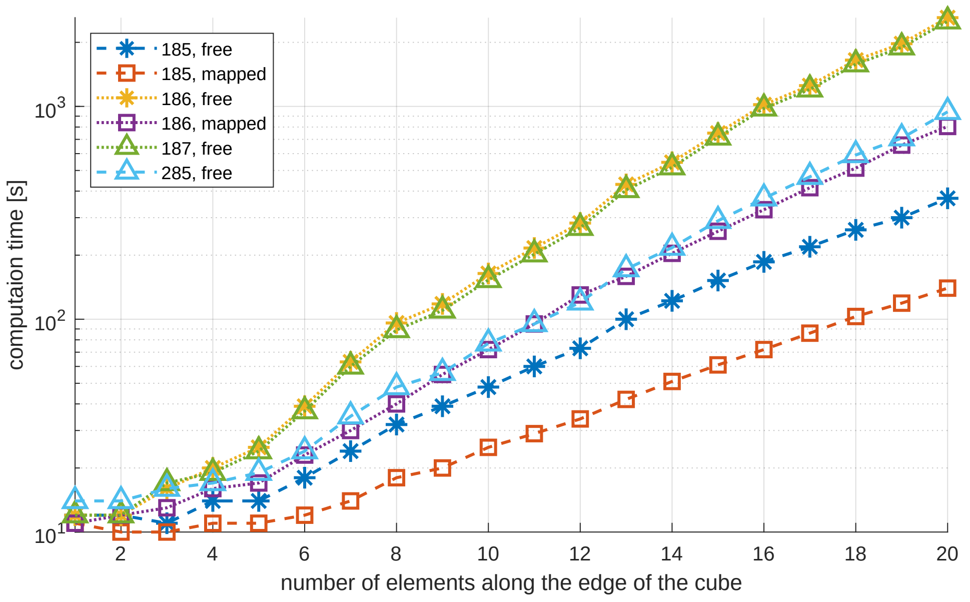


Figure 4.14: Computation time plotted against number of elements along the side of the cube

During the simulations, the reaction forces in the side supports, which prevent rigid body motion, were also checked. These were nonzero in several randomly selected simulations but insignificant in comparison with the reaction forces in the direction of deformation.

4.1.5 Conclusion of the subchapter

It was found that in simulations of uniaxial cube deformation, the choice between a linear or quadratic element has no significant effect on the obtained results. It has also been shown that hexagonal elements degenerated into tetrahedrons give the same results as their tetrahedral equivalents. The mesh density affects the results, but only regarding the free mesh, where the accuracy increases with rising mesh density. In the mapped mesh simulations, the accuracy of the results does not globally change as the mesh density increases. The accuracy of the results is thus significantly better for the free mesh than for the mapped mesh. However, even the results from the mapped mesh simulations are very accurate, as the largest relative error between the calculated stress and the prescribed stress was only $6 \cdot 10^{-6}\,\%$.

Because the deformation curves from the simulations exactly corresponded to the prescribed deformation curve of the material model, it is up to the user whether he will tune the material model to the engineering deformation curve or the real deformation curve. The final deformation curve will always correspond to the defined material model, so it is only a matter of what results one wants to receive.

4.2 Auxetic structure

This subchapter is dedicated to parametric implicit simulations of quasi-static deformation of the structure presented in subsection 3.2.

4.2.1 Investigated simulation parameters

Simulations were mainly focused on the parameters presented in this section.

Model type

Two models were used in the simulations - the basic one consisting of SOLID type elements and a simplified one using the beam analogy of the structure consisting of BEAM type elements.

- **Basic solid model**
  The basic model, which is based on the computer model according to which the structure was printed, is represented by the volume of the structure meshed by solid-type elements. An example of the model for different mesh densities is in figures 4.16 and 4.17.
- **Simplified beam model**
  Because the structure is composed of a large number of regularly arranged beams, a simplified model using beam-type elements was created in addition to the basic model. The geometry will not be given by the volume of the body but only by lines representing the beams of the structure, to which the stiffness will be assigned. Thanks to such simplification, this model will contain much fewer elements than the basic model, and the computation time will be significantly reduced. At the same time, however, this model comes with a fundamental shortcoming, which is the incorrect deformation at the beam joints. While beam joints in the

basic model correspond to the real structure, the beams in the simplified model meet only at one single point, and can thus deform too freely (see figure 4.15). Due to the dimensional parameters of the structure, the ratio l'/l is equal to 0.53, i.e., only 53 % of the re-entrant beams deform in the desired unlimited way. While the remaining 47 % of the beam is already connected to the neighboring beams in the real structure (as well as in the basic model), in this simplified model, the beam can be freely deformed along its entire length. Thus, such a model can be expected to be significantly more flexible than the basic model. An example of a simplified model is shown in figure 4.18.

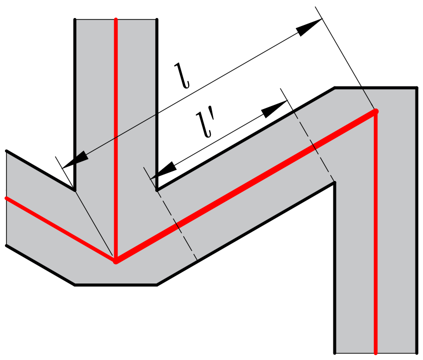


Figure 4.15: Models comparison in real proportion (grey - basic model, red - simplified model).

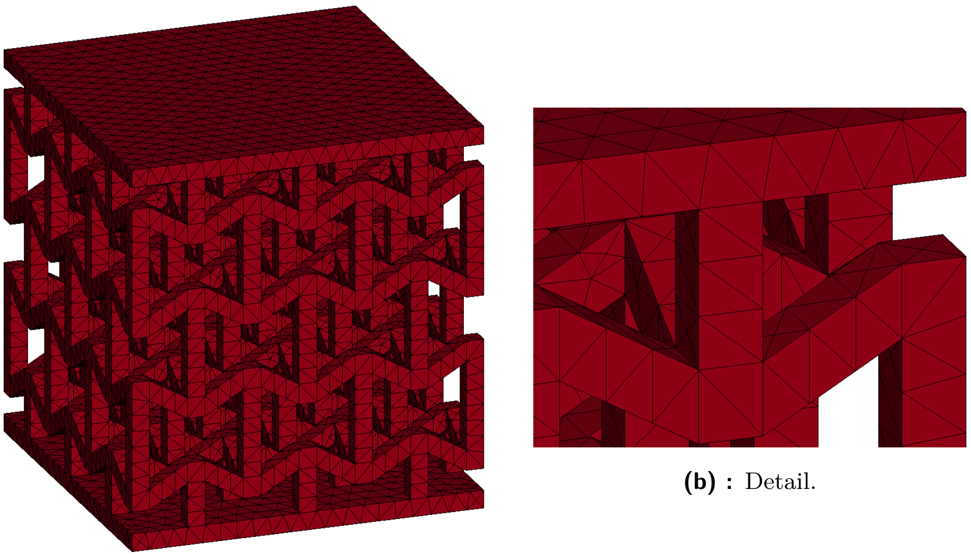
Mesh density

In the case of the basic model, the density of the mesh was controlled by the size of the element, which was chosen so that its n-multiples exactly fill the length of the cross-sectional edge of the beams. In other words, the mesh density was given by the number of elements that will be along the cross-sectional edge of the beams in the structure. Simulations were created for the number of elements along the cross-sectional edge in the range of 1-3. The model with one element along the edge of the cross-section is shown in figure 4.16, the model MD of 3 is shown in figure 4.17. Table 4.1 summarises the numbers of elements and nodes of this model for different mesh densities.

elements along the CS edge	total elements	nodes with lin. elements	nodes with quad. elements
1	39 841	13 886	80 823
2	195 255	54 947	346 425
3	553 629	134 395	903 330

Table 4.1: Number of elements and nodes in the models of the structure using SOLID elements

In the simplified model, the mesh density was controlled by the number of elements that make up each beam of the structure. Simulations were created for the number of elements on the beam in the range of 1-20. The model with mesh density 10 is shown in figure 4.18.

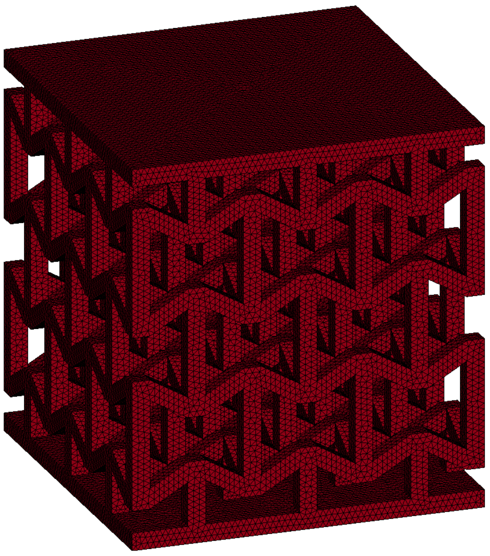

(a) : Whole structure.

(b) : Detail.

Figure 4.16: Mesh of the solid model of the structure with one element along the edge of the CS.

Elements

In the basic model, the SOLID285 and SOLID187 elements presented in the subsection 4.1.1 were varied.


(a) : Whole structure.

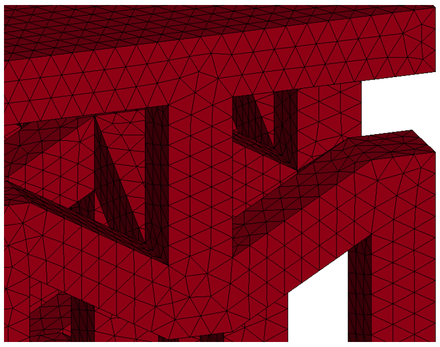
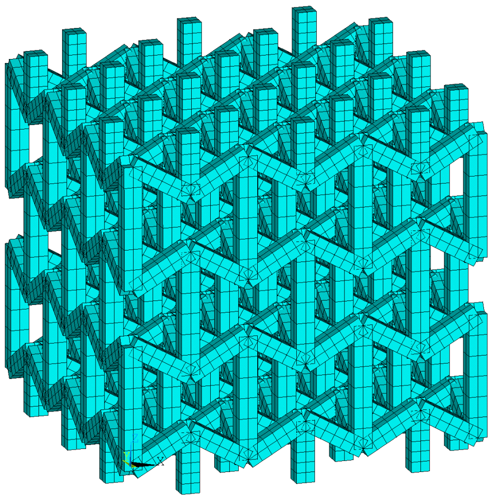

(b) : Detail.

Figure 4.17: Mesh of the solid model of the structure with three elements along the edge of the CS.


(a) : Whole structure.

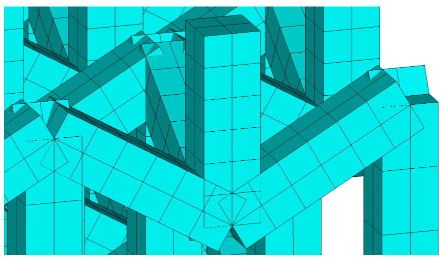
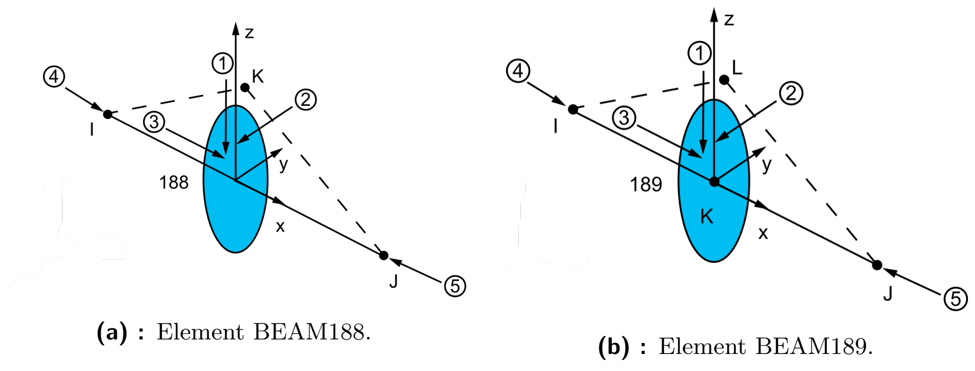

(b) : Detail.

Figure 4.18: Mesh of the beam model of the structure with five elements along the beams.

The linear element BEAM188 and the quadratic element BEAM189 were varied in the beam model. Their schemes are shown in figure 4.19. Nodes of these elements have 6 DoFs - translations in the direction of x, y, and z axes and rotations around these axes.

Both elements are based on Tymoshenko's beam theory, which includes the effects of shear stresses presented in the beam, and are suitable for large rotations and deformations in nonlinear applications [29].



(a) : Element BEAM188.

(b) : Element BEAM189.

Figure 4.19: Schemes of BEAM type elements used in simulations with the structure [29].

4.2.2 Principles of simulations

The script *APDL_structure_deformation_solid* controlling the simulation of the basic structure model is in many ways identical to the script *deformation_cube_APDL* presented in the subsection 4.1.2. The difference is mainly in the creation of the geometry of the structure, which does not need to be modeled, and which is only imported using command *IGESIN* in the *.iges* format. The mesh density is then controlled by defining the size of the element with the *ESIZE* command. The most distinct attribute is reading the results. While the structure is supported and deformed in the same way as the cube, the displacement is not read from the displacement of the top plate nodes but is read from the nodes closest to the middle of the joint faces shown in figure 4.20. The displacement was read from these 12 marked joints in two perpendicular planes, and the deformation was taken as the average of the approaches of these six pairs of nodes. In addition, the standard deviation from these values was calculated, which indicates the measure of the asymmetry of the deformation. The value 8.106 mm was taken as the initial length l_0. In this way, the greatest possible correspondence with the performed experiment is achieved.

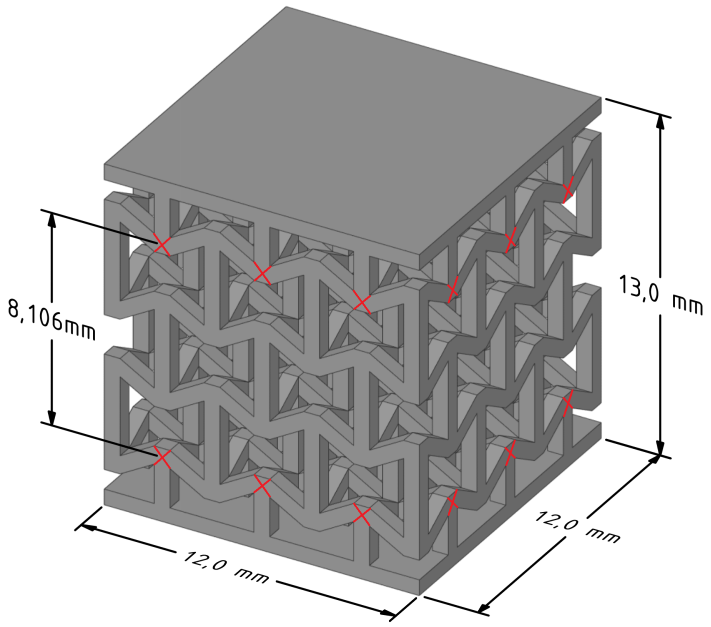


Figure 4.20: Points from which the displacements were read.

The main characteristic of the *APDL_structure_deformation_beam* script controlling the simulation of the simplified beam model is defining the model geometry. The geometry is created thanks to the repeated cellular structure of the cube and its central and axial symmetry by using several nested cycles. The support of the structure was modeled in accordance with the real structure, where the upper and lower beams are clamped to the upper and lower plates. Because these plates are assumed to deform only slightly compared to the rest of the structure, all DoFs are removed from all upper and lower nodes in simulations with this simplified model. The only exceptions are the upper nodes of the structure, for which a forced displacement is defined instead of removing the translation DoF in the direction of the z-axis. The displacements of this simplified model were read in the same beam joints as in the basic model. Instead of searching for the nearest node, however, thanks to the deterministic mesh, displacements were always calculated from the same nodes exactly in the beam joints. Although the distance between the top and bottom displacement-reading nodes differs slightly from the basic model, the initial length of l_0 was chosen identically to the basic model to ensure comparability of the results.

4.2.3 Automation

The whole process of automation of simulations with structure is based on the automation of simulations with a cube presented in subsection 4.1.3. Nevertheless, there are a few differences. An overview of the automation process is in figure 4.21.

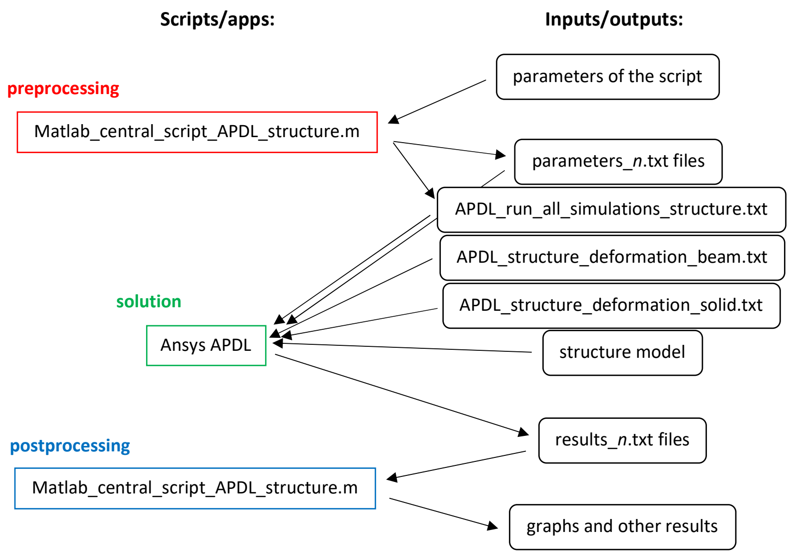

Scripts/apps:
Inputs/outputs:
preprocessing
Matlab_central_script_APDL_structure.m
parameters of the script
parameters_*n*.txt files
APDL_run_all_simulations_structure.txt
APDL_structure_deformation_beam.txt
APDL_structure_deformation_solid.txt
solution
Ansys APDL
structure model
results_*n*.txt files
postprocessing
Matlab_central_script_APDL_structure.m
graphs and other results

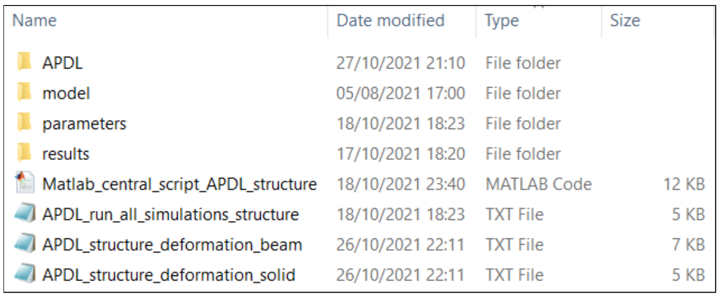
Figure 4.21: Block diagram of the simulations performation process in APDL with the structure.

Prerequisites

The following folders and files must be stored in the same directory for proper functioning (see figure 4.22):

- folder *APDL* - working directory for APDL
- folder *model* - model of the structure is saved in *.iges* format here
- folder *parameters* - parameters of the individual simulations will be generated here
- folder *results* - results of the individual simulations will be saved here

- file *Matlab_central_script_APDL_structure.m* - script which is the foundation of the automation process
- file *APDL_run_all_simulations_structure.txt* - script, which runs all generated simulations
- file *APDL_structure_deformation_beam.txt* - script for the simulation of the deformation of the beam-modelled structure
- file *APDL_structure_deformation_solid.txt* - script for the simulation of the deformation of the solid-modelled structure



Name	Date modified	Type	Size
APDL	27/10/2021 21:10	File folder	
model	05/08/2021 17:00	File folder	
parameters	18/10/2021 18:23	File folder	
results	17/10/2021 18:20	File folder	
Matlab_central_script_APDL_structure	18/10/2021 23:40	MATLAB Code	12 KB
APDL_run_all_simulations_structure	18/10/2021 18:23	TXT File	5 KB
APDL_structure_deformation_beam	26/10/2021 22:11	TXT File	7 KB
APDL_structure_deformation_solid	26/10/2021 22:11	TXT File	5 KB

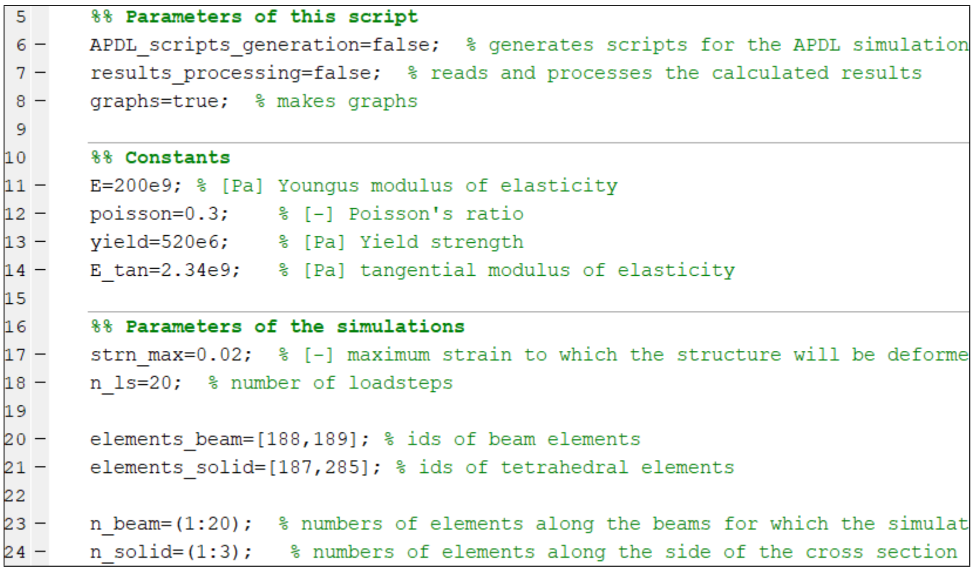
Figure 4.22: Necessary items in the directory.

Conduction of parametric simulations

Very similarly to the simulations with the cube, the generation of simulation parameters was done using the script *Matlab_central_script_APDL_structure.m* in the Matlab environment. An illustration of the script set to generate parameter files for APDL with the parameters that were used in the first simulations set (subsection 4.2.4) is shown in figure 4.23.

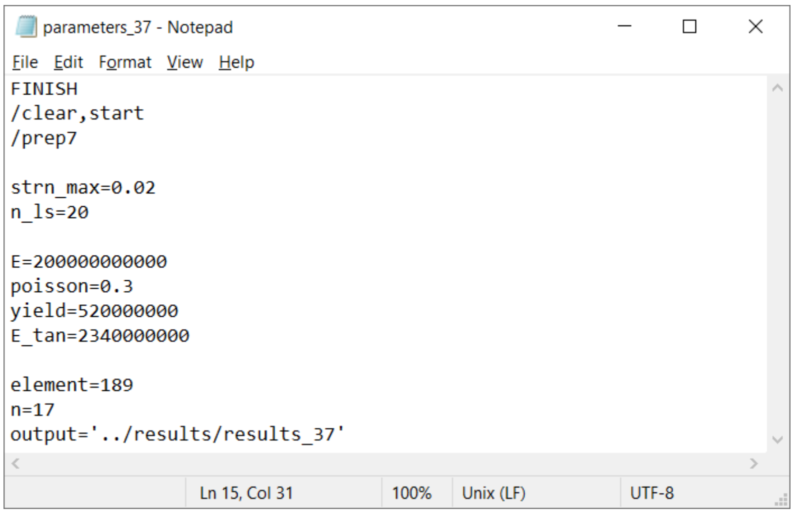
Parameter files generated prior to computing the simulations are very similar to those in simulations with the cube. An example of one parameter file is in figure 4.24.

Also, the script *APDL_run_all_simulations_structure.txt* that runs all simulations and which is created while generating parameter files is very similar to its cube counterpart. Illustration is shown in figure 4.25.



```
%% Parameters of this script
APDL_scripts_generation=false;  % generates scripts for the APDL simulation
results_processing=false;  % reads and processes the calculated results
graphs=true;  % makes graphs

%% Constants
E=200e9; % [Pa] Youngus modulus of elasticity
poisson=0.3;    % [-] Poisson's ratio
yield=520e6;    % [Pa] Yield strength
E_tan=2.34e9;   % [Pa] tangential modulus of elasticity

%% Parameters of the simulations
strn_max=0.02;  % [-] maximum strain to which the structure will be deforme
n_ls=20;  % number of loadsteps

elements_beam=[188,189]; % ids of beam elements
elements_solid=[187,285]; % ids of tetrahedral elements

n_beam=(1:20);  % numbers of elements along the beams for which the simulat
n_solid=(1:3);   % numbers of elements along the side of the cross section
```

Figure 4.23: Illustration of the defined constants and parameters in Matlab.

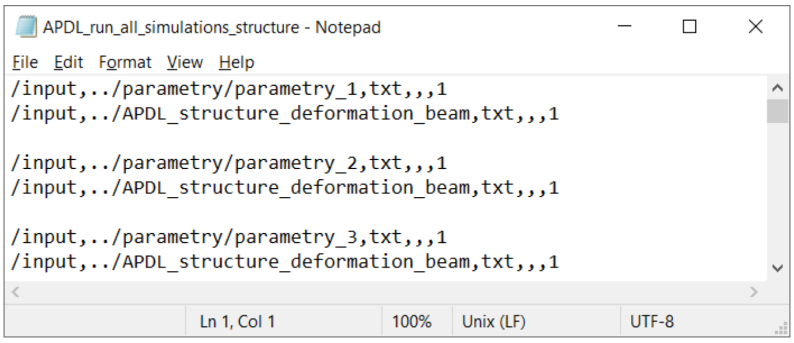


```
FINISH
/clear,start
/prep7

strn_max=0.02
n_ls=20

E=200000000000
poisson=0.3
yield=520000000
E_tan=2340000000

element=189
n=17
output='../results/results_37'
```

Figure 4.24: Illustration of a *.txt* file with parameters.

The result files look almost the same. Only one extra column is added, which contains the standard deviation of the strain. An example is in figure 4.26.

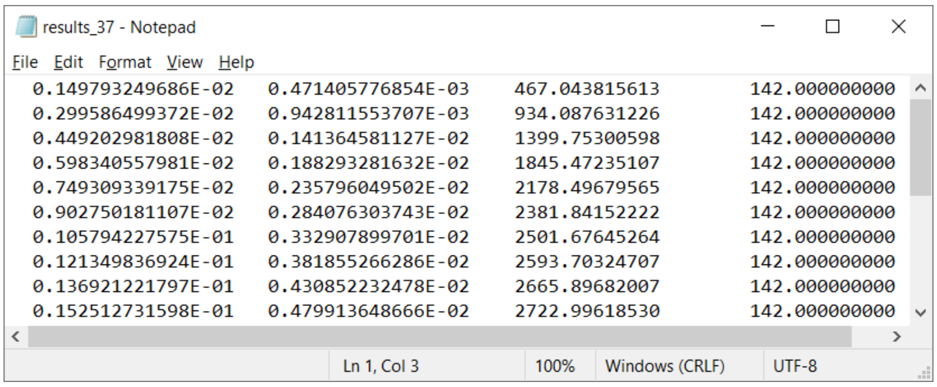
After completing the computations of all simulations, the result files can also be automatically loaded into one complete table by running the script *Matlab_central_script_APDL_cube.m* set for processing the results again. An example of the sorted data can be seen in the picture 4.27.



APDL_run_all_simulations_structure - Notepad

```
/input,../parametry/parametry_1,txt,,,1
/input,../APDL_structure_deformation_beam,txt,,,1

/input,../parametry/parametry_2,txt,,,1
/input,../APDL_structure_deformation_beam,txt,,,1

/input,../parametry/parametry_3,txt,,,1
/input,../APDL_structure_deformation_beam,txt,,,1
```

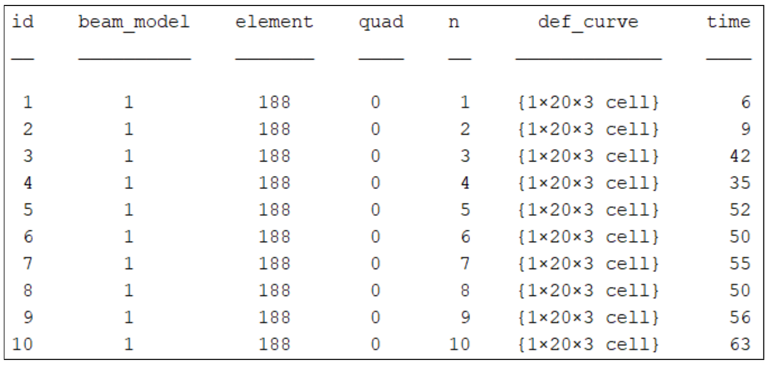
Figure 4.25: Illustration of the APDL script which runs all the simulations.



results_37 - Notepad

```
  0.149793249686E-02  0.471405776854E-03  467.043815613      142.000000000
  0.299586499372E-02  0.942811553707E-03  934.087631226      142.000000000
  0.449202981808E-02  0.141364581127E-02  1399.75300598      142.000000000
  0.598340557981E-02  0.188293281632E-02  1845.47235107      142.000000000
  0.749309339175E-02  0.235796049502E-02  2178.49679565      142.000000000
  0.902750181107E-02  0.284076303743E-02  2381.84152222      142.000000000
  0.105794227575E-01  0.332907899701E-02  2501.67645264      142.000000000
  0.121349836924E-01  0.381855266286E-02  2593.70324707      142.000000000
  0.136921221797E-01  0.430852232478E-02  2665.89682007      142.000000000
  0.152512731598E-01  0.479913648666E-02  2722.99618530      142.000000000
```

Figure 4.26: Illustration of a *.txt* file with results (left to right: strain [-], reaction force [N], standard deviation [N], computation time [s]).



id	beam_model	element	quad	n	def_curve	time
1	1	188	0	1	{1×20×3 cell}	6
2	1	188	0	2	{1×20×3 cell}	9
3	1	188	0	3	{1×20×3 cell}	42
4	1	188	0	4	{1×20×3 cell}	35
5	1	188	0	5	{1×20×3 cell}	52
6	1	188	0	6	{1×20×3 cell}	50
7	1	188	0	7	{1×20×3 cell}	55
8	1	188	0	8	{1×20×3 cell}	50
9	1	188	0	9	{1×20×3 cell}	56
10	1	188	0	10	{1×20×3 cell}	63

Figure 4.27: Illustration of the database of the parameters and results.

4.2.4 Simulations set 1

This set of simulations is focused on the comparison of the two structure models as well as the contribution of mesh density and elements to the results.

Used parameters

A total of 46 simulations were performed. Six simulations used the basic model for both solid elements and all three mesh densities, 40 simulations used the simplified model for both beam elements and all 20 mesh densities. Used parameters are summarised in figure 4.28.

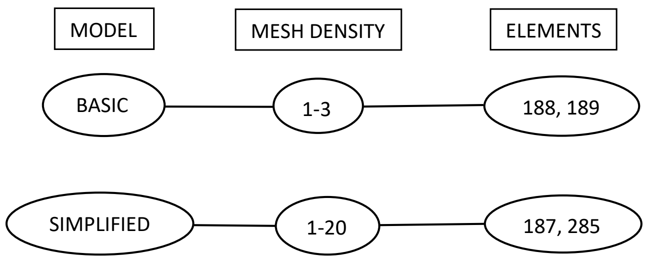


Figure 4.28: Scheme of used parameters for simulations set 1 with the structure in APDL.

The structure was also deformed to a maximum overall relative deformation of 2 %, but at 20 LS. This maximum deformation was chosen due to the poor convergence of the most simplified models close to deformation of 3 %. 20 LS were chosen to reduce the computation time.

Results

The graph in the figure 4.29 shows the deformation curves of the selected simulations. Indeed, the assumption that the simplified structure model will be significantly more flexible than the basic structure model has been confirmed.

The graphs in figure 4.30 show the convergence of the solution with increasing mesh density. While in the simplified model, the convergence is relatively fast and evident, in the basic model, there is not enough data to perform a deeper analysis. In both cases, however, models with linear elements are stiffer than models with quadratic elements. Regardless of the choice of an element, the results should converge to the same values, which explains why the convergence curve is steeper for linear elements.

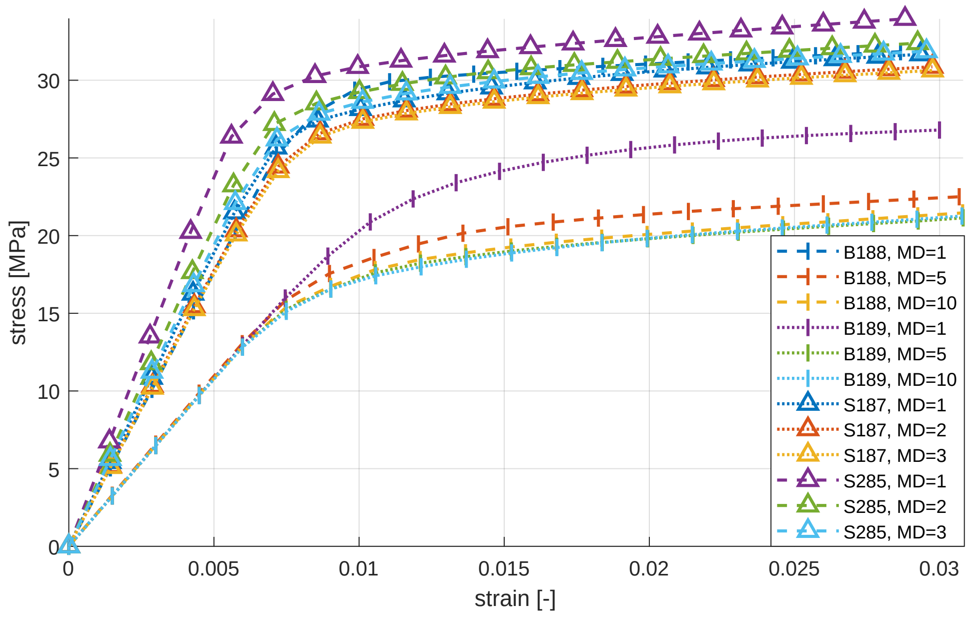
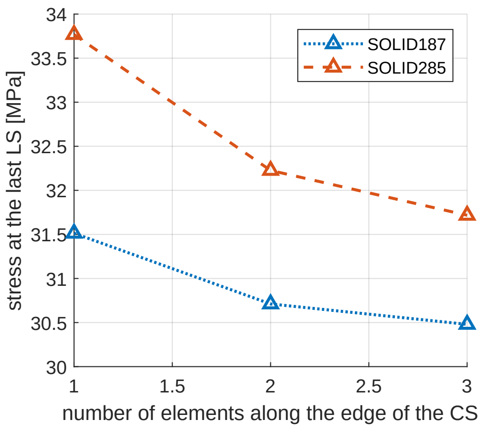
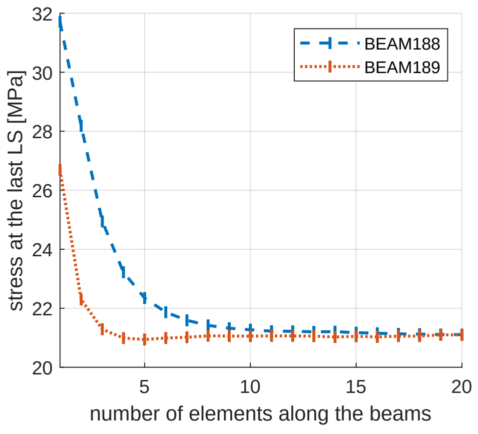


Figure 4.29: Deformation curves of selected simulations.



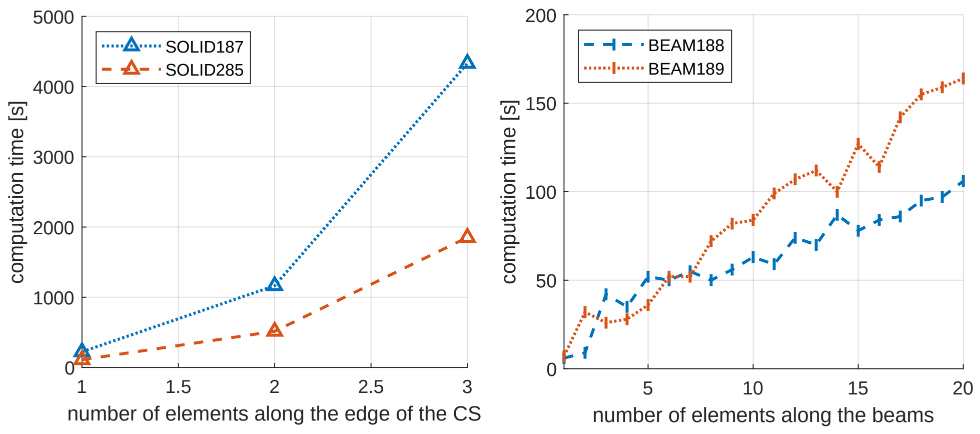
(a) : basic SOLID model



(b) : simplified BEAM model

Figure 4.30: Stress at the last LS plotted against the mesh density for different elements and models.

The graphs in figure 4.31 show the computation time of the simulations. The simplified model shows significantly lower time requirements compared to the basic model. The time of the longest simplified-model simulation is comparable to the fastest simulation with the basic model. The longest simulation took 72 minutes.



(a) : basic SOLID model **(b) :** simplified BEAM model

Figure 4.31: Computation time plotted against the mesh density for different elements and models.

4.2.5 Simulations set 2

This set of simulations is focused on the comparison between experimental and numerical results.

Used parameters

The most accurate models were the basic model with three quadratic elements SOLID187 along the edge of the CS and the simplified model with 10 quadratic elements BEAM189 along the beams. Further simulations were performed for these models, this time up to a maximum total deformation of 18 % at 100 loading steps. The total deformation of 18 % was chosen because it is approximately the largest deformation without any contact among the beams in the structure. With a larger deformation, it would be necessary to define the interactions during contact among the elements to maintain accuracy, which would significantly complicate and prolong the simulations procedure. 100 LS was chosen to obtain more detailed deformation curves.

Results

Due to problems with the convergence of the solution, the simulation with the simplified model was eventually not performed. It was tested to change the length of the load-steps and mesh density and to switch between linear and quadratic element. Even with the solution predictor turned off and the convergence tolerance reduced, it was not possible to get past a total deformation of around 3 %.

A comparison of the deformation curve from the experiment and the deformation curve obtained by simulation with the basic model with three SOLID187 elements along the edge of the CS is shown in the figure 4.32. The maximally deformed structure is then shown in figure 4.33. Both curves are fairly close in shape (not in the stress values) to about 3 % of deformation, then they misalign. The calculation of this simulation took 7 hours and 49 minutes.

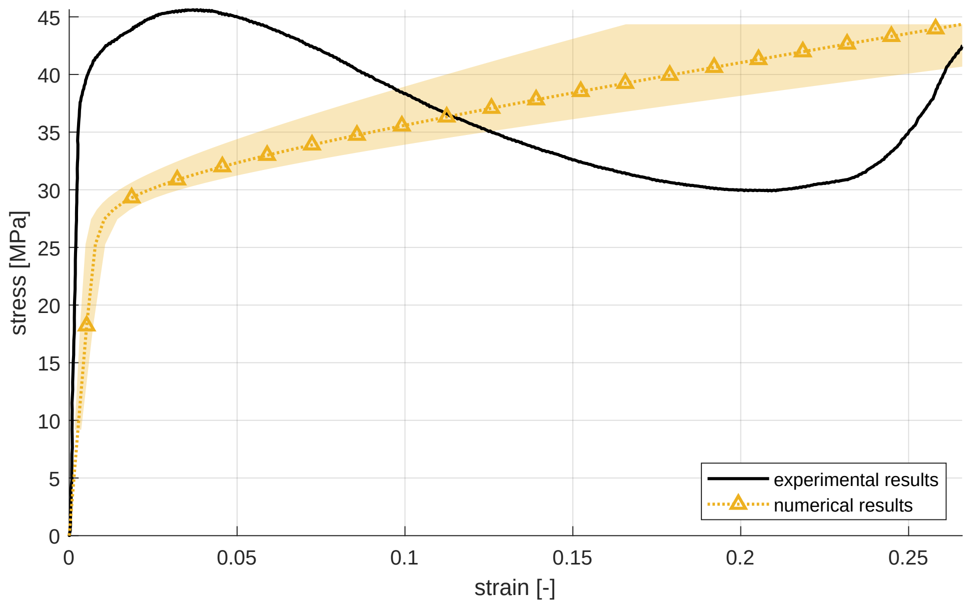


Figure 4.32: Comparison of the stress-strain curves from experiment and from the simulation with the basic model with three SOLID187 elements along the edge of the CS.

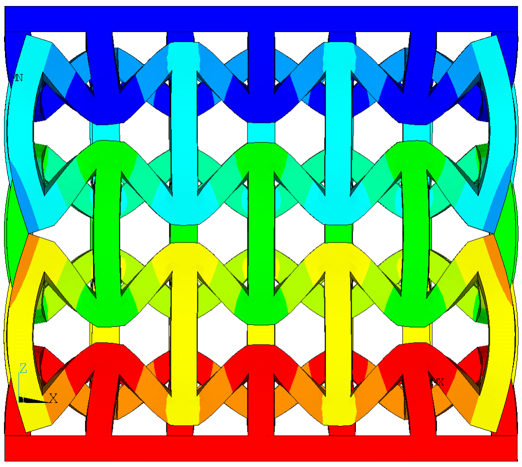


Figure 4.33: Deformed structure at a total deformation of 18 %.

4.2.6 Conclusion of the subchapter

Two structure models were introduced and validated against each other. It was discovered that deeper analysis using the simplified beam model can not be performed for two reasons. Firstly, the simplified model shows poor convergence at larger prescribed compressive strain, and secondly, it is too flexible to compare well with the experimental data. Even the selected solid element model, which should give the most accurate results, does not correspond very well with the results from the experiment. A more detailed analysis of this difference is discussed in subchapter 6.2.

Chapter 5

Explicit numerical simulations in LS-DYNA

As in the previous chapter, the goal here is to perform a large number of simulations with different input parameters using the LS-DYNA program. The quasi-static problem will be converted into a dynamic problem with the search for the strain-rate, in which the results still correspond to the quasi-static problem, and at the same time, the simulations require acceptable computation time. While APDL is a programming language that allows code algorithmization, LS-DYNA scripts can not be algorithmized in any way. The entire automation process is thus again transferred to Matlab.

5.1 Auxiliary scripts and functions

Several functions and scripts have been written to automate simulations in LS-DYNA. Although it would be possible to write the whole automation process in only one script for each specimen, such code would very likely be relatively complex and confusing. Therefore, the automation process was divided into several sub-scripts, which slightly prolong the experimenter's work in fulfilling the tasks of this work, however, they provide a greater overview of the ongoing processes. Also, thanks to their versatility, some scripts and functions might be utilized beyond this thesis. Two scripts and two functions, which were necessary for this work, but might have a broader application, are presented in this subchapter.

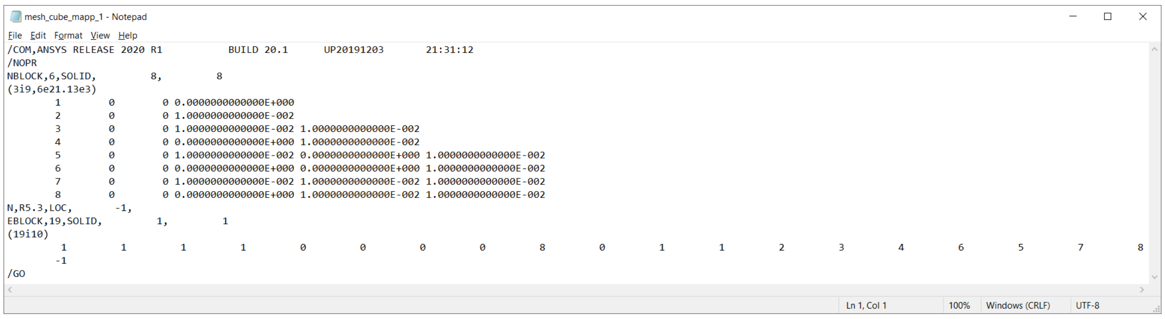
5.1.1 Script for mesh conversion from *.cdb* to *.k* format

The process of mesh creation in LS-DYNA works significantly differently from APDL. While in APDL, the creation of the mesh is controlled by several commands directly in the source script, in the script for LS-DYNA, it is necessary to have already defined a list of all nodes with their coordinates and a list of all elements with their corresponding nodes. Mesh can be created in preprocessing applications, such as LS PrePost, but it is still a complex algorithmic or manual operation. In order to automate the mesh creation process, the opportunity of exporting the mesh from APDL with the command *CDWRITE* in the format *.cdb* was used. An example of a mesh in such a format is shown in figure 5.1a. However, the mesh inserted in the source script for LS-DYNA must have a different format - *.k*. An example of the same mesh in such a format is shown in figure 5.1b.

Therefore, a script called *cdb2k.m* was written in which only two directories have to be specified. First directory containing one or more meshes in *.cdb* format and the second one into which the meshes should be reformated. The script then searches the files stored in the input directory, selects those with the suffix *.cdb* and automatically converts them one by one with their respective names to the output directory, this time with the suffix *.k*. The use of the script is limited to tetrahedral and hexahedral linear elements only.

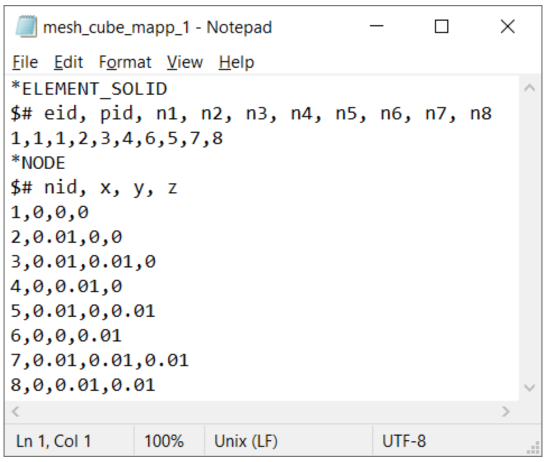
5.1.2 Functions for substituting variables in a text file

Because it is not convenient to use variables for numeric or text values in the LS-DYNA scripts by **PARAMETER* command, one universal header was created for each specimen, in which the variable names are marked with an asterisk on both sides (for example **sample_variable**). Their value is then controlled by another script in Matlab using a function named



```
/COM,ANSYS RELEASE 2020 R1          BUILD 20.1      UP20191203       21:31:12
/NOPR
NBLOCK,6,SOLID,         8,         8
(3i9,6e21.13e3)
        1        0        0 0.0000000000000E+000
        2        0        0 1.0000000000000E-002
        3        0        0 1.0000000000000E-002 1.0000000000000E-002
        4        0        0 0.0000000000000E+000 1.0000000000000E-002
        5        0        0 1.0000000000000E-002 0.0000000000000E+000 1.0000000000000E-002
        6        0        0 0.0000000000000E+000 0.0000000000000E+000 1.0000000000000E-002
        7        0        0 1.0000000000000E-002 1.0000000000000E-002 1.0000000000000E-002
        8        0        0 0.0000000000000E+000 1.0000000000000E-002 1.0000000000000E-002
N,R5.3,LOC,       -1,
EBLOCK,19,SOLID,         1,         1
(19i10)
         1         1         1         1         0         0         0         0         8         0         1         1         2         3         4         6         5         7         8
        -1
/GO
```

(a) : Format *.cdb*.



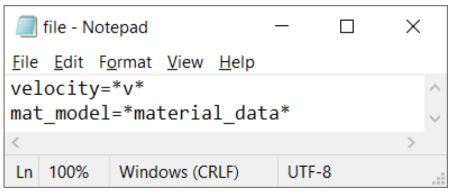
```
*ELEMENT_SOLID
$# eid, pid, n1, n2, n3, n4, n5, n6, n7, n8
1,1,1,2,3,4,6,5,7,8
*NODE
$# nid, x, y, z
1,0,0,0
2,0.01,0,0
3,0.01,0.01,0
4,0,0.01,0
5,0.01,0,0.01
6,0,0,0.01
7,0.01,0.01,0.01
8,0,0.01,0.01
```

(b) : Format *.k*.

Figure 5.1: Examples of writing a mesh of the cube formed by one hexagonal element in two different formats.

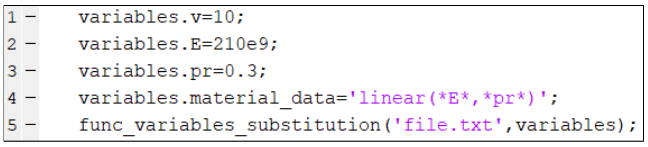
func_variables_substitution.m, which finds the variables marked in this way and replaces their names with the specified numeric or text values.

The input to this function is the path to the text file, in which the variables are to be replaced, and the structure (a form of data type in Matlab) that contains numeric and text variables. The function searches in the text file for the variable names corresponding to the variable names in the structure and replaces them with their values (numeric values or other texts). Since the function in the text file first replaces the text variables and then the numeric variables, it is possible to insert text with the names of other numerical variables, which the user wants to substitute in the script. An example of how this function works is shown in the figure 5.2.



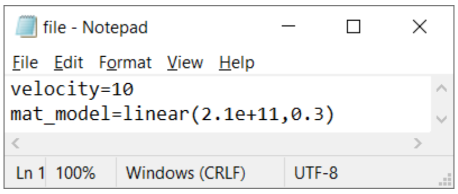
```
velocity=*v*
mat_model=*material_data*
```

(a) : Text file before running the function.



```
variables.v=10;
variables.E=210e9;
variables.pr=0.3;
variables.material_data='linear(*E*,*pr*)';
func_variables_substitution('file.txt',variables);
```

(b) : Script pro substituting the variables.



```
velocity=10
mat_model=linear(2.1e+11,0.3)
```

(c) : Text file after running the function.

Figure 5.2: Illustration of using the function for substituting the variables.

For operation with LS-DYNA, the format for inserting numeric variables was set in such a way, that the maximum number of valid digits was maintained, and at the same time, the requirements for the maximum length of numbers were met.

This relatively simple but very effective function has become the basis of the process of automating the generation of *.k* scripts for parametric simulations. To create any number of scripts, it was only necessary to write one header with generally defined variables, which was then copied and modified multiple times using this function.

5.1.3 Script for automated k-script solving

Very often, one has more *.k* scripts ready and would like to have them all solved. Because it would be necessary to manually run the solution of each of them individually using the LS-Run application, a script called *LSD_solver.m* was written, which performs this work automatically.

The prerequisite for its operation is that each k-script is stored in its own folder, and all these folders are located in a single common directory. In the script, the path to this directory and the path to the solver have to be defined. The script then automatically detects the names of the folders in the directory and solves the individual *.k* scripts one by one.

Because the base license does not allow the use of multiple processor threads, this script runs *.k* scripts on only a single thread at a time. Automated parallel solution of several *.k* scripts at once would be possible only with advanced programming skills. However, in addition to the running solving script, it is possible to manually run the solution of other *.k* scripts in another application (for example, in LS-Run) up to the number that the license allows. The solving script recognizes that the given *.k* script has already been solved (or is currently being solved) and skips its solution. In addition, due to the nature of this work, the time required to solve individual *.k* scripts varied significantly. To solve dozens or hundreds of *.k* scripts, it was usually needed to run only some of the most time-consuming ones manually via LS-Run, while in the meantime, the rest was automatically solved by the solving script.

5.1.4 Function for calculating the area between two curves

The area between the calculated and the prescribed deformation curve was used as the measure of accuracy in this work. Such metric was already used in the implicit simulations, but in that case, all deformation curves were defined at the same values of deformation, and the calculation of the area was relatively straightforward. To determine the area between two general multilinear curves, the function *func_area_between_curves.m* was written. The inputs are the vectors x and y of one curve, the vectors x and y of the second curve, and the limits of x values for which the area is to be calculated. The function returns the exact area (only limited by numerical precision) between the curves, regardless of whether the vectors of x values will be defined at different points or how many intersections the curves have.

5.2 Cube

This subchapter is dedicated to parametric explicit simulations of quasi-static deformation of the homogeneous cube presented in the subchapter 3.1.

5.2.1 Investigated simulation parameters

Simulations were mainly focused on the parameters presented in this section.

Deformation speed mode

Two deformation speed modes were used. Constant one with constant deformation speed during the whole simulation, which is characterized by a sudden introduction of a deformation speed at the beginning of the simulation. This leads to some initial oscillations of the specimen's body at higher strain-rates and therefore distorting the initial linear part of the stress-strain curve. To mitigate this problem, a linearly increasing deformation speed mode was introduced. Such simulations begin with the specimen at rest, and the deformation speed linearly increases to a maximum value of double the equivalent constant speed. This mode can be used either to get more precise results or to use higher strain-rates with similar results. Both modes are compared in figure 5.3.

The strain-rate corresponding to the constant deformation speed mode refers to the actual strain-rate during the simulation, while the strain-rate corresponding to the linearly increasing deformation speed mode refers to the average strain-rate during the simulation.

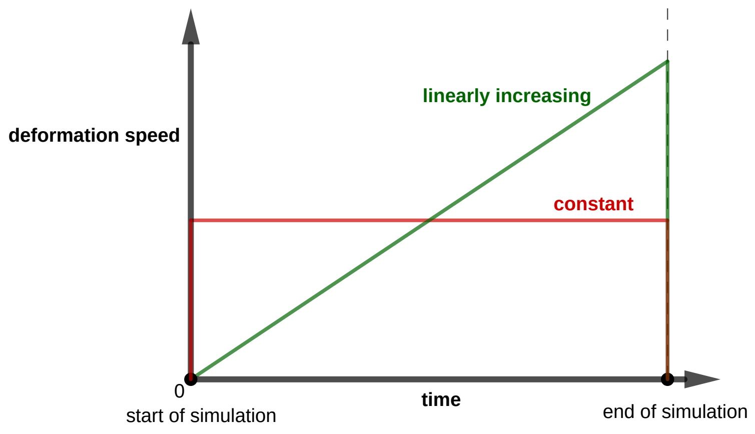


Figure 5.3: Deformation speed modes.

Mesh density

The density of the mesh given by the number of elements along the edge of the cube was varied in range of 2-20. One element along the edge was not chosen due to very poor results or negative volume error messages, which terminated the simulations.

Mesh type

In the simulations, mapped and the free meshes were varied. Because a script was produced for transferring mesh from APDL to LS-DYNA, the mesh used in these simulations is identical to the mesh in the previous chapter for better comparability of the results.

Elements

A total of 6 element formulations were used - 3 hexagonal (ELFORM 1, 2, and 3) for the mapped mesh and 3 tetrahedral (ELFORM 10, 16, and 17) for the free mesh:

- ELFORM 1 - constant stress hexahedron
- ELFORM 2 - fully integrated S/R hexahedron
- ELFORM 3 - fully integrated quadratic 8-noded hexahedron with nodal rotations
- ELFORM 10 - 1 point constant stress tetrahedron
- ELFORM 16 - fully integrated 5-point 10-noded tetrahedron
- ELFORM 17 - fully integrated 5-point 10-noded tetrahedron [27, 30, 31]

Material models

2 material models were used - MAT_PLASTIC_KINEMATIC (MAT_003) and MAT_ISOTROPIC_ELASTIC_PLASTIC (MAT_012):

- MAT_003 - Model suitable for the definition of isotropic or kinematic hardening with the possibility of including strain-rate effects. The model was defined without the inclusion of strain-rate effects and with isotropic hardening. The input parameters are Young's modulus, Poisson's ratio, yield strength, tangential modulus, and density.
- MAT_012 - Very economical model with isotropic hardening without the possibility of including strain-rate effects. The input parameters are shear modulus, bulk modulus, yield strength, tangential modulus, and density [30].

strain-rate

Different strain-rates were considered in the simulations to find the optimal one with a quasi-static stress-strain curve and short computation time. Used strain-rates were selected as geometric series with a common ratio of 2.

5.2.2 Principles of simulations

The following simulations are based on the implicit simulations described in the subchapter 4.1 and are configured to be as similar as possible. The material model parameters stay the same, the mesh is exactly the same, and the cube base is also supported in the same way.

In these simulations, however, the reaction forces are not only read from the upper wall but from the lower one as well. Using **DATABASE_BNDOUT* command the upper wall reaction forces are saved in *bndout* file and using **DATABASE_NODFOR* command the lower wall reaction forces are saved in *nodfor* file. This way, it is possible to investigate the inertial forces in the specimen based on the difference of these two reactions.

While for implicit simulations, a forced displacement was defined in each load-step, in explicit simulations, a continuous speed of the nodes in the upper wall of the cube is prescribed. Time-step is then defined for which the specified results are written to the output files. The **CONTROL_HOURGLASS* command prevents hourglass modes. Hourglass control type 6 was chosen, as it should be effective for honeycomb structures, and therefore well suited for future use [32]. The commands **DATABASE_GLSTAT* and **DATABASE_MATSUM* both create output files in which it is possible to check the energy balance of the model. Computation time is read from *lsrun.out.txt* output file.

5.2.3 Automation

The automation process combines several different scripts that need to be run manually in an appropriate order. An overview of the automation process is shown in figure 5.4.

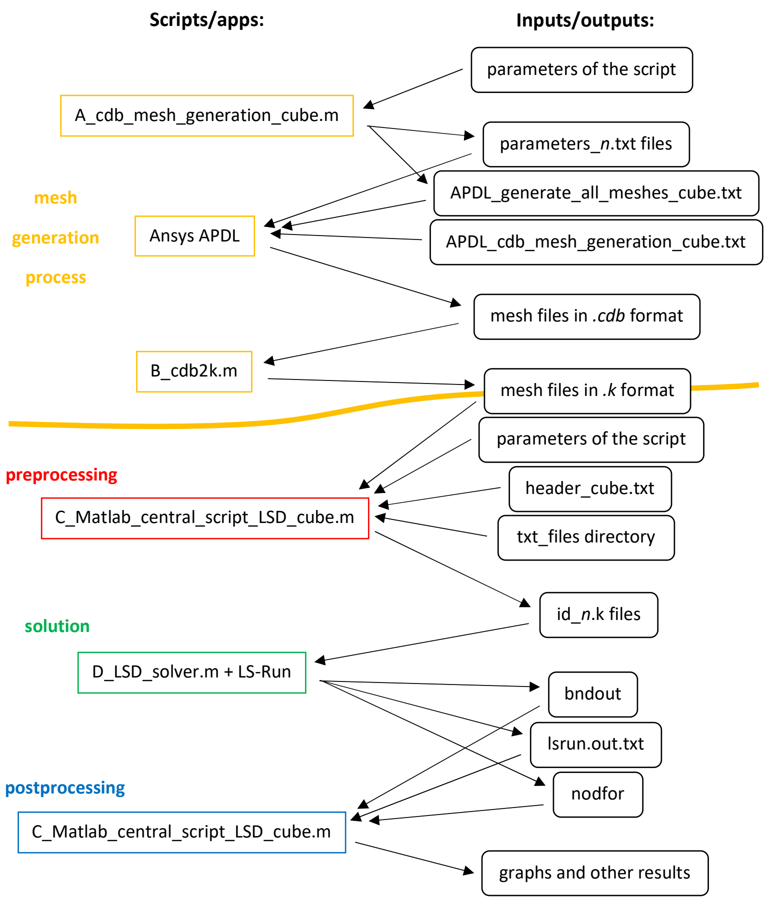


Figure 5.4: Block diagram of the simulations performation process in LS-DYNA with the cube.

Prerequisites

The following folders and files must be stored in the same directory for proper functioning (see figure 5.5):

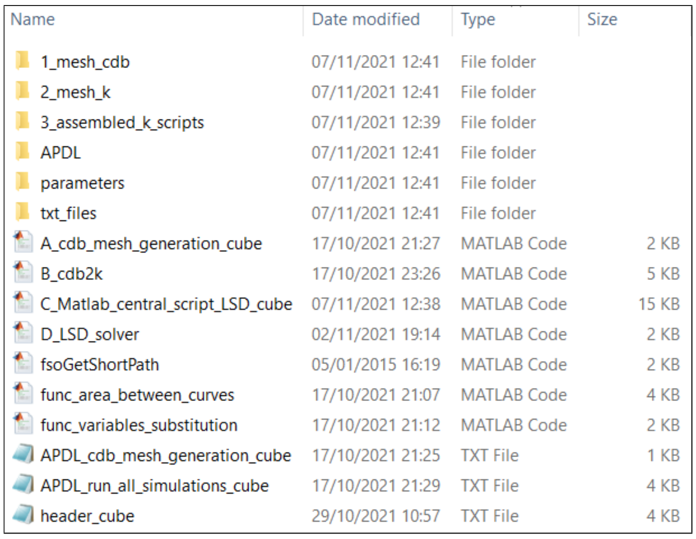
- folder *1_mesh_cdb* - mesh in format *.cdb* will be saved here
- folder *2_mesh_k* - mesh in format *.k* will be saved here
- folder *3_assembled_k_scripts* - assembled *.k* scripts will be saved here
- folder *APDL* - working directory for APDL
- folder *parameters* - parameters of APDL simulations, which generate mesh, will be saved here

- folder *txt_files* - auxiliary *.txt* files for assembling *.k* scripts are stored here
- file *A_cdb_mesh_generation_cube.m* - script which controls the mesh generation in APDL
- file *B_cdb2k.m* - script for mesh conversion from *.cdb* to *.k* format
- file *C_Matlab_central_script_LSD_cube.m* - script which is the foundation of the automation process
- file *D_LSD_solver.m* - script, which solves the assembled *.k* scripts
- file *fsoGetShortPath.m* - downloaded function, which converts path according to Windows conventions [33]
- file *func_area_between_curves.m* - function, which calculates area between two curves
- file *func_variables_substitution.m* - function which substitutes variables in *.k* scripts
- file *APDL_cdb_mesh_generation_cube.txt* - script for mesh generation
- file *APDL_generate_all_meshes_cube.txt* - script, which runs all created APDL simulations and generates mesh
- file *header_cube.txt* - header with simulation commands, which is used in every *.k* script

Conduction of parametric simulations

The process of automation was divided into several steps:

- **1. Mesh generation**
  In the script *A_mesh_generation.m*, the range of mesh densities for which mesh files will be generated can be defined. The mesh will be generated in both mapped and free types, and the selected range may be higher than the one that will be used later in the simulations. The



Name	Date modified	Type	Size
1_mesh_cdb	07/11/2021 12:41	File folder	
2_mesh_k	07/11/2021 12:41	File folder	
3_assembled_k_scripts	07/11/2021 12:39	File folder	
APDL	07/11/2021 12:41	File folder	
parameters	07/11/2021 12:41	File folder	
txt_files	07/11/2021 12:41	File folder	
A_cdb_mesh_generation_cube	17/10/2021 21:27	MATLAB Code	2 KB
B_cdb2k	17/10/2021 23:26	MATLAB Code	5 KB
C_Matlab_central_script_LSD_cube	07/11/2021 12:38	MATLAB Code	15 KB
D_LSD_solver	02/11/2021 19:14	MATLAB Code	2 KB
fsoGetShortPath	05/01/2015 16:19	MATLAB Code	2 KB
func_area_between_curves	17/10/2021 21:07	MATLAB Code	4 KB
func_variables_substitution	17/10/2021 21:12	MATLAB Code	2 KB
APDL_cdb_mesh_generation_cube	17/10/2021 21:25	TXT File	1 KB
APDL_run_all_simulations_cube	17/10/2021 21:29	TXT File	4 KB
header_cube	29/10/2021 10:57	TXT File	4 KB

Figure 5.5: Necessary items in the directory.

script is derived from the central script controlling the simulations of the cube in subchapter 4.1, so by running it, it will compile several parameter files in the folder *parameters* and create an APDL script named *APDL_generate_all_meshes_cube*. After loading this script into the APDL, it starts the mesh creation process, while individual mesh files in the *.cdb* format will be saved in the *A_mesh_cdb* folder under specific names. Then *cdb2k.m* script has to be run, which converts the mesh to *.k* format and saves it under the same name (but with a different extension) to the *2_mesh_k* folder.

- **2. Preprocessing - *.k* scripts assembly**

  In the script *C_Matlab_centra_script_LSD_cube.m* parameters for which *.k* scripts will be created can be defined - the range of the mesh density, hexahedral elements for the mapped mesh and tetrahedral elements for the free mesh, material models, strain-rates, and deformation speed modes. Because the assembly of the *.k* scripts is relatively complex, the material model, elements, and deformation speed modes can be selected only from those presented in the subsection 5.3.1. Running a script set to generate *.k* scripts will generate simulation *.k* scripts with all combinations of defined parameters using the *func_variables_substitution.m* function by merging the header with substituted parameters, the corresponding mesh file, and some extra auxiliary text files.

- **3. Solution - .*k* scripts solving**
  Running the script *D_LSD _solver.m* will gradually solve the individual scripts. To shorten the total computation time, it is advisable to run several of the most time-consuming simulations manually using the LS-Run application.

- **4. Posprocessing - results reading and processing**
  By running the script *C_Matlab_central_script_LSD_cube.m* set to load the results, the simulation results (deformation curves and simulation durations) are loaded into one united database.

5.2.4 Simulations set 1

In the initial attempts to simulate the quasi-static deformation of the cube, it was found that while maintaining truly quasi-static conditions (at strain-rate 0.5 mm/s), these simulations were extremely time-consuming - it would take years or decades to solve simulations similar to those in the previous chapter. It was therefore decided that the solution time must be significantly reduced in some way. This was achieved by increasing the strain-rate (and thus shortening the simulation time) and by introducing the linearly increasing displacement speed mode, which mitigates the wave behaviour of the specimen at high strain-rates. This first parametric study is therefore focused primarily on comparison between the two displacement speed modes and the selection of a suitable strain-rate, at which the results will correspond to the results of quasi-static simulations, and simultaneously the time required for their computation will be reduced to a minimum.

Used parameters

A total of 648 simulations was created, where displacement speed modes, strain-rates, material models, mesh densities, mesh types, and element formulations were varied. Used parameters are summarised in figure 5.6. The cube was only deformed to a strain of 2,5 % to capture the wave behaviour in this range with a total of 1001 points at which the results were read.

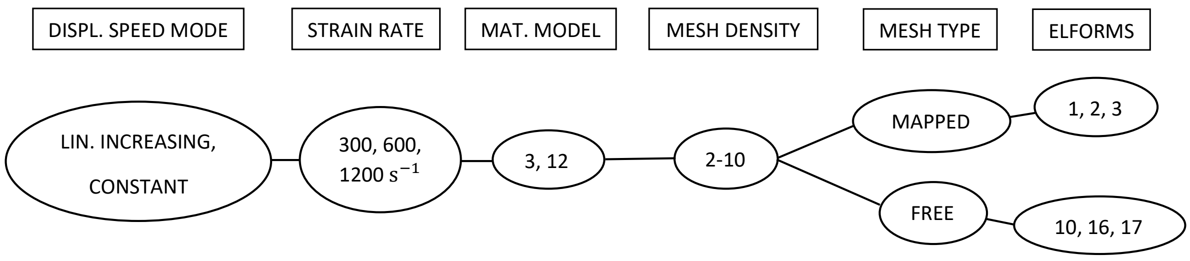
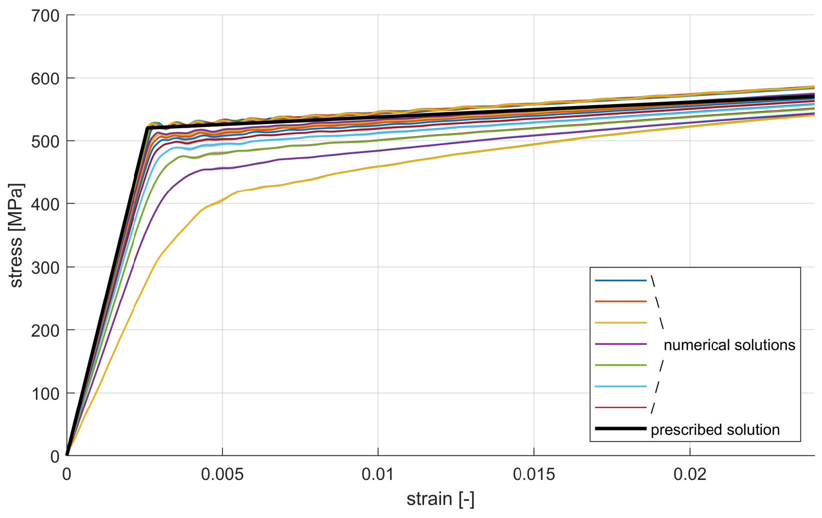


Figure 5.6: Scheme of used parameters for simulations set 1 with the cube in LS-DYNA.

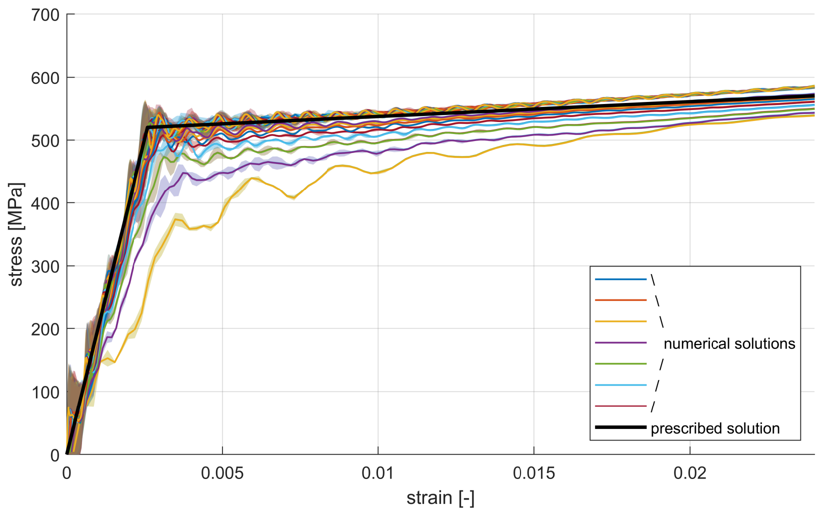
Results

Figure 5.7 shows all calculated deformation curves sorted into six groups according to used deformation speed mode and strain-rate. Comparing the two deformation speed modes, it can be seen that the linearly increasing deformation speed mode gives more accurate results with a lot smaller oscillations. At constant deformation speed mode, the deformation curve gets disoriented right at the beginning of the simulation, which is caused by the sudden speed introduction. The amplitudes of the waves are higher in comparison with the other mode. At the linearly increasing deformation speed mode, the theoretically linear part of the curve stays linear, and wave behaviour of the material starts manifesting when the curves pass the yield point. The amplitudes of the waves are smaller in comparison with the other mode, and the reaction forces from both walls are more uniform (the area surrounding the curves is thinner). Additionally, the curves near the linear part and yield point are smoother because more data is read from the simulations at the beginning than at the end.

strain-rate $600\,\mathrm{s}^{-1}$ was chosen to be the best for further research. Although the deformation curve is still a little wavy, the resulting inaccuracies will be negligible when the simulations are performed up to higher strains. This strain-rate might also be used in further simulations on the structure as the resulting errors will be minor considering the measurement errors of the material parameters. And because different parts of the structure reach yield stress at different times, the resulting wave behaviour might be even less impactful.



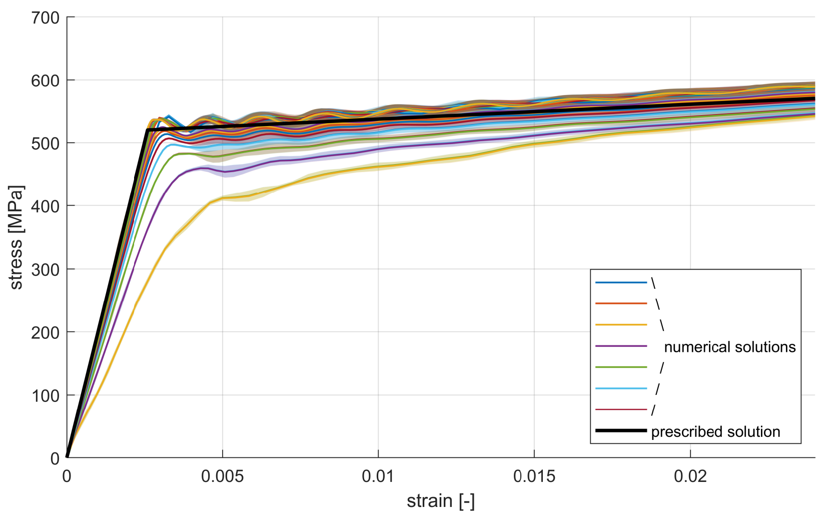
(a) : strain-rate 300 s^{-1}, linearly increasing deformation speed.



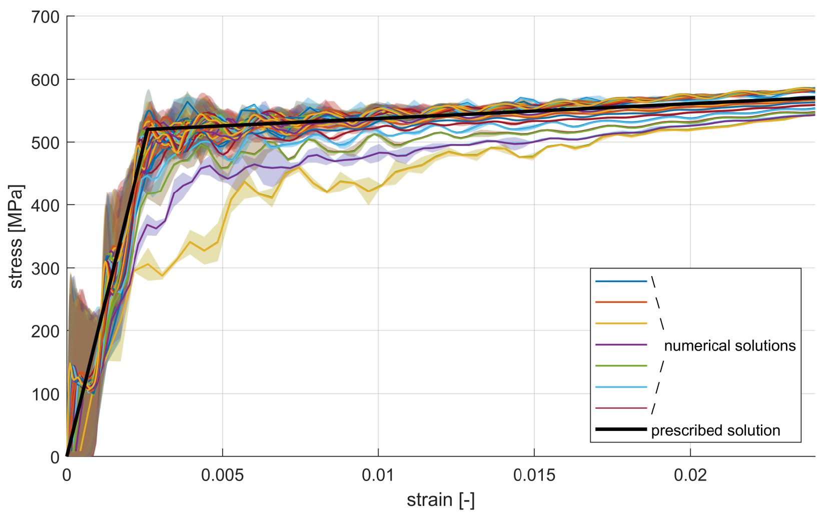
(b) : strain-rate 300 s^{-1}, constant deformation speed.

Figure 5.7: Comparison of prescribed and numerically calculated stress-strain curves for multiple strain-rates and two types of deformation speed modes (curves display the average stress calculated from reactions of the top and bottom walls, the edges of the area surrounding them are stresses from these walls).

The energy balance of the model was also checked for the performed simulations. When examining several randomly selected simulations with the highest strain-rate, it was found that, except for the first few time-steps, the internal energy of the model is always at least several orders of magnitude higher than the kinetic energy of the model. This means that the deformation forces are dominant in comparison with the kinetic forces.



(c) : strain-rate $600\,\mathrm{s}^{-1}$, linearly increasing deformation speed.

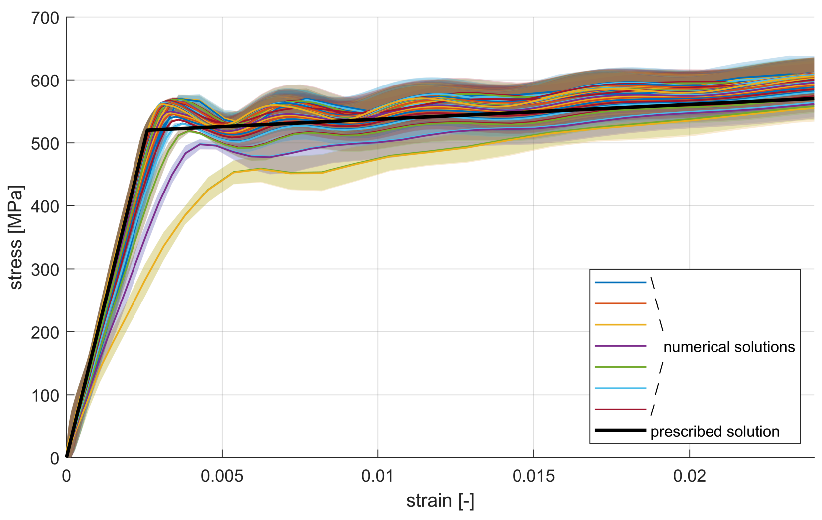


(d) : strain-rate $600\,\mathrm{s}^{-1}$, constant deformation speed.

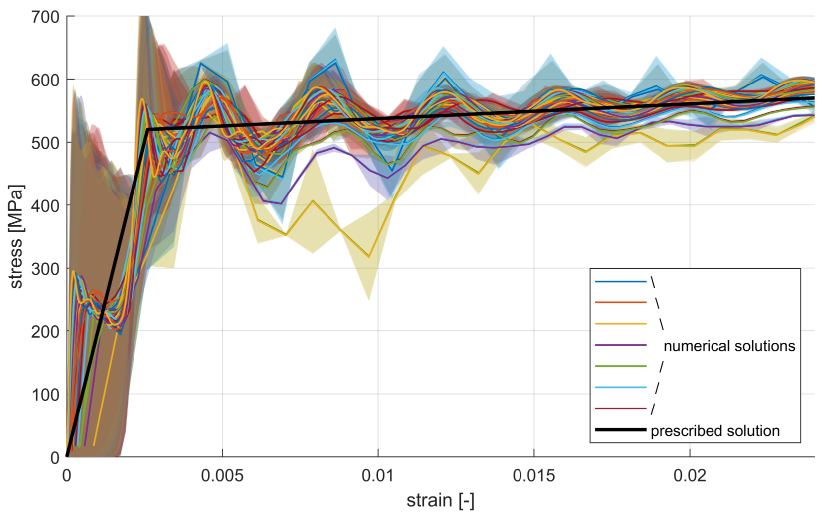
Figure 5.7: Comparison of prescribed and numerically calculated stress-strain curves for multiple strain-rates and two types of deformation speed modes (curves display the average stress calculated from reactions of the top and bottom walls, the edges of the area surrounding them are stresses from these walls).

5.2.5 Simulations set 2

After finding out the most suitable deformation speed mode (linearly increasing) and strain-rate ($600\,\mathrm{s}^{-1}$), this study aims to select more accurate of the two introduced material models.



(e) : strain-rate $1200\,\mathrm{s}^{-1}$, linearly increasing deformation speed.



(f) : strain-rate $1200\,\mathrm{s}^{-1}$, constant deformation speed.

Figure 5.7: Comparison of prescribed and numerically calculated stress-strain curves for multiple strain-rates and two types of deformation speed modes (curves display the average stress calculated from reactions of the top and bottom walls, the edges of the area surrounding them are stresses from these walls).

Used parameters

A total of 108 simulations was created, where material models, mesh densities, mesh types, and element formulations were varied. Used parameters are summarised in figure 5.8. The cube was deformed to a strain of 50 % to examine the divergence of the deformation curves with a total of 5001 points at which the results were read.

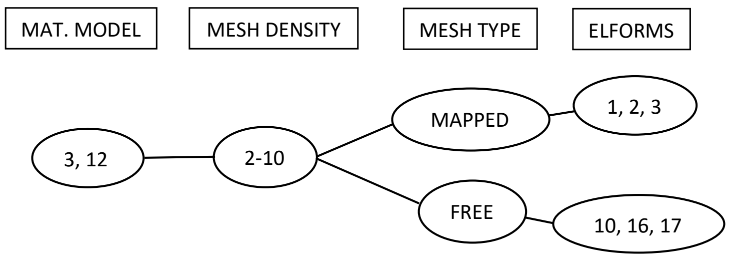


Figure 5.8: Scheme of used parameters for simulations set 2 with the cube in LS-DYNA.

Results

In the figure 5.9 deformation curves from all simulations are presented. It can be seen that all deformation curves from the simulations diverge relatively quickly from the prescribed solution. At low strains, the curves of the corresponding pairs of simulations differing by the material model overlap almost exactly. The difference becomes more visible at higher strains, where the curves are more spread out. The majority of the curves follow a very similar line (the thick blue and orange line with the highest stress), while the other curves correspond mainly to simulations with element formulation 3.

Figure 5.10 displays the solution error dependence on the mesh density calculated by the area between the numerical and prescribed deformation curve on the whole interval of strains. Because the majority of the deformation curves followed the same pattern, it is not surprising that the error stays constant for almost all simulations using element formulations 1, 2, 10, 16, and 17. Even though the simulations with element formulation 3 have the lowest error, the fact that the results get worse with a denser mesh suggests that these simulations should be evaluated very carefully. However, most importantly, in all instances (except for two cases with the lowest mesh density), material model 12 showed to be a little more accurate than material model 3.

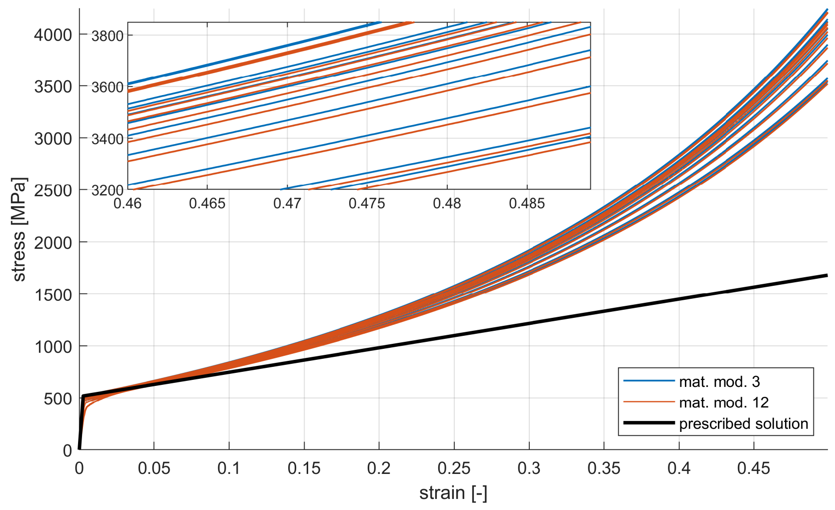


Figure 5.9: Comparison of prescribed and numerically calculated stress-strain curves (curves display the average stress calculated from reactions of the top and bottom walls, the edges of the area surrounding them are stresses from these walls).

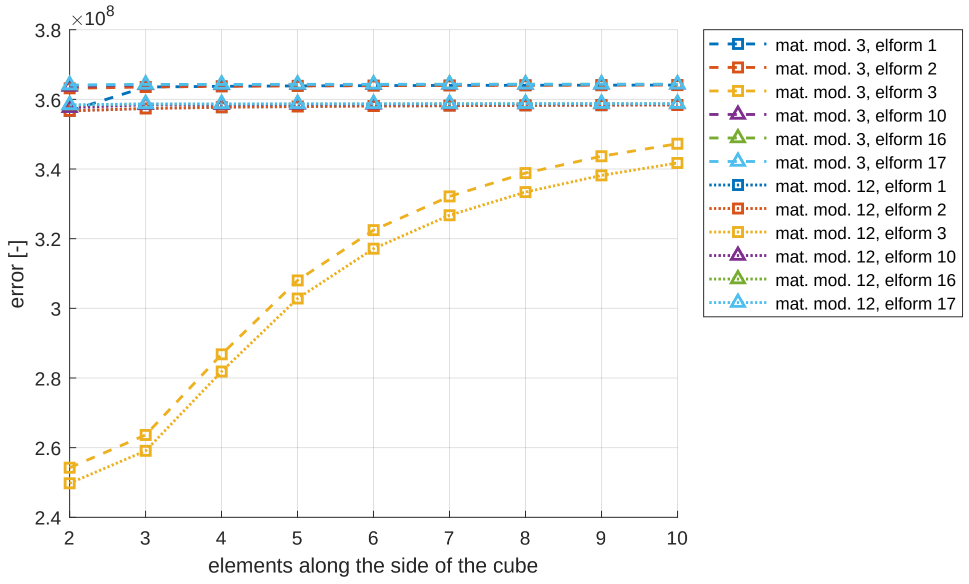


Figure 5.10: Solution error plotted against mesh density for different elements and material models.

The graph 5.11 shows the dependence of the solution time on the mesh density for all used element formulations and both material models. It can be seen that the difference in solution time between material models is negligible. Given the fact that all simulations with material model 3 required 1183 s and all simulations with material model 12 required 1182 s, both material models can be considered just as time-consuming. Therefore, material model 12 will be used from now on.

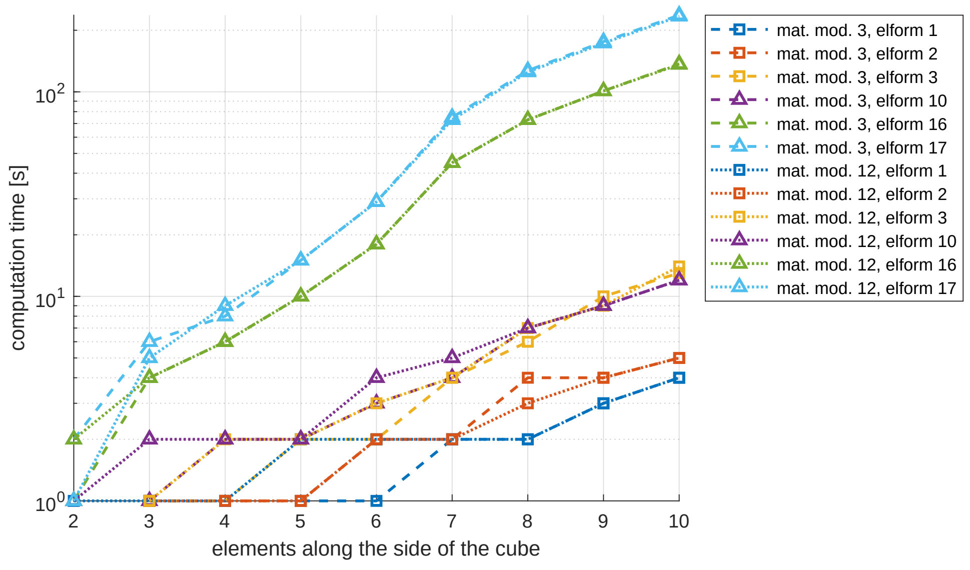


Figure 5.11: Computation time plotted against mesh density for different elements and material models.

5.2.6 Simulations set 3

Knowing the better of the material models, the last simulations set was performed, which basically only extends the results obtained in the last set for higher mesh densities.

Used parameters

A total of 114 simulations was created, where only mesh density, mesh type, and elforms were varied. Used parameters are summarised in figure 5.12. The cube was deformed to a strain of 50 %, and results were read at 5001 points.

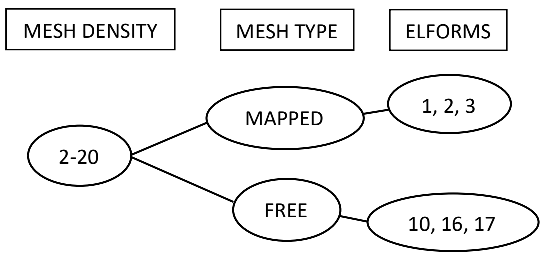


Figure 5.12: Scheme of used parameters for simulations set 3 with the cube in LS-DYNA.

Results

In the figure 5.13 it can be observed that cubes with all elements and all densities except for element formulation 3 have very similar deformation curves. Deformation curves of element formulation 3 look very different from all other curves due to the higher flexibility of the cube.

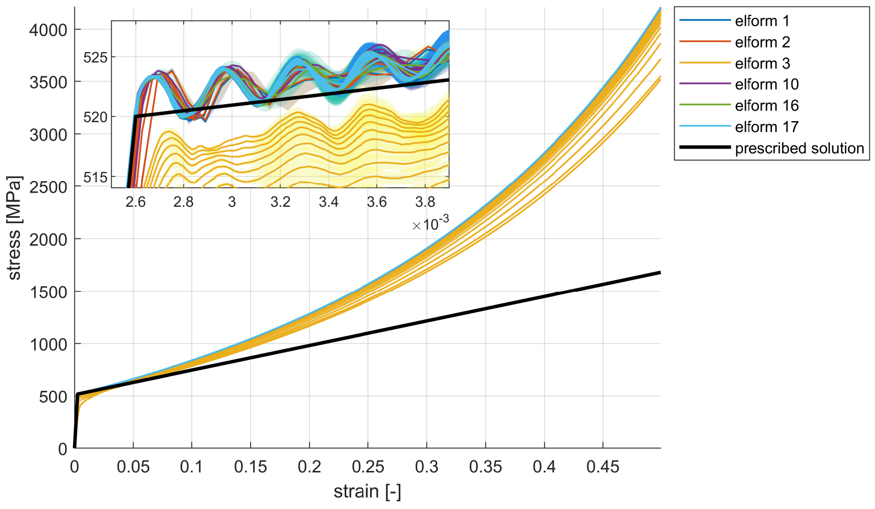


Figure 5.13: Comparison of prescribed and numerically calculated stress-strain curves (curves display the average stress calculated from reactions of the top and bottom walls, the edges of the area surrounding them are stresses from these walls).

In figures 5.14 and 5.15 continuing trends from the graphs 5.10 and 5.11 respectively are displayed. The error of the solutions stays almost constant for all the mesh densities and element formulations except for element formulation 3. Despite simulations with element formulation 3 having smaller solution errors, the stress-strain curve is rather incorrect in the linear deformation stage. Simulations with free mesh were generally much more time-consuming than simulations with mapped mesh, which was caused by having many more elements and nodes. Simulations with element formulation 17 were the most time-consuming, while simulations with element formulation 1 were the least time-consuming, with a difference of approximately two orders of magnitude.

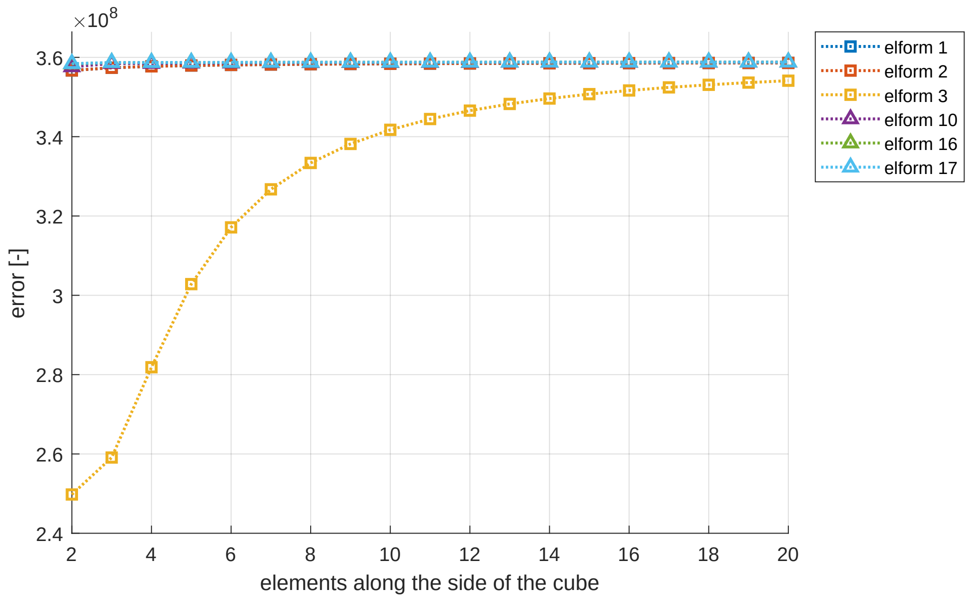


Figure 5.14: Solution error plotted against mesh density for different elements and material models.

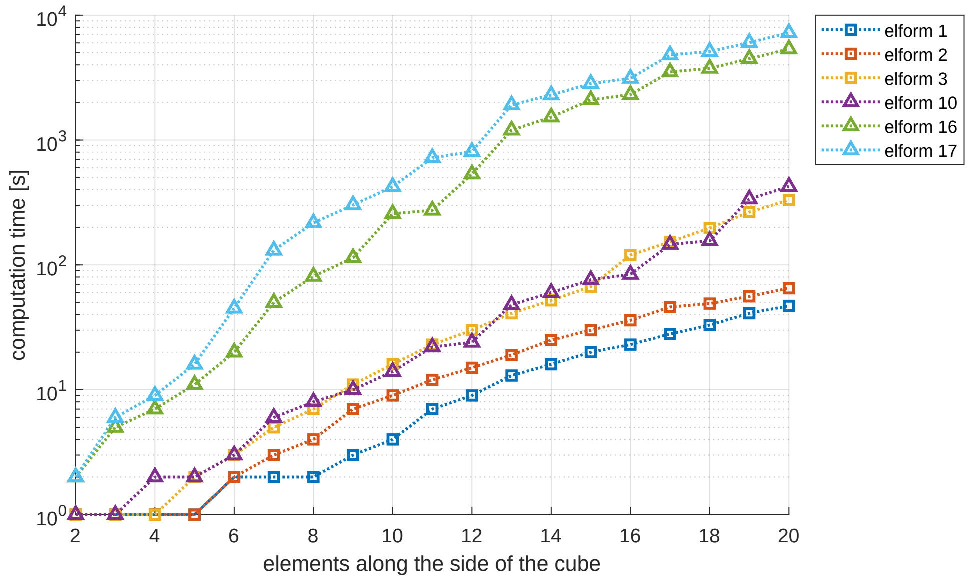


Figure 5.15: Computation time plotted against mesh density for different elements and material models.

5.2.7 Conclusion of the subchapter

By evaluating the results of the parametric simulations, it was found that, independently of any varied parameters, the numerical deformation curves diverged significantly from the prescribed solution. At a deformation of around 45 %, the value of the stress in the numerical model is already approximately twice as high as the prescribed one. This fundamentally limits the application of the explicit FEM formulation for quasi-static simulations.

It was shown that linearly increasing deformation speed mode returns more accurate results than constant deformation speed mode and the material model 12 returns more accurate results than the material model 3, both with almost the same computation costs. strain-rate of $600\,\mathrm{s}^{-1}$ was selected as the ideal strain-rate, which both provides very accurate results and requires little computation cost. Different element formulations with their corresponding mesh types were examined, and all element formulations 1, 2, 10, 16, and 17 return very similar results. Element formulation 3 turned out to have a smaller solution error but was inaccurate in the linear region of deformation.

5.3 Auxetic structure

This subchapter is dedicated to parametric explicit simulations of quasi-static deformation of the structure presented in the subchapter 3.2.

5.3.1 Investigated simulation parameters

Simulations were mainly focused on the parameters presented in this section.

strain-rate

Although a suitable strain-rate was established performing simulations with the cube, simulations with different strain-rates will still be performed to find how it contributes to results with the aim of shortening the computation time even more.

Mesh density

The density of the mesh was given by the number of elements along the cross-sectional edge of the beams in the structure. The mesh used in the simulations was also transferred from APDL and therefore is exactly the same as in subchapter 4.2.

Elements

Tetrahedral element formulations 10, 16, and 17 were varied, which were introduced in the subsection 5.2.1.

5.3.2 Principles of simulations

The principle of simulations does not change much compared to the previous chapters. The change from explicit simulations on a cube is the determination of the deformation, which, as previously defined, is calculated from the displacements of 12 nodes. The list of ids of these 12 nodes for 3 different mesh densities was manually stored in the script *C_Matlab_central_script_LSD_structure.m*. In the *.k* scripts, the command **SET_NODE_LIST* defines the list of nodes for which the command **DATABASE_HISTORY_NODE_SET* outputs their displacements at defined times to a file *nodout*. The displacements are then read and processed as part of the results processing by the central Matlab script. The major change compared to implicit simulations with the structure is the definition of contact between elements, which is relatively easy to define in LS-DYNA using command **CONTACT_AUTOMATIC_SINGLE_SURFACE*. Another change compared to explicit simulations with the cube is the introduction of mass scaling using command **CONTROL_TIMESTEP*. Time-step was always controlled by setting a scale factor of how much longer will the time-step be in comparison with the initial time-step. Mass of small elements creating the structure was then changed accordingly during the course of the simulations. Reaction forces from the top and bottom plates are again outputted to files *bndout* and *nodfor* respectively.

5.3.3 Automation

An overview of the automation process is in figure 5.16. The principle of automation is very similar to the principle described in the subchapter 5.2. Because the automation process is even simpler than the automation process for the cube (only one displacement speed mode, one material model and one mesh type is used), there is no need to analyze the automation principal any further. Some automation processes were even skipped and done manually. For example, 3 needed mesh files were exported from APDL manually and because of high memory requirements, scripts were usually solved using LS-Run manually, where specific memory was manually assigned.

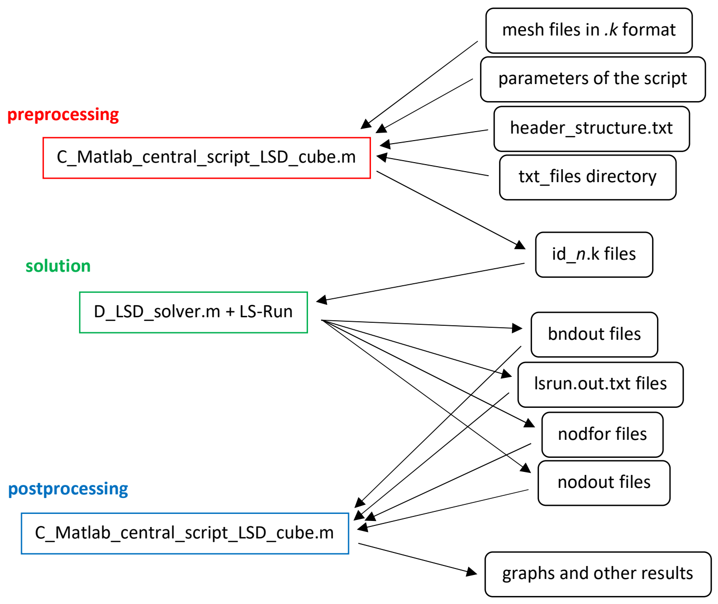


Figure 5.16: Block diagram of the simulations performation process in LS-DYNA with the structure.

5.3.4 Simulations set 1

The objective of this simulations set is to find how different strain-rates contribute to the resulting deformation curves.

Used parameters

A total of 12 simulations was created, where strain-rates and element formulations were varied. For all simulations, only mesh density 2 was used, as it is more accurate than MD 1 and not as costly as MD 3. Used parameters are summarised in figure 5.17. The structure was deformed to a total strain of 18 % so that no contact is involved in the simulations. This was done to decrease the computation cost of the simulations by eliminating the contact searching algorithm as well as to get deformation curves that are not affected by different collapse processes. Results were read from a total of 1001 points and a time-step scale factor 10 was used.

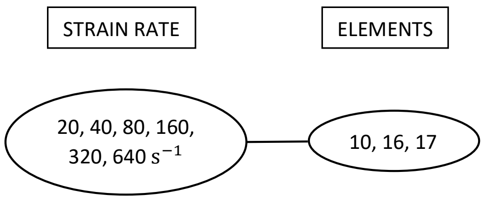


Figure 5.17: Scheme of used parameters for simulations set 1 with the structure in LS-DYNA.

Results

In the graph 5.18 deformation curves from all simulations can be seen in comparison with the experimental data. It is clear that the deformation curves are different for different strain-rates even though no strain-rate effects are incorporated in the material model. Deformation curves change significantly even for low strain-rate simulations, despite forces from the top and bottom plates being very similar.

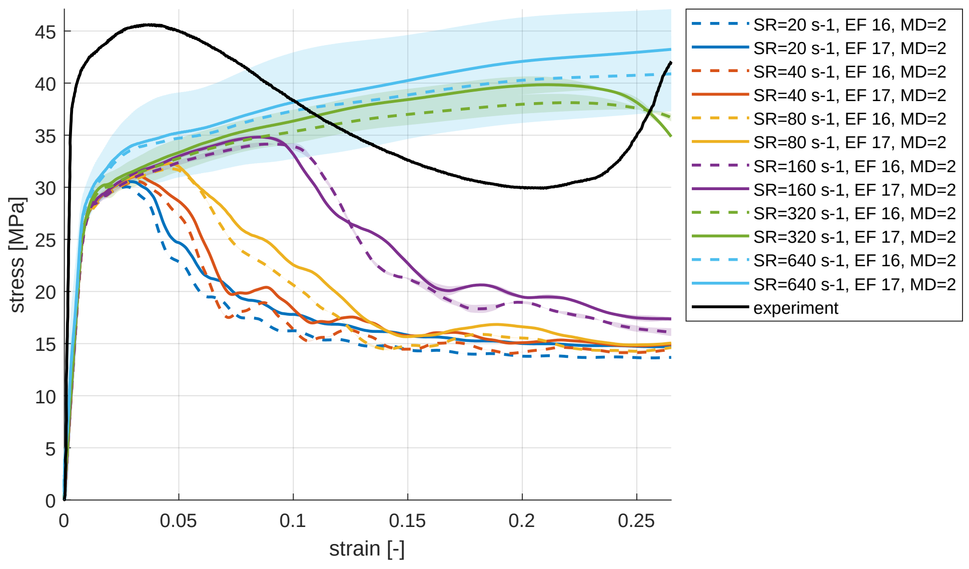


Figure 5.18: Deformation curves of calculated simulations (curves display the average stress calculated from reactions of the top and bottom plates, the edges of the area surrounding them are stresses from these plates).

It can be seen that with lower strain-rates, the differences among deformation curves get smaller and smaller. Simulations with even lower strain-rates would be beneficial for deeper understanding, however, the longest simulation already took almost 16 hours and simulations with even lower strain-rates were not feasible. Strain-rate of $80\,\mathrm{s}^{-1}$ was chosen for further simulations as it offers relatively good computation times with little distortion from lower strain-rate simulations. Moreover, at strains of $15 - 25\,\%$ the stress is very similar to the stress of lower strain-rate simulations.

5.3.5 Simulations set 2

Following the previous simulations set, the objective of this simulations set is to study the differences among different elements and different mesh densities.

Used parameters

9 simulations with all elements and all mesh densities were performed at strain-rate of $80\,\mathrm{s}^{-1}$. Used parameters are summarised in figure 5.19. Results were once again read from a total of 1001 points and time-step scale factor 10 was used. No contact was defined as the structure was deformed to a total deformation of $18\,\%$.

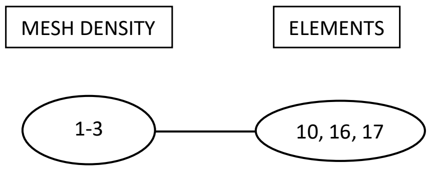


Figure 5.19: Scheme of used parameters for simulations set 2 with the structure in LS-DYNA.

Results

Two simulations with element formulation 16 and mesh densities 1 and 3 were prematurely terminated due to too big mass increases of some elements. The first one was successfully solved by changing the time-step scale factor from 10 to 8. In the second case, not only time-step scale factor was changed from 10 to 8, but also element erosion had to be implemented. This was done by tweaking a scale factor *dtmin* in the **CONTROL_TERMINATION* command, and setting *erode* parameter in the **CONTROL_TIMESTEP* to 1. Thanks to that, during the course of the simulation, 4 problematic elements were automatically deleted from the model for having too large mass increases. With these modifications, a slight change of results might be expected.

In figure 5.20 all deformation curves are presented. It can be clearly seen that the deformation curves of the simulations with element formulations 16 and 17 are very similar in shape to the experimental curve. Simulations with element formulation 10, however, do not reflect the experimental data and the other simulations data at all.

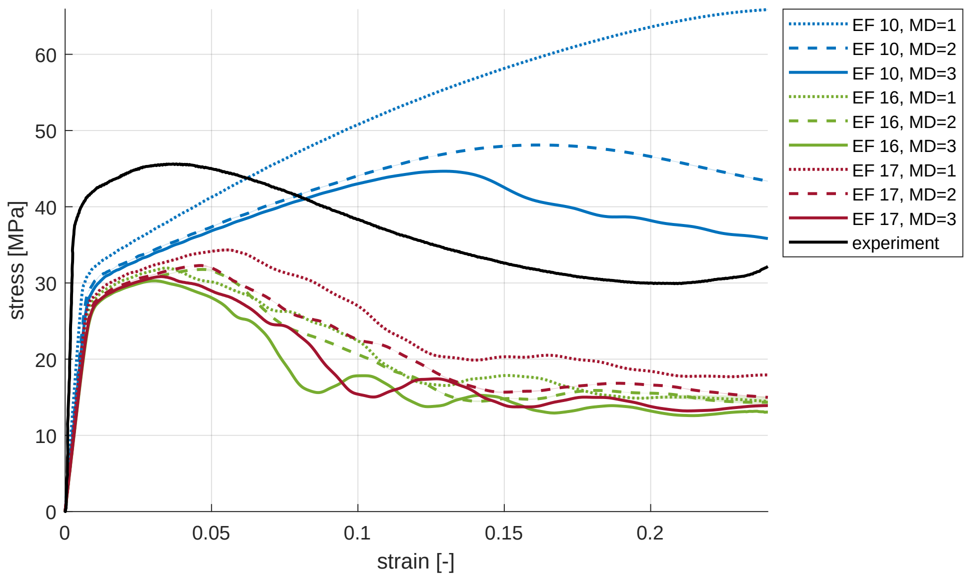


Figure 5.20: Deformation curves of calculated simulations (curves display the average stress calculated from reactions of the top and bottom plates, the edges of the area surrounding them are stresses from these plates).

Computation times of the simulations are shown in figure 5.21. The longest simulation took almost 33 hours and the shortest one 76 seconds. No relevant change of results due to modifications of 2 simulations was observed.

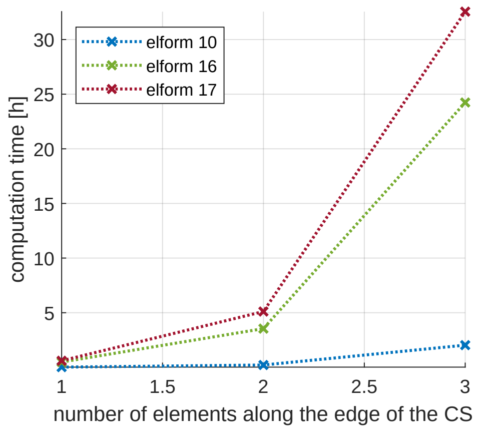


Figure 5.21: Computation time plotted against the mesh density for different element formulations.

5.3.6 Simulations set 3

In this set, the exactly same simulations as in the previous set will be performed to a higher total deformation.

Used parameters

The same parameters as in the previous set were used (see fig. 5.19). Command **CONTACT_AUTOMATIC_SINGLE_SURFACE* was used to implement contact between elements, and the structure was deformed to a total deformation of 50 %.

Results

Simulations with MD=3 and EF 16 and 17 were not performed as their estimated computation time was too high (71 and 134 hours, respectively). Simulation with MD=1 and EF 17 terminated with report *forrtl: severe (157): Program Exception - access violation*. It was discovered that by removing the contact defining command, simulation would run flawlessly. Wrong command, however, can not be the primary cause of the error, as the same command is used in all other simulations without issues. Checking on the internet, no official explanation was found. According to some unverified discussion forums information, this might be due to accessing memory the program is not allowed to access.

Figure 5.22 shows all calculated deformation curves. The result of the contact algorithm can be clearly seen as all deformation curves rise steeply at higher strains. It can be seen that all simulations with EF 10 are not really similar to neither the experimental results nor the other numerical results, and thus can not be considered correct. Other simulations, however, share some similarities among themselves and with the experimental data. The initial phase at strains $0 - 20\,\%$ is very similar in shape among all curves. Even the shape of the simulation curves is in general fairly similar to the experimental curve. Deformed structures are presented in figure 5.23.

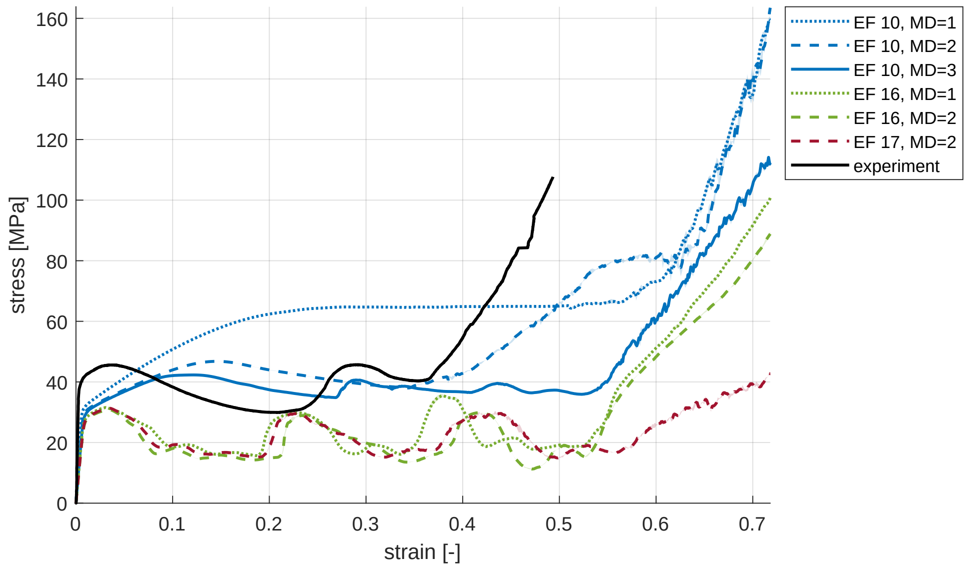
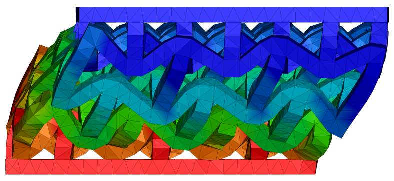


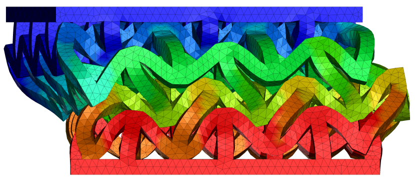
Figure 5.22: Deformation curves of calculated simulations (curves display the average stress calculated from reactions of the top and bottom plates, the edges of the area surrounding them are stresses from these plates).

5.3.7 Conclusion of the subchapter

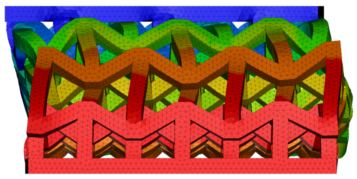
Running 3 sets of simulations, it was found that simulations with strain-rate $80\,\mathrm{s}^{-1}$ correspond quite well with the experimental results and have relatively



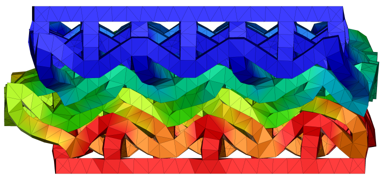
(a) : EF 10, MD=1



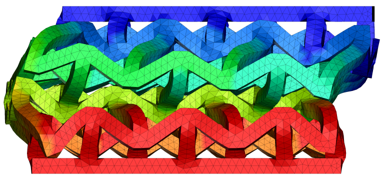
(b) : EF 10, MD=2



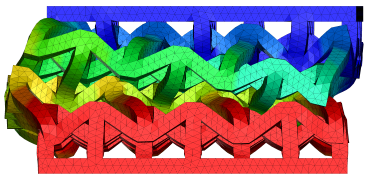
(c) : EF 10, MD=3



(d) : EF 16, MD=1



(e) : EF 16, MD=2



(f) : EF 17, MD=2

Figure 5.23: Deformed structures at total deformation of 50 %.

acceptable computation costs. Element formulation 10 showed to be time-efficient but very inaccurate, which significantly limits its use. Results of simulations with element formulations 16 and 17 are very similar as well as simulations with these elements but different mesh densities.

Chapter 6

Discussion

This chapter explains why the linear material model was not considered in the simulations, the achieved results are evaluated, and the informative value of the obtained results is examined.

6.1 Omission of the linear material model

The inclusion of a linear material model was intentionally omitted from the simulations. This significantly reduced the number of simulations that had to be calculated and simplified the whole process of creating simulations. The definition of a linear material model is suitable in those simulations where it is assumed that the deformations will take place only in the linear region. However, all simulations performed in this work extended relatively far beyond the linear region, thus confirming that simulations with a linear material model would indeed yield distorted results.

6.2 Evaluation of achieved results

In figure 6.1 the most representative simulations from both APDL and LS-DYNA are compared with the experimental results. It can be seen that the implicit simulations do not really match the experimental results. Moreover, because defining contact in APDL is a relatively advanced task, the implicit deformation curves do not extend to higher strains. On the other hand, explicit simulations reflect the experimental results quite well, and thanks to the easy contact definition, extend even beyond the experimental results.

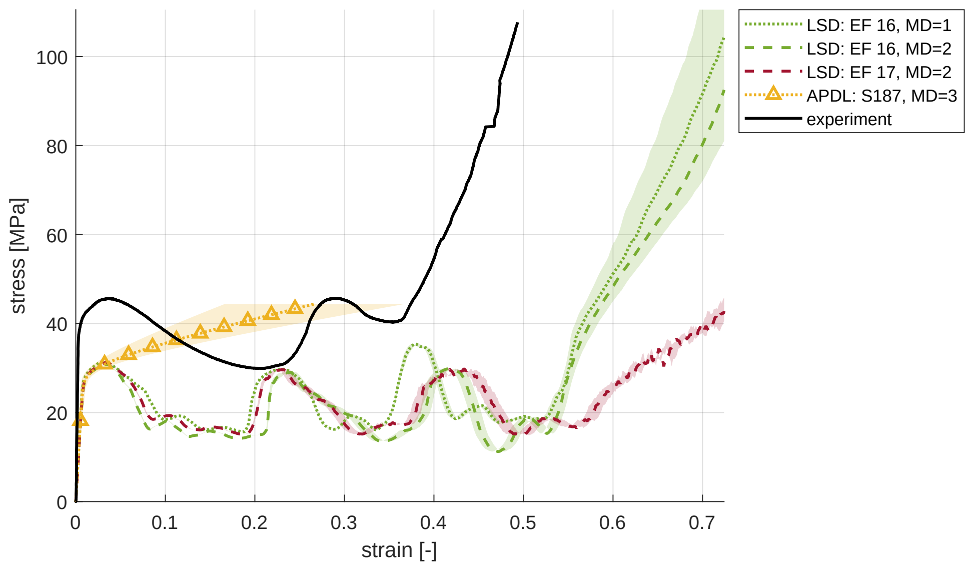


Figure 6.1: Comparison of different deformation curves (edges of the area along the curves reflect the standard deviation of strains).

Several factors can then cause the discrepancy among the simulations and the experiment. Firstly, in APDL, the structure was deformed very symmetrically, while in LS-DYNA, the structure deformed in an uncontrolled manner, similar to a real experiment. This explains why the stress only rises in APDL simulations, while in LS-DYNA, there are multiple waves in the deformation curve corresponding to collapses of the vertical beams. The misalignment between experimental and explicit curves can be caused by a wrongly assigned material model, which was determined from possibly inaccurate parameters. Another reason for the inaccuracies is the printing method, which is not capable of printing sharp edges and corners and thus does not correspond well with the numerical models. Lastly, it can be caused by oversimplified numerical model, which unlike the real structure, does not crack when deformed.

6.3 Evaluation of the applicability of results

All this work was closely focused on the study of the same structure with a relatively small number of cells ($3 \times 3 \times 2$). However, in practical applications where such structure might be used, the number of used cells can vary significantly. Therefore, for a broader application of the results, it is crucial to determine whether the data obtained from the simulations are transferable to structures with a different number of cells.

To do this, the beam model of the structure presented in subsection 4.2.1 was used, for which the number of cells in the structure can be changed parametrically and which has a relatively low computation cost. A script has been written in Matlab that can be used to generate scripts with parameters and the main script for running all simulations in APDL, very similar to the ones in chapter 4. The same geometric dimensions of the cell and material properties were used as in the rest of the work. The model used a quadratic element BEAM189 with a mesh density given by three elements along the beam. The structure was deformed to a total relative deformation of 1 % in one single load-step, and only the number of cells was changed. For the numbers of cells along the x and y axes in the range of 1-10, numbers of cells along the z axis ranged 1-12.

Both graphs in figures 6.2 and 6.3 show that the stress decreases with increasing number of cells (regardless of in which direction) and converges to some value. Absolute values of stress in the graphs are not important, rather the proportions between them are. It can be seen that the results for the examined structure can vary largely, based on the number of cells. This suggests, that the results are relevant mainly for analysis of structures with very similar number of cells.

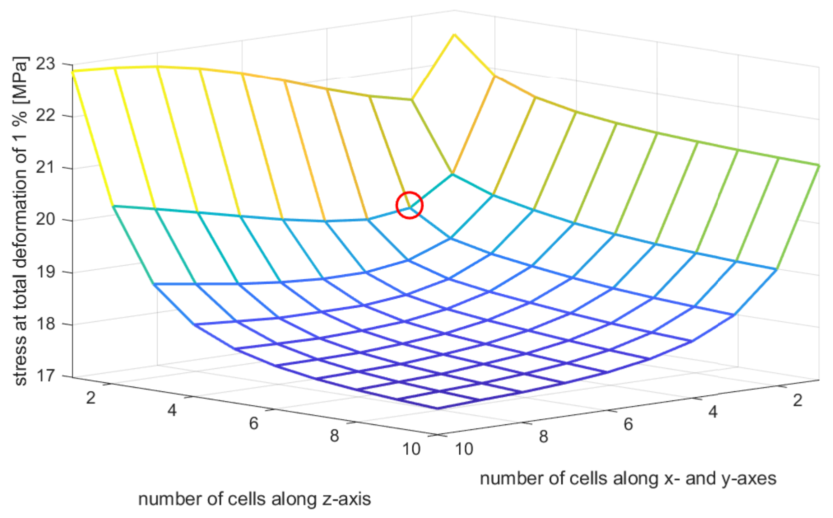


Figure 6.2: Stress in the deformed structure plotted against number of cells along the base axes and z axis (examined structure is highlighted in red).

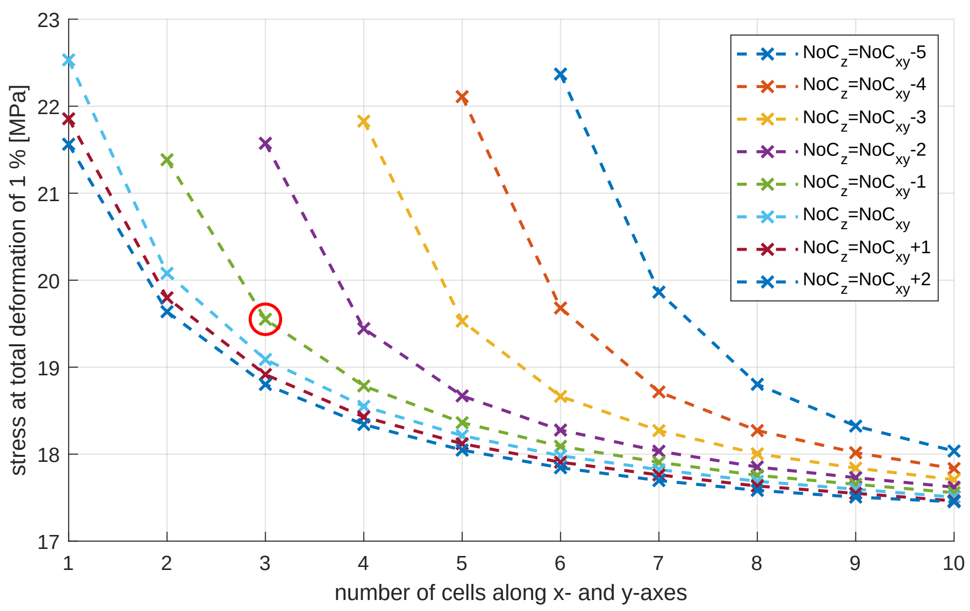


Figure 6.3: Stress in the deformed structure plotted against number of cells along the base axis for different structure proportions (examined structure is highlighted in red; NoC_{xy} - number of cells along x and y axes, NoC_z - number of cells along z axis.

Chapter 7

Conclusion

Many scripts and functions were presented to achieve the objectives of the work, thanks to which a total of 1037 simulations were performed in 9 simulations sets. These scripts and functions can be later used to simulate the same or similar problems with different parameters, or they can be used as templates for other parametric studies. Thanks to simulations with the cube always preceding simulations with the structure, better parameters could be selected to maximize the value of the results, and computation times could be effectively used.

In the chapter 4 two scripts for each specimen were introduced, which can be used to automatically create and calculate parametric simulations in the Ansys APDL environment. An exceptional accuracy of the obtained results from the deformation of the cube was found, which very precisely copied the prescribed solution. Furthermore, two computational models of the 3D re-entrant honeycomb structure were created - one consisting of solid type elements and the other simplified consisting of beam type elements. Differences in their behaviour were demonstrated using parametric simulations, and the results of one simulation with a solid model were compared with data from a real experiment.

In the chapter 5 several scripts and functions were introduced, which together automated the process of creating and calculating simulations in the LS-DYNA environment. Parametric simulations of the deformation of the homogeneous cube revealed a rapid divergence of the calculated deformation curve from the prescribed solution. Nevertheless, the deformation speed mode and strain-rate were presented, at which the results obtained are not marked by inertial forces and which can be used to significantly shorten the calculation of quasi-static simulations compared to using the real strain-rate used in experiments. In simulations with the structure, both higher strain-rates and mass scaling were used for computation time acceleration, which proved to be very effective. A suitable combination of strain-rate and mass scaling factor was found, which enabled an effective computation of many simulations. The resulting deformation curves were compared with the experimental curve and showed to be fairly similar.

A deeper analysis of similarities and differences among the results was presented in chapter 6 together with an explanation of why the linear material model was not used. It was also demonstrated that the results from this work do not reflect the 3D-reentrant honeycomb structure in general but reflect only this specific structure with this number of cells.

In conclusion, there are also some further improvements which could be done in this work, but are out of its scope. Contact could be defined among elements in the APDL simulations on the structure to obtain correct deformation curves even for higher strains. A better LS-DYNA solver which could solve multiple scripts at the same time would be convenient for a faster solution without manual work. Finally, the input parameters could be tweaked so that the numerical results correspond better to the experimental ones. Nevertheless, a solid foundations for future work have been laid.

All presented scripts, functions, and results (saved in *.mat* files, not the original ones) are attached to this work.

Appendix A

Bibliography

[1] Z. Chen, L. Liu, S. Gao, W. Wu, D. Xiao, and Y. Li. Dynamic response of sandwich beam with star-shaped reentrant honeycomb core subjected to local impulsive loading. *Thin-Walled Structures*, 161:107420, 2021.

[2] J. T. Katsikadelis. Chapter 11 - the finite element method. In John T. Katsikadelis, editor, *Dynamic Analysis of Structures*, pages 359–522. Academic Press, 2020.

[3] Z. Bi. Chapter 1 - overview of finite element analysis. In Zhuming Bi, editor, *Finite Element Analysis Applications*, pages 1–29. Academic Press, 2018.

[4] M. Fusek and J. Rojíček. Metoda Konečných Prvků I. `https://projekty.fs.vsb.cz/463/edubase/VY_01_010/METODA%20KONE%C4%8CN%C3%9DCH%20PRVK%C5%AE%20I.pdf`. Accessed on 25.10.2021.

[5] J. Petruška. Počítačové metody mechaniky II. `http://www.kvm.tul.cz/getFile/id:2499/Petru%C5%A1ka%20-%20Po%C4%8D%C3%ADta%C4%8Dov%C3%A9%20metody%20mechaniky%20II.pdf`. Accessed on 6.11.2021.

[6] M. Fusek and R. Halama. MKP a MHP. https://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/metoda_konecnych_prvku_a_hranicnich_prvku.pdf. Accessed on 25.10.2021.

[7] Ansys Learning. Discussion of Time-Step Size — Lesson 3. https://www.youtube.com/watch?v=NW1uzVi4cp0&t=214s&ab_channel=AnsysLearning. Accessed on 23.10.2021.

[8] Ansys, Inc. Theory Reference for the Mechanical APDL and Mechanical Applications. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjipb-s7a70AhXjgv0HHXQrBpUQFnoECAgQAQ&url=https%3A%2F%2Fwww.researchgate.net%2Ffile.PostFileLoader.html%3Fid%3D5687835e5cd9e345098b4568%26assetKey%3DAS%253A313357117132800%2540145172156653 8&usg=AOvVaw0_-wLTQIXLmf43S_KYt0B3. Accessed on 29.10.2021.

[9] Newton's method. https://en.wikipedia.org/wiki/Newton%27s_method. Accessed on 27.10.2021.

[10] Livermore Software Technology. LS-DYNA Theory Manual. https://www.dynasupport.com/manuals. Accessed on 25.10.2021.

[11] Time step size. https://www.dynasupport.com/tutorial/ls-dyna-users-guide/time-step-size. Accessed on 29.10.2021.

[12] Mass scaling. https://www.dynasupport.com/howtos/general/mass-scaling. Accessed on 29.10.2021.

[13] I have very long run times. What can I do? https://www.dynasupport.com/faq/general/i-have-very-long-run-times.-what-can-i-do. Accessed on 4.11.2021.

[14] M. Neuhäuserová, P. Koudelka, J. Falta, M Adorna, T. Fíla, and Zlámal P. Strain-Rate and Printing Direction Dependency of Compressive Behaviour of 3D Printed Stainless Steel 316L. *Acta Polytechnica CTU Proceedings*, 2019. https://ojs.cvut.cz/ojs/index.php/APP/article/view/6029.

[15] P. Kelly. Solid Mechanics Part II: Engineering Solid Mechanics - small strain. https://pkel015.connect.amazon.auckland.ac.nz/

SolidMechanicsBooks/Part_II/08_Plasticity/08_Plasticity_06_Hardening.pdf. Accessed on 15.11.2021.

[16] J. Podešva. Počítačové Modelování Nelineárních Problémů. https://projekty.fs.vsb.cz/147/ucebniopory/978-80-248-2763-6.pdf. Accessed on 3.11.2021.

[17] I. Report. Constitutive Modeling and Material Behavior. http://www.riteh.uniri.hr/media/filer_public/c7/b4/c7b4b975-9474-4b66-a04f-ff3597ba61e7/d711_constitutive_modeling_and_material_behavior_interim_report.pdf. Accessed on 3.11.2021.

[18] ISO/ASTM. ISO/ASTM 52900:2021: Additive manufacturing — General principles — Fundamentals and vocabulary, 2021. https://www.iso.org/obp/ui/#iso:std:iso-astm:52900:ed-2:v1:en. Accessed on 3.11.2021.

[19] Z. Liu, D. Zhao, P. Wang, M. Yan, C. Yang, Z. Chen, J. Lu, and Z. Lu. Additive manufacturing of metals: Microstructure evolution and multistage control. *Journal of Materials Science Technology*, 100:224–236, 2022.

[20] X. Zhang and F. Liou. Chapter 1 - introduction to additive manufacturing. In Juan Pou, Antonio Riveiro, and J. Paulo Davim, editors, *Additive Manufacturing*, Handbooks in Advanced Manufacturing, pages 1–31. Elsevier, 2021.

[21] 3D Printing - Additive. https://make.3dexperience.3ds.com/processes/powder-bed-fusion. Accessed on 4.11.2021.

[22] Intel. Intel® Xeon® W-2265 Processor. https://ark.intel.com/content/www/us/en/ark/products/198015/intel-xeon-w2265-processor-19-25m-cache-3-50-ghz.html. Accessed on 26. 10. 2021.

[23] J. Chen, W. Chen, H. Hao, S. Huan, and W. Tao. Mechanical behaviors of 3d re-entrant honeycomb polyamide structure under compression. *Materials Today Communications*, 24:101062, 2020.

[24] T. Fíla, P. Koudelka, Zlámal P., J. Falta, M. Adorna, M. Neuhäuserová, J. Luksch, and Jiroušek O. Strain Dependency of Poisson's Ratio of

SLS Printed Auxetic Lattices Subjected to Quasi-Static and Dynamic Compressive Loading. *Advanced Engineering Materials*, 2019. https://doi.org/10.1002/adem.201900204.

[25] T. Fíla, Zlámal P., O. Jiroušek, J. Falta, P. Koudelka, D. Kytýř, T. Doktor, and J. Valach. Impact Testing of Polymer-filled Auxetics Using Split Hopkinson Pressure Bar. *Advanced Engineering Materials*, 2017. https://doi.org/10.1002/adem.201700076.

[26] Renishaw. *SS 316-0407 powder for additive manufacturing*, 2018. https://www.engineeringtoolbox.com/friction-coefficients-d_778.html.

[27] P Koudelka. *Numerical modelling of auxetic structures.* PhD thesis, Czech Technical University in Prague, 2020. https://dspace.cvut.cz/handle/10467/94303.

[28] Engineering ToolBox. Friction - Friction Coefficients and Calculator. https://www.engineeringtoolbox.com/friction-coefficients-d_778.html. Accessed on 26.10.2021.

[29] Summary of Element Types. https://www.mm.bme.hu/~gyebro/files/ans_help_v182/ans_elem/Hlp_E_CH3_2.html. Accessed on 5.11.2021.

[30] Livermore Software Technology. LS-DYNA Keyword User's Manual. https://www.dynasupport.com/manuals. Accessed on 19.10.2021.

[31] T. Erhart. Review of Solid Element Formulations in LS-DYNA. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiz4oLYj4f0AhVvhPOHHQtZB6gQFnoECAcQAQ&url=https%3A%2F%2Fwww.dynamore.de%2Fde%2Fdownload%2Fpapers%2Fforum11%2Fentwicklerforum-2011%2Ferhart.pdf&usg=AOvVaw2vXyAwZO-xkQpxi3Z8ySWT. Accessed on 25.10.2021.

[32] Hourglass (HG) Modes. https://ftp.lstc.com/anonymous/outgoing/support/FAQ_docs/hourglass.pdf. Accessed on 3.9.2021.

[33] J. Briot. Short path name on Windows (COM server). https://uk.mathworks.com/matlabcentral/fileexchange/48950-short-path-name-on-windows-com-server. Accessed on 21.10.2021.