TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Diplomarbeit

Clustering technique for efficient coupled simulations of grain structure and precipitation kinetics

zur Erlangung des akademischen Grades

Diplom-Ingenieurs

im Rahmen des Studiums

Technische Physik

eingereicht von

Elias Theil, BSc.

Matrikelnummer: 01242432

ausgeführt am Institut Institut für Festkörperphysik
der Fakultät für Technische Physik der Technischen Universität Wien
in Kooperation mit LKR Leichtmetallkompetenzzentrum Ranshofen GmbH

Betreuung
Betreuer: Univ. Prof. Dr. **Karsten Held**
Mitwirkung: Dr.tech. **Evgeniya Kabliman** &
Dr.tech. **Johannes Kronsteiner**

Wien, 17.11.2021 _____ _____

(Verfasser) (Betreuer)

Acknowledgements

First and foremost I want to thank my great co-workers. My thanks extends especially to my supervisors Johannes Kronsteiner and Evgeniya Kabliman, who provided valuable technical and theoretical knowledge and helped develop the clustering algorithm used in this thesis. Another special thanks goes out to Hugo Drexler for always providing fuel for my brain in the form of snacks.
Further I would like to thank Univ. Prof. Dr. Karsten Held for accepting to supervise this work on behalf of the TU Wien.
Additionally I want to thank all my various friends for the invigorating discussions, deep support and fun times we had. They provided a good balance to my work and studies.
In the end, I would also like to thank my family for supporting me through my long study years. You made all of this possible.

I would like to thank the Austrian Research Promotion Agency for financial support of this research work ('Modular Multiscale Modelling for Metal Additive layer Manufacturing', FFG-No. 874169) as part of the program 'Beyond Europe'.

Abstract

It is well known that the material structure at underlying levels (e.g. grain structure, precipitation kinetics, solute atoms, etc.) significantly influences macroscopic material properties. Modern forming simulations of metals therefore require a multiscale modelling approach to incorporate various physical effects on different scales. Efficient computation techniques are urgently needed to couple these different scales.

The focus of this thesis was the development of such a technique to describe the microstructure evolution of aluminium during deformation and heat treatment processes. In this context, two main goals were defined in this work. First, the various physical models needed to be investigated and sourced. Special attention was given to the model which described the grain structure development based on the evolution of a dislocation density. Second, the multiscale model to predict the simultaneous change in grain structure and precipitation kinetics needed to be implemented. While the corresponding material models had been already set up, the coupling of the grain structure evolution to precipitation kinetics required a new algorithm to make the computation feasible.

During the thesis, such an algorithm named *flexible clustering of parameters (flexiCluP)* was developed and tested. It aims to collect the various grain structure calculations from all the finite elements throughout the simulation specimen into few clusters with similar values. This way the precipitation kinetics can be computed once per cluster instead of once per element, which significantly reduces the simulation time without sacrificing too much accuracy.

For testing purposes, the simulations of uniaxial hot compression tests were performed and compared to available experiments. The results have demonstrated an improvement over uncoupled simulations, however some accuracy was still lacking. The possible reasons were analyzed in the end of the thesis, together with future steps for further improvement of the accuracy and the model.

Finally, both goals of this work were accomplished. First, the microstructure model used in the present work was sourced and its detailed implementation was described. Second, the coupling algorithm was successfully developed and coupled simulations could be run. This coupling algorithm was kept very general and could be thus used for other multiscale simulations.

Kurzfassung

Moderne Simulationen von Metallumformungen benötigen einen ”multiscale” Ansatz um physikalische Effekte auf unterschiedlichen Skalen einzubinden. Der Grund dafür ist, dass die zugrunde liegende Materialstruktur (Kornstruktur, Ausscheidungskinetik, gelöste Atome, ...) die makroskopischen Materialeigenschaften nachhaltig beeinflussen. Daher sind effiziente Algorithmen notwendig, um die verschiedenen Skalen miteinander zu verbinden. Der Fokus dieser Diplomarbeit war die Entwicklung eines derartigen Algorithmus zur Beschreibung der Mikrostrukturdynamik bei Aluminium während Verformng und Wärmebehandlung. Dabei gab es zwei Hauptziele. Erstens sollten die verschiedenen physikalischen Modelle untersucht und mit Quellen versehen werden. Insbesondere das Kornstrukturmodell stand dabei im Vordergrund. Zweitens sollte das Gesamtmodell für Kornstruktur und Ausscheidungskinetik implementiert werden. Während die einzelnen Materialmodelle schon bereitstanden, fehlte ihre gemeinsame Kopplung. Diese Kopplung benötigte einen neuen Algorithmus um die Rechenzeit gering zu halten.

Im Laufe der Diplomarbeit wurde ein solcher Algorithmus namens *flexible clustering of parameters (flexiCluP)* entwickelt und getestet. Der Algorithmus sammelt die verschiedenen Berechnungen der Kornstruktur in den finiten Elementen während eines Simulationsschrittes in wenigen Clustern zusammen. Dabei ist die Kornstruktur innerhalb eines Clusters sehr ähnlich und die Ausscheidungskinetik kann einmal für den gesamten Cluster statt einmal pro Element berechnet werden. Der Algorithmus reduziert damit die Rechenzeit ohne signifikant an Genauigkeit zu verlieren.

Um den Algorithmus zu testen wurden einachsige Stauchversuche simuliert und anschließend mit verfügbaren Experimenten verglichen. Die Resultate stellten eine Verbesserung zu ungekoppelten Simulationen dar, waren aber teilweise noch immer ungenau. Mögliche Gründe dafür und zukünftige Schritte zur Verbesserung der Genauigkeit und des Modells wurden am Ende dieser Diplomarbeit analysiert.

Zusammenfassend wurden beide Ziele dieser Arbeit erreicht. Erstens wurde das Mikrostrukturmodell mit Quellen versehen und erklärt. Zweitens war die Entwicklung des Koppelungsalgorithmus erfolgreich und es konnten gekoppelte Simulationen berechnet werden. Dabei wurde der Algorithmus aber allgemein gehalten, sodass er auch für andere ”multiscale” Simulationen verwendet werden kann.

Contents

Chapter 1

Introduction

In modern materials science, the modelling and subsequent simulation of materials plays an important role. Such simulations can support traditional experiments and reduce the number of prototypes when developing a new part, or they can highlight problems with a manufacturing process that previously went unnoticed. Thus it is necessary to obtain a good theoretical understanding of the underlying physical processes, and then to transform this understanding into usable material models for simulation.
This thesis concerns itself with the simulation of aluminium parts during pre heat treatment, deformation and post heat treatment. It aims to accurately predict the grain structure, which is a highly relevant parameter of a part, e.g. for determining its yield strength. A multiscale modelling approach is used to combine three different physical scales for an accurate description of the grain structure dynamics.

On the largest length scale, the Finite Element Method (FEM) model describes the deformation and thermal dynamics within the metal specimen at a scale of 10^{-2} m to 1 m. Such deformation has a direct impact on the microstructure at scales of about 10^{-6} m to 10^{-4} m. It locally breaks the crystal structure of the metal and leads to the formation of crystal defects such as dislocations, while heating can anneal out these defects. In turn, the density of crystal defects has a huge impact on metal strength, and thus the microstructure can influence the deformation behaviour at a large scale. On an even smaller length scale, precipitates of a size 10^{-8} m to 10^{-9} m or even the solute atoms of a size lower than 10^{-10} m can inhibit the microstructure change by impeding movement of the dislocations. In turn, the precipitation kinetics depend strongly on the microstructure, since crystal dislocations can serve as nucleation sites for precipitates. Figure 1.1 shows these relationships schematically. To study and implement this multiscale approach ,the following goals are defined in this thesis.

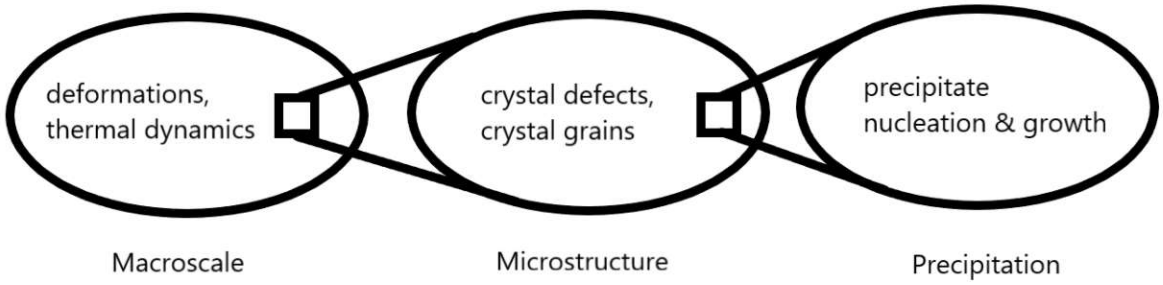

Fig. 1.1: Illustration of the interaction of different models within a multiscale model.

The first goal is to study and source the models used at the different length scales, especially the microstructure model which describes the grain structure dynamics and is called the Mean Dislocation Density Model (MD^2M). This study will form a necessary basis for potential further development of these models and should also provide a good point of entry to the whole subject.

The second goal is to develop and implement an efficient algorithm for coupling the simulations at different length scales. Two programs will be used in the thesis: *LS-DYNA*, an FEM solver with user written code for the grain structure development, and *MatCalc*, a precipitation kinetics calculator based on the CALPHAD approach. The FEM divides the simulated part into thousands of elements, and calling a separate instance (i.e. *MatCalc* or any other relevant software) for each of these elements is simply not feasible, as it exceeds any realistic computational capacity. The coupling algorithm should address this issue and also be general enough that it can be used for similar problems in multiscale modelling.

The present thesis is structured as follows. First, the theory will be explained in chapter 2. It concerns the microstructure evolution during metal working and heat treatment as well as the modelling of precipitation and macroscopic material behaviour.
Afterwards, the experiments will be detailed in chapter 3. This includes the physical experiments as well as the simulation programs and their numerical implementation.
In chapter 4, the newly developed coupling algorithm and its implementation are explained. Then, the results together with a discussion of the experiments and the simulations are given in chapter 5.
Finally, a summary of this work is given together with an outlook of possible future challenges in Chapter 6.

For ease of writing and reading, the author has chosen to employ the pronoun "we" and write the subsequent text in first person plural.

Chapter 2

Theory

The models used in this thesis are based on three theoretical fields, namely the theory of microstructure evolution, the theory of solid state precipitation in metals and a theory for numerical calculation of macroscopic materials behaviour, called the Finite Element Method (FEM). These theoretical fields are combined in a multiscale modelling approach, where different physical scales are modelled and then linked to obtain an accurate description of simultaneous microstructure evolution and precipitation kinetics during thermo-mechanical processing of metals such a hot deformation and heat treatment.

In this chapter, we will discuss the three different theories. First we will talk in depth about the microstructure of metals, and we will use the resulting equations to understand the MD^{2}M material model. Afterwards we discuss precipitation dynamics, and finally the FEM. At the end of each section, there is a list of the variables used for ease of reference.

2.1 Microstructure

While the chemical composition of a metal is important for many different material properties such as density or heat conductivity, it is not the only relevant parameter. For a long time, blacksmiths and other artisans have noticed that the previous manufacturing process plays a crucial role for the hardness and formability of a metal. Chemically identical materials can have vastly different properties, for example when they are hardened through cold working. These differences in mechanical behaviour can be explained when we look at the microscopical structure. The atoms within a metal are ordered in a crystal lattice, and defects or changes in this ordering can have a huge impact on macroscopic parameters, such as the yield strength.

The following theories comprise the MD^{2}M model. Its numerical implementation is detailed for future reference in the appendix.

2.1.1 What is Microstructure

Overview The *microstructure* of a metal is its microscopical structure. It entails the periodic lattice and all kinds of defects within this lattice. A defect is anything that breaks the periodicity, and examples are vacancies, substitutional atoms, mismatch in lattice structure or pockets of differing phases. Some examples of defects are shown in figure 2.1. These defects can be ordered by their dimensionality. Point defects consist of vacancies and substitutional atoms. Line defects are so called dislocations, and they are mismatch lines running through the crystal. Surface defects are interfaces between different phases or lattice structures. Finally, volume defects encompass a wide range of different things, as any three-dimensional abnormal structure within the crystal can be classified as such. A prominent example of a volume defect are the precipitates, which are explained in more depth in section 2.2.
Out of all these different structures, we will now take a closer look at *dislocations* and afterwards at *grains* and their boundaries. Dislocations are mismatch lines and as such a one-dimensional defect. Grains are the small crystals that make up the polycristalline metal, and their interfaces, or *grain boundaries*, are two-dimensional defects of great importance.

The following section is based on [3] and [12].

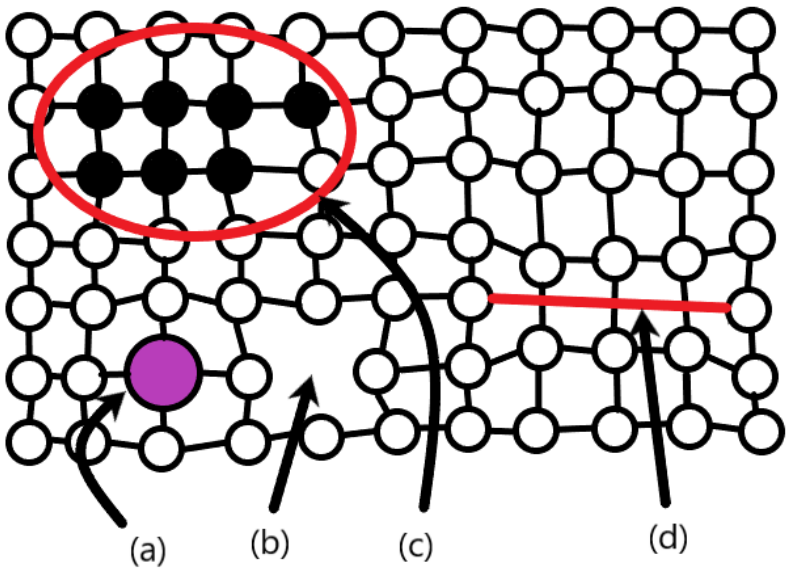

Fig. 2.1: Examples of crystal defects within a two-dimensional lattice: (a) substitutional atom; (b) vacancy; (c) different chemical composition; (d) dislocation line

Dislocations We imagine stacking a crystal row by row, but we make a mistake and skip one row. The lattice above the missing row now has to distort to fit the reduced number of rows, and this phenomenon is called a *dislocation*. Depending on which row was skipped, the dislocation is either an *edge dislocation* or a *screw dislocation*.[1] Figure 2.2 gives an example of both. Dislocations can be described by their Burgers vector, which is commonly denoted as b.[2] This vector is obtained by going around the crystal lattice in a closed loop within the undistorted crystal and then within the distorted crystal. Because of the distortion, the endpoint within the distorted crystal will not be identical to the starting point, and the difference between the two points is the Burgers vector. This can also be seen in figure 2.2. The dislocation extends as a line along a crystal plane until it ends in a point defect like a vacancy or a substitutional atom. Dislocations can move within a crystal, they can be annihilated when encountering a dislocation in the opposite direction and they can be created when stress is applied to the crystal.
A common parameter to measure the "amount" of dislocations within a crystal is the *dislocation density* ρ. It can be understood in one of two equivalent ways. We imagine a cube of metal with volume V. Within this volume there is a certain length L_{disl} of dislocations, which we can imagine like different threads running through the cube. We then cut the cube along a surface of size A and count the number of times the dislocations run "through" the surface, given by N_{disl}. The dislocation density is then given by:

$$\rho = \frac{L_{disl}}{V} = \frac{N_{disl}}{A} \tag{2.1}$$

We see that the dislocation density is of dimension $[m]^{-2}$ and it describes the length of dislocations per unit volume or equivalently the number of times that dislocations pass through a unit surface within the metal.

Grains and Subgrains Almost all metals are polycrystals, which means they don't consist of one single crystal but instead many small crystals together. Figure 2.3 shows a 2-dimensional example. The interfaces between these small crystals are two-dimensional defects and they are called grain boundaries. Each crystal is accordingly called a grain. Grain size, shape and lattice orientation have a big influence on the mechanical behaviour of the metal, such as deformation or breaking. Within a crystal, there can be smaller grains, so-called subgrains. The difference between the subgrains and the grains is the lattice misorientation at their boundaries, given by their relative change in lattice angle.

[1] Actually, most real dislocations are a mixture of both, but the local displacement can be described by either an edge or screw form.
[2] Although the Burgers vector is, as the name suggests, a vector, we will only be using its length $|b|$ and therefore take $b \sim |b|$ as its magnitude.

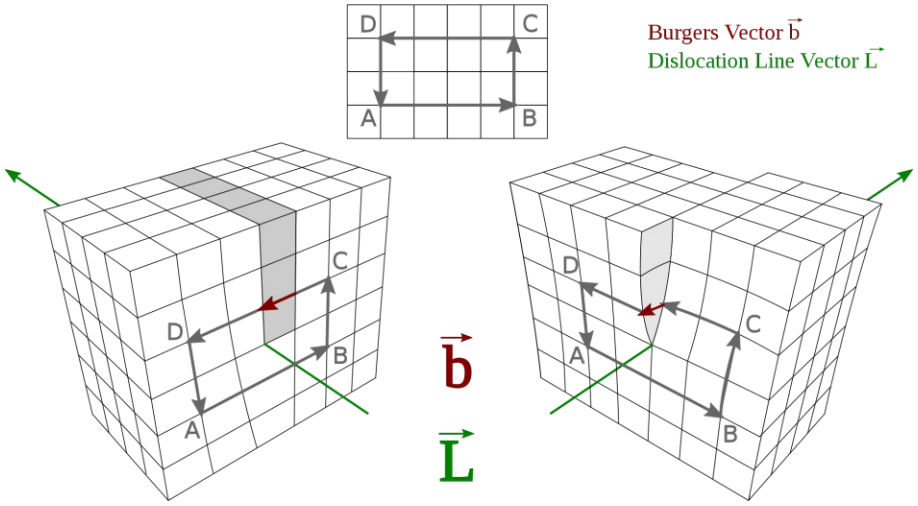

Fig. 2.2: Schematic overview of edge dislocations (left) and screw dislocations (right). The Burgers vector is indicated by b, while the direction of the dislocation line is indicated by L.
(Source: [10], *licensed under Creative Commons Attribution-Share Alike 4.0 International license*)

Subgrain boundaries have a small change in angle (commonly $\lesssim 15°$), whereas grains have a bigger change. Figure 2.4 shows this difference.
At first, the distinction between grains and subgrains seems arbitrary, but it becomes clear as soon as we take a closer look at their boundaries. For smaller boundary angles, both crystal lattices are still relatively coherent and the boundary is nothing more than a series of dislocation lines at regular intervals. At larger boundary angles, the interface becomes incoherent and thus its properties become independent of the relative orientation. This leads to a different boundary energy and boundary behaviour, as grains are more "sharply" separated than subgrains.
Two important parameters associated with grains and subgrains are the *grain size* δ^G and the *subgrain size* δ^{sub}. They correspond to the average diameter of the grains and subgrains respectively, and they are important variables of microstructure calculations.

Influence on Mechanical Properties Plastic deformation is enabled at an atomic level by the generation and movement of dislocations. It is easy to imagine that a lot of microscopic dislocations stack up to a macroscopic bending of the metal. Their mobility is thus an important indicator for the strength of the material. It can be influenced by precipitates that impede the movement of dislocations, which is why in general impurities increase strength. However they can also be blocked by subgrain and grain boundaries, meaning that metals with smaller grain sizes will be harder than the same metals with

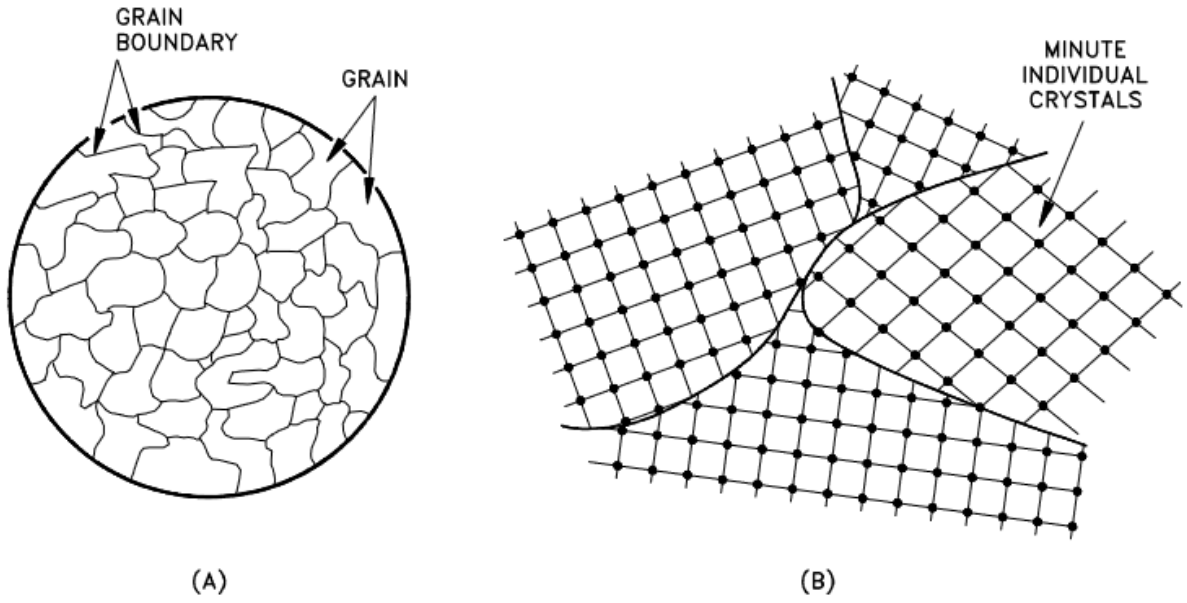

Fig. 2.3: Two-dimensional explanation of grain structure; (A) overall distribution of grains and grain boundaries within the metal; (B) zoomed in view of different crystal structures and grain boundaries. (Source: [7], with permission of the US Department of Energy)

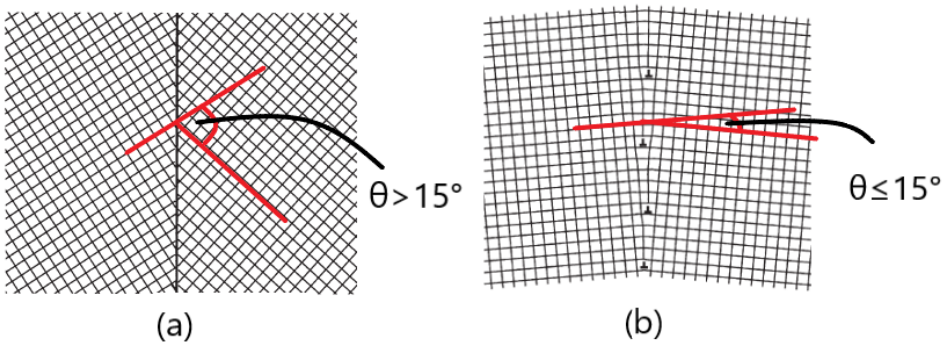

Fig. 2.4: Examples for crystal boundaries; (a) grain boundary, there is no coherence along the boundary, and the boundary angle $\theta > 15°$; (b) subgrain boundary, the crystal is distorted but coherent, and the boundary angle $\theta \leq 15°$; the small inverted T's indicate dislocation lines running perpendicular to the picture. (Source: [12], with permission of Elsevier)

larger grains. This effect is called the Hall-Petch strengthening.
The Hall-Petch relation is given by the following equation (see [18]):

$$\sigma_F = \sigma_0 + \alpha_1 Gb\sqrt{\rho} + \alpha_2 Gb(\frac{1}{\delta^{sub}} + \frac{1}{\delta^G}) \tag{2.2}$$

Here, σ_F is the flow stress (or sometimes called yield stress) above which the material plastically deforms. Equation 2.2 tells us that σ_F depends on some σ_0 unrelated to dislocations, subgrains and grains, on the shear modulus G, the Burgers vector b, the dislocation density ρ, the subgrain size δ^{sub} and the grain size δ^G. The parameters α_1 and α_2 are fitting parameters. The significance of the flow stress σ_F is further elaborated in subsection 2.3.2.
We see that a high dislocation density ρ and a small subgrain size δ^{sub} and grain size δ^G lead to a high deformation resistance, which is consistent with our previous explanations.

2.1.2 Work Hardening and Recovery during Metal Working

Overview During metal working like forging or rolling, the strain causes dislocations to be created and aggregate into subgrains. However there are also recovery processes that counter the work hardening. In this subsection, we will derive equations for the different mechanisms that govern the dislocation density within a worked metal.

The following subsection shows a slight modification of the model used in [13].

Hardening When a metal is deformed under work, the deformation often corresponds to a creation of dislocations to accommodate the plastic strain. These dislocations move until they get stuck with other dislocations to form and expand subgrain or grain boundaries. This, in turn, results in a hardening of the material as described in equation 2.2.
As stated in [13], the increase in dislocation density due to hardening $d\rho_{hard}$ can be described by the following equation:

$$d\rho_{hard} = \frac{f_T\sqrt{\rho}}{Ab} d\varepsilon_p \tag{2.3}$$

Here, f_T is the crystal dependent Taylor factor, ρ the dislocation density, b the Burgers vector, A a fitting parameter and $d\varepsilon_p$ the increase in plastic strain.

Glide Recovery When dislocations with opposite Burgers vectors meet, they will annihilate and bring the crystal to a more ordered state. This process is called recovery, and it is one of the major forces opposing the work hardening described above. One mechanism for

recovery is the so called *dislocation glide*. The mechanical strain of the deformation causes the existing dislocations to glide along crystal planes through the metal. Subsequently, when two dislocations with opposite Burgers vector come within an annihilation distance d_{ann}, they cancel each other out.
According to [13], the decrease of dislocation density due to glide recovery $d\rho_{glide}$ is given by:

$$d\rho_{glide} = -B\frac{2d_{ann}f_T}{b}\rho d\varepsilon_p \tag{2.4}$$

It is dependent on a fitting parameter B, the annihilation distance d_{ann}, the Taylor factor f_T, the Burgers vector b, the dislocation density ρ itself and the change in plastic strain $d\varepsilon_p$.
According to [13], the annihilation distance can be expressed as:

$$d_{ann} = Gb^4\frac{1}{2\pi(1-\nu)Q_{vac}} \tag{2.5}$$

Here, G is the shear modulus, b is again the Burgers vector, ν is the Poisson number of the material and Q_{vac} is the vacancy formation energy of the material.

Climb Recovery Another big factor in recovery is the so-called *dislocation climb*. In this process, dislocations at subgrain and grain boundaries "climb" from one level of the lattice to the next through movement of vacancies. In contrast to the hardening and the glide mechanisms, which are activated through plastic strain, the climb mechanism is activated by thermal movement of vacancies within the boundary region. After a dislocation has climbed, there again exists the possibility for it to interact with a dislocation with opposite Burgers vector and to annihilate.
As described in [13], the decrease in dislocation density due to climb $d\rho_{climb}$ is given by:

$$d\rho_{climb} = -2C \cdot D\frac{Gb^3}{k_BT}(\rho^2 - \rho_{eq}^2)dt \tag{2.6}$$

It is dependant on the diffusion coefficient of vacancies D, the shear modulus G, the average length of the Burgers vector b, the temperature T, the dislocation density ρ itself, the equilibrium dislocation density ρ_{eq} and the time increment dt. Finally, the parameter C is a fitting parameter.
We remark that climb recovery is thermally activated, and that there is climb for systems with $d\varepsilon_p = 0$. The equilibrium dislocation density ρ_{eq} is due to the fact that many metals retain some dislocation density even when there is no change in plastic strain ε.

Influence of Precipitates In standard literature, the diffusion coefficient D in equation 2.6 is given by (see [13]):

$$D = \nu_D b^2 \exp\left(\frac{-Q_D}{k_B T}\right) \tag{2.7}$$

Here, ν_D is the Debye frequency of the metal atoms, b is the Burgers vector, Q_D is the diffusion activation energy and T the temperature. The diffusion coefficient can, however, be influenced by solute atoms. We modify equation 2.7 in accordance with [18] and [17]:

$$D = \chi \cdot \nu_D b^2 \exp\left(\frac{-Q_D}{k_B T}\right) \tag{2.8}$$

Since the solute atom concentrations are closely linked to precipitation, we call χ the *precipitation parameter*. It is given by:

$$\chi = \left(\sum_{k=1}^{m} c_k\right)^{-1} \tag{2.9}$$

Here, c_k is the concentration of the k-th solute atom, and their dynamics are explained in section 2.2. We see that an increase in solute atom concentration leads to a decrease in diffusion, which fits the real world application.

Work Hardening Model All previously discussed effects together provide a work hardening equation, which describes the change in dislocation density under metal working. The equation consists of the hardening term $d\rho_{hard}$, the glide term $d\rho_{glide}$ and the climb term $d\rho_{climb}$, and it is given by adding equations 2.3, 2.4 and 2.6 to get:

$$d\rho = d\rho_{hard} + d\rho_{glide} + d\rho_{climb} =$$

$$= \frac{f_T\sqrt{\rho}}{Ab} d\varepsilon_p - B\frac{2d_{ann}f_T}{b}\rho d\varepsilon_p - 2C \cdot D\frac{Gb^2}{k_B T}(\rho^2 - \rho_{eq}^2)dt \tag{2.10}$$

Relationship between Dislocations and Subgrains During the deformation, the newly created dislocations will form subgrains within the grains. As described above for the hardening, almost all dislocations become stuck shortly after being created. This leads to an aggregation of dislocations in certain parts of the crystal and, eventually, these aggregations become subgrain boundaries with the subgrains between them. A commonly used (see [19]) formula to describe this process is given by:

$$\delta^{sub} = \frac{K}{\sqrt{\rho}} \tag{2.11}$$

Here, δ^{sub} is the subgrain size, ρ is the dislocation density and K is a fitting parameter. The subgrain size is relevant for determining the flow stress in equation 2.2 will become an important parameter for recovery during the heat treatment.

2.1.3 Recovery and Grain Growth during Heat Treatment

Overview There are three main processes going on with the microstructure during a heat treatment. They are *grain growth*, *recovery* and *recrystallization*. In this subsection, we will focus on grain growth and recovery as we can use the same theoretical approach to describe both. Recrystallization will be the subject of subsection 2.1.4.
We have already talked about recovery in subsection 2.1.2. The annihilation of dislocations corresponds to a growth of the subgrains, as the dislocations are mostly stored in their boundaries and bigger subgrains correspond to less boundary surface, which in turn requires less dislocations. Grain growth works similarly, as it is the thermally activated growth of grains to reduce the energy of grain boundaries. Microscopically, recovery and grain growth are different processes because subgrain boundaries and grain boundaries have different properties. However, they are similar in that a (sub)grain boundary is moving because of some driving force. A schematic description of the processes, grain growth, recovery and recrystallization, is given in figure 2.5.
From now on, we will shift our point of view of the microstructure. Beforehand, we took dislocations and their density ρ as the fundamental parameter, as shown in equation 2.10, and we calculated the subgrain size δ^{sub} accordingly, as shown in equation 2.11. Now, we will turn this relationship around and take a look at the dynamics of the subgrain size δ^{sub} and grain size δ^{G}. After that we can compute the dislocation density ρ by inverting equation 2.11.

The following subsection is based on different chapters of [12] to put together a theory of grain growth and recovery during heat treatment.

Grain Boundary Migration A very general approach to describe the movement of subgrains and grains is given on page 123 of [12]:

$$v = MP \tag{2.12}$$

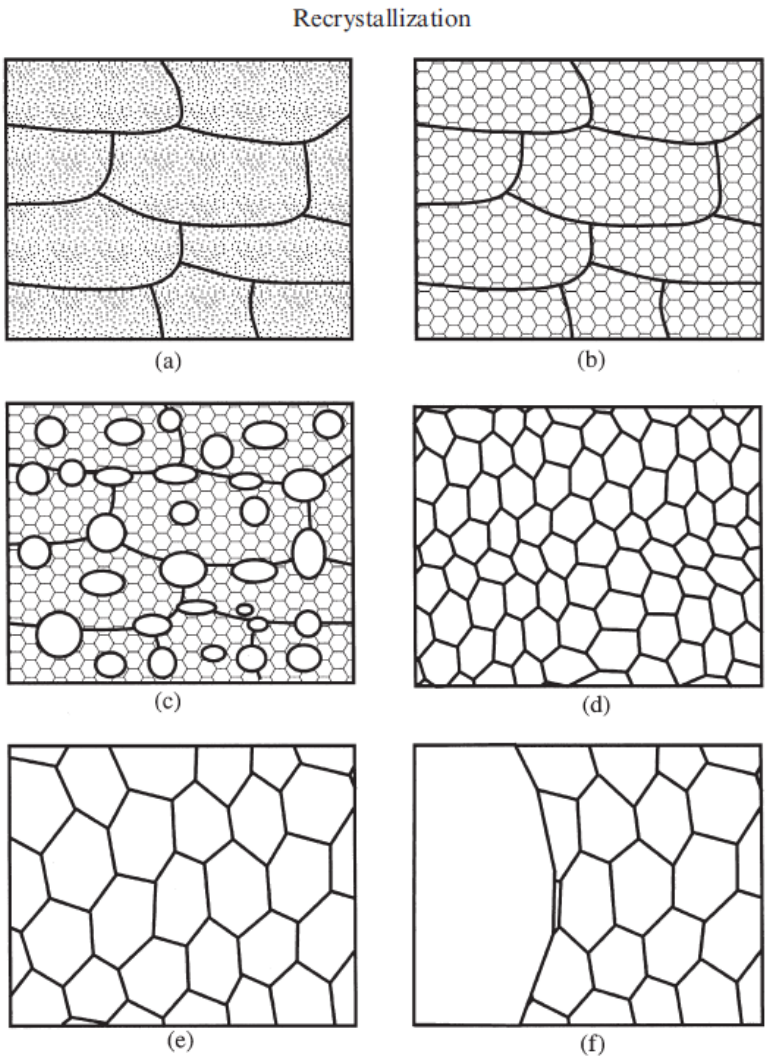

Fig. 2.5: Schematic diagram of recovery and recrystallization; (a) deformed state, (b) recovered grains with subgrains inside, (c) partially recrystallized, (d) fully recrystallized, (e) grain growth and (f) abnormal grain growth (not covered in this thesis). (Source: [12], with permission of Elsevier)

Here, v gives the migration speed of the boundary, M is the *grain boundary mobility*, and P is the *driving force*[3]. The variable v can now be roughly identified as either the growth of the subgrain size $\frac{d\delta^{sub}}{dt}$ or growth of the grain size $\frac{d\delta^{G}}{dt}$. This very general equation hides the complex processes of grain growth and recovery in M and P, but it suits our needs to describe them in a unified format.

The mobility of grain boundaries M_{GB} is given according to page 131 of [12] by the following equation:

$$M_{GB} = C_M \frac{Db}{k_B T} \tag{2.13}$$

The relevant parameters are a dimensionless fitting parameter C_M, the diffusion coefficient D for the material, the Burgers vector b and the temperature T. This equation incorporates the fact that thermally activated diffusion processes control the speed of migration of grain boundaries through the dependence of M_{GB} on D and T.

Measurements have shown (see page 130 of [12]) that the mobility of subgrain boundaries corresponds roughly linearly to their misorientation angle θ. In turn, this misorientation angle corresponds roughly linearly to the boundary surface energy γ_{sub} of the subgrains (see page 95 of [12]). Thus we make the assumption, that the subgrain mobility M_{sub} grows linearly with the subgrain boundary energy until it reaches the maximal value of the grain boundary mobility M_{GB}. This assumption can be expressed by the following equation:

$$M_{sub} = M_{GB} \frac{\gamma_{sub}}{\gamma_{GB}} \tag{2.14}$$

Here, γ_{GB} is the surface energy of the grain boundaries. This chain of linear relations helps us to combine the mobility of subgrain boundaries M_{sub} and grain boundaries M_{GB} into one model.

Zener Drag Many metals contain impurities. These impurities can lead to small aggregates within the crystal structure called precipitates, which have a different phase and/or chemical composition. Their dynamics are further discussed in section 2.2. For now, we will just take a look at how subgrains and grains interact with these precipitates. As it turns out, they slow down the grain growth and recovery, because they inhibit subgrain boundaries and grain boundaries from moving over them. This effect is known as *Zener pinning* or *Zener drag*, and it reduces the driving pressure P in equation 2.12. A schematic

[3] Although it is called a *force*, P is actually given as $[J][m]^{-3}$ and denotes the pressure on the grain boundary.

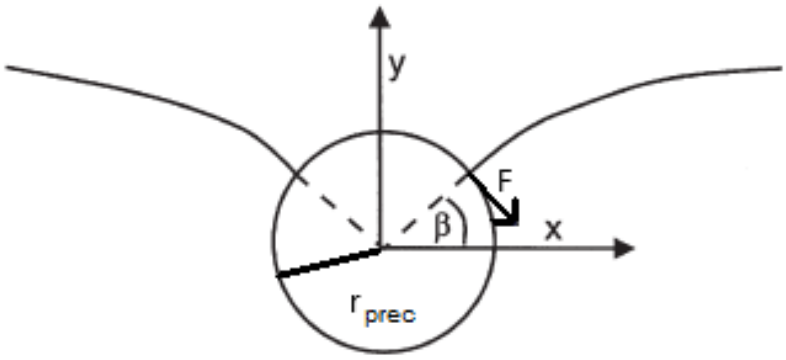

Fig. 2.6: This figure shows the interaction between a subgrain or grain boundary and a spherical particle. The precipitate has a radius r_{pr}, intersects the boundary at an angle β and a force F acts upon the grain boundary and prevents it from passing over the particle. (Source: [12], with permission of Elsevier)

explanation is given in figure 2.6.
The Zener drag P_Z is given by the following equations (see chapter 4.6 of [12]):

$$P_{Z_{sub}} = 2N_{pr}\pi\gamma_{sub}r_{pr}^2$$

$$P_{Z_{GB}} = 2N_{pr}\pi\gamma_{GB}r_{pr}^2 \tag{2.15}$$

Here, N_{pr} is the precipitate density, γ_{sub} and γ_{GB} are the boundary energies of the subgrain and grain boundary, and r_{pr} is the precipitate radius.
The Zener drag is the main antagonist of recovery and recrystallization, and it links the dynamics of subgrains, grains and dislocations to the dynamics of precipitates. The behaviour of N_{pr} and r_{pr} can be calculated with the theories described in section 2.2.

Grain Growth Grains grow naturally over time, as larger grains mean less grain boundaries, which in turn means less energy stored in the crystal. This process works by small grains shrinking and large grains growing even larger, and it is detailed in subfigures (d) and (e) of figure 2.5. We will describe the grain growth through the evolution of the (mean) grain size δ^G.
We have already established the grain boundary mobility M_{GB} in equation 2.13, and so it remains to calculate the driving force for grain growth P_G. In chapter 11 from [12], we get the following expression:

$$P_G = \frac{\alpha\gamma_{GB}}{R^G} = \frac{3\gamma_{GB}}{\delta^G} \tag{2.16}$$

Here, γ_G is the grain boundary energy, R^G the average grain radius and $\alpha \sim 3/2$ a geometrical coefficient. Since δ^G is the grain diameter, we also substitute $2R^G = \delta^G$ to get

the second equation.

We can now plug equation 2.16 into equation 2.12 and remember that the Zener drag $P_{Z_{GB}}$ reduces the driving force to $P_G - P_{Z_{GB}}$. The velocity v can be interpreted as $2v = 2\frac{dR^G}{dt} = \frac{d\delta^G}{dt}$. Altogether, this gives the following expression:

$$d\delta^G = M_{GB}\left(\frac{6\gamma_{GB}}{\delta^G} - 2P_{Z_{GB}}\right)dt$$

$$d\delta^G = 0 \quad \text{if} \quad \left(\frac{6\gamma_{GB}}{\delta^G} - 2P_{Z_{GB}}\right) < 0 \tag{2.17}$$

The second equation is due to the fact that the Zener drag P_Z is only a dragging force. It pins the grain boundary in place at the precipitates, but it doesn't cause the grains to shrink.

Recovery As discussed above, we know that recovery leads to a movement of subgrain boundaries and an increase of the subgrain size δ^{sub}. Therefore we can model it the same way as we did the grain size δ^G, as evidenced in chapter 6 of [12]. Recovery is shown in subfigures (a) and (b) of figure 2.5.

The mobility M_{sub} is already given from equation 2.14, so we only need to calculate the driving force P_{sub}. Similarly to equation 2.16, the following expression gives the driving force for subgrain growth:

$$P_{sub} = \frac{3\gamma_{sub}}{\delta^{sub}} \tag{2.18}$$

Here, γ_{sub} is the boundary energy of subgrain boundaries and δ^{sub} is the subgrain size. However, in reality we observe that metals retain some dislocation density even after recovery. We already incorporated this fact in equation 2.6, and we will do the same here by introducing the equilibrium subgrain size δ^{sub}_{eq}. This leads to:

$$P_{sub} = \frac{3\gamma_{sub}}{\delta^{sub}} - \frac{3\gamma_{sub}}{\delta^{sub}_{eq}} \tag{2.19}$$

Finally, we follow the same line of reasoning as for equations 2.17 to arrive at:

$$d\delta^{sub} = M_{sub}\left(\frac{6\gamma_{sub}}{\delta^{sub}} - \frac{6\gamma_{sub}}{\delta^{sub}_{eq}} - 2P_{Z_{sub}}\right)dt$$

$$d\delta^{sub} = 0 \quad \text{if} \quad \left(\frac{6\gamma_{sub}}{\delta^{sub}} - \frac{6\gamma_{sub}}{\delta^{sub}_{eq}} - 2P_{Z_{sub}}\right) < 0 \tag{2.20}$$

Again, the second equation is due to the fact that the Zener drag P_Z is only a dragging force.

2.1.4 Recrystallization during Heat Treatment

Overview Recrystallization is another important process during heat treatment. It describes the sudden change in crystal structure from dislocation-rich crystal to almost perfect crystal. The driving force for this sudden change comes from the deformation, where the energy is stored through increase of dislocations and decrease in subgrain size. When this stored energy inside the subgrain dislocations is above a certain threshold, some subgrains can spontaneously reform themselves into almost perfect crystals with drastically reduced dislocation density. These new crystals form new grains and they will grow as long as thermally possible or until they encounter other recrystallized subgrains and the whole metal is recrystallized.
A schematic description of recrystallization is given by figure 2.5 in steps (c) and (d).

The approach described in the following section is heavily based on chapter 7 of [12], but it also takes inspiration from [23].

Driving Force for Recrystallization We first look into the requirements for recrystallization. Equation 2.12 will once again be our starting point, since recrystallization also involves grain boundaries moving through the crystal. The newly formed structures are grains and therefore have high angle boundaries, which means we can use M_{GB} from equation 2.13. The driving force P^{rex} comprises of three terms:

$$P^{rex} = P^{rex}_{\rho} + P^{rex}_{\delta^{sub}} + P^{rex}_{\delta^{rex}} \tag{2.21}$$

These terms are due to the dislocation density (P^{rex}_{ρ}), the reduction of subgrain boundaries ($P^{rex}_{\delta^{sub}}$) and the surface energy of the recrystallized grain ($P^{rex}_{\delta^{rex}}$).
We see from page 18 of [12] that the excess energy density e_ρ in the unrecrystallized material is given by:

$$e_\rho = C_\rho \rho G b^2 \tag{2.22}$$

Here, C_ρ is a fitting parameter, G is the shear modulus and b is the Burgers vector. This energy creates a driving force P^{rex}_{ρ}, and if we assume this pressure to be equal in all directions, we get:

$$P^{rex}_{\rho} = e_\rho = C_\rho(\rho - \rho_{eq})Gb^2 \tag{2.23}$$

The additional variable ρ_{eq} denotes the equilibrium dislocation density, and we already discussed its significance for equations 2.6 and 2.19.
When the recrystallized grain grows, it pushes back the other subgrains. This effect leads to another driving force for recrystallized grains which we denote by $P^{rex}_{\delta^{sub}}$. The driving

force acts in a similar way as the driving force for subgrain growth in equation 2.19 in that it stems from smaller subgrains shrinking and larger subgrains growing. This prompts us to take inspiration form [23] and express $P^{rex}_{\delta^{sub}}$ by a similar term:

$$P^{rex}_{\delta^{sub}} = \frac{3\gamma_{sub}}{\delta^{sub}} \tag{2.24}$$

Here, γ_{sub} denotes the boundary energy of the subgrain boundary and δ^{sub} is the subgrain size.

When recrystallized grains with grain size δ^{rex} grow, they also increase the boundary energy which is on average given by $E_{\delta^{rex}} = \gamma_{GB}\pi(\delta^{rex})^2$ with γ_{GB} being the boundary energy of grain boundaries. From this term we can calculate the pressure on the surface of a recrystallized grain and we get for the driving force $P^{rex}_{\delta^{rex}}$[4]:

$$P^{rex}_{\delta^{rex}} = -\frac{2}{A_{\delta^{rex}}}\frac{dE_{\delta^{rex}}}{d\delta^{rex}} = -\frac{2}{\pi(\delta^{rex})^2}\frac{d}{d\delta^{rex}}\left(\gamma_{GB}\pi(\delta^{rex})^2\right) = -\frac{4\gamma_{GB}}{\delta^{rex}} \tag{2.25}$$

Here, $A_{\delta^{rex}}$ is the surface of a recrystallized grain with diameter δ^{rex}.

We can now insert the equations 2.23, 2.24 and 2.25 in equation 2.21 to arrive at the following expression for the driving force P^{rex} for recrystallized grain growth:

$$\begin{aligned} P^{rex} &= P^{rex}_{\rho} + P^{rex}_{\delta^{sub}} + P^{rex}_{\delta^{rex}} \\ &= C_{\rho}(\rho - \rho_{eq})Gb^2 + \frac{3\gamma_{sub}}{\delta^{sub}} - \frac{4\gamma_{GB}}{\delta^{rex}} \end{aligned} \tag{2.26}$$

For very small grains, the last term dominates and prevents recrystallization. However as soon as it has reached a critical size δ^{rex}_{crit}, the positive contributions dominate and it grows into a recrystallized subgrain. We take into account the Zener drag $P_{Z_{GB}}$ that also acts on recrystallized grains and set $P^{rex} - P_{Z_{GB}} = 0$ to determine this critical recrystallization size δ^{rex}_{crit}:

$$\delta^{rex}_{crit} = \frac{4\gamma_{GB}}{P^{rex}_{\rho} + P^{rex}_{\delta^{sub}} - P_{Z_{GB}}} \tag{2.27}$$

Here, γ_{GB} is the boundary energy of grain boundaries, P^{rex}_{ρ} is the driving force created by the dislocations and $P^{rex}_{\delta^{sub}}$ is the driving force created by subgrain shrinking. We now have a threshold value for the start of recrystallization.

[4] The factor 2 in the following equation is due to the fact that we calculate with the grain size δ^{rex}, which is the average diameter and not the average radius of the recrystallized grains.

Since the contribution of the recrystallized boundary energy $P^{rex}_{\delta^{rex}}$ becomes irrelevant for sufficiently large recrystallized grains, we will ignore it from now on to get:

$$P^{rex} \sim C_\rho(\rho - \rho_{eq})Gb^2 + \frac{3\gamma_{sub}}{\delta^{sub}} \tag{2.28}$$

Nucleation for Recrystallization Not all subgrains immediately recrystallize once their size reaches δ^{rex}_{crit}, since δ^{sub} is just the *average* diameter of subgrains, so only the larger ones are eligible for recrystallization. Additionally, recrystallization is a thermally activated process, so there is some statistical variation for the number of grains recrystallizing. A simple model for these circumstances is given on page 231 of [12]. We assume a thermal distribution of potential nuclei and get the following set of equations:

$$\frac{dN^{rex}}{dt} = C^{rex} \exp\left(-\frac{Q^{rex}}{k_B T}\right) \quad \text{if} \quad \delta^{sub} \geq \delta^{rex}_{crit}$$

$$\frac{dN^{rex}}{dt} = 0 \quad \text{if} \quad \delta^{sub} < \delta^{rex}_{crit} \tag{2.29}$$

Here we have N^{rex} as the total number of nuclei per volume, C^{rex} as a fitting parameter[5], Q^{rex} as the activation energy for recrystallization, T as the temperature, δ^{sub} as the subgrain size and δ^{rex}_{crit} as the critical recrystallization size.

Recrystallized Fraction Recrystallization is often measured in terms of the recrystallized volume fraction X^{rex}. We calculate it by multiplying N^{rex} with the average volume of a recrystallized grain:

$$X^{rex} = N^{rex}\frac{\pi}{6}(\delta^{rex})^3 \tag{2.30}$$

However, there is one last adjustment we have to make to our theory. Recrystallization, be it through growth of recrystallized grains or through nucleation, can naturally only occur in a non-recrystallized volume. Therefore we must modify the growth equation 2.12 and the nucleation rate given by equation 2.29 by the factor $(1 - X^{rex})$, which represents the non-recrystallized volume fraction.

We now insert the driving force $P^{rex} - P_{Z_{GB}}$ given by equations 2.15 and 2.28 into equation 2.12 and take into account the critical recrystallized grain size δ^{rex}_{crit} and the factor $(1-X^{rex})$ to get the following evolution equations for the recrystallized grain size δ^{rex}:

[5] The fitting parameter can instead be taken as a dynamical variable proportional to $(\delta^G_0(\delta^{sub})^2)^{-1}$, as shown in [23]. This represents the nucleation at boundaries.

$$\left.\begin{aligned} d\delta^{rex} &= M_{GB}\left(P_{\rho}^{rex} + P_{\delta^{sub}}^{rex} - P_{Z_{GB}}\right)(1 - X^{rex})dt \\ d\delta^{rex} &= 0 \quad \text{if} \quad \left(P_{\rho}^{rex} + P_{\delta^{sub}}^{rex} - P_{Z_{GB}}\right) < 0 \end{aligned}\right\} \text{if} \quad \delta^{sub} \geq \delta_{crit}^{rex}$$

$$d\delta^{rex} = 0 \quad \text{if} \quad \delta^{sub} < \delta_{crit}^{rex} \tag{2.31}$$

Here, M_{GB} is the grain boundary mobility, P_{ρ}^{rex} is the dislocation dependent recrystallization driving force, $P_{\delta^{sub}}^{rex}$ is the subgrain shrinking dependent recrystallization driving force, $P_{Z_{GB}}$ is the Zener drag, X^{rex} is the recrystallized volume fraction, δ^{sub} is the subgrain size and δ_{crit}^{rex} is the critical recrystallization size.

We modify equation 2.29 in the same way and we get for the nucleation rate of recrystallization:

$$\frac{dN^{rex}}{dt} = C^{rex}(1 - X^{rex}) \exp\left(-\frac{Q^{rex}}{k_B T}\right) \quad \text{if} \quad \delta^{sub} \geq \delta_{crit}^{rex}$$

$$\frac{dN^{rex}}{dt} = 0 \quad \text{if} \quad \delta^{sub} < \delta_{crit}^{rex} \tag{2.32}$$

Here N^{rex} is the total density of nuclei, C^{rex} is a fitting parameter, Q^{rex} is the activation energy for recrystallization, T is the temperature, X^{rex} is the recrystallized volume fraction, δ^{sub} is the subgrain size and δ_{crit}^{rex} is the critical recrystallization size.

Together, equations 2.30, 2.31 and 2.32 cover the dynamics of recrystallization in our system by describing respectively the recrystallized fraction, the growth of recrystallized grains and the nucleation rate of recrystallized grains.

2.1.5 List of Symbols

A, B, C	fitting parameters for Work Hardening Model
C^{rex}	fitting parameter for recrystallization nucleation
C_M	grain boundary mobility fitting parameter
C_ρ	fitting parameter for dislocation energy density
b	Burgers vector (magnitude)
d_{ann}	annihilation distance of dislocations
D	diffusion coefficient
f_T	crystal Taylor factor
G	shear modulus
k_B	Boltzmann constant
K	fitting parameter for subgrain-dislocation relationship
L_{disl}	length of dislocations within unit volume
M_{sub}	subgrain boundary mobility
M_{GB}	grain boundary mobility
N_{disl}	number of dislocations crossing unit surface
N_{pr}	number of precipitates per unit volume
N^{rex}	number of recrystallization nuclei per unit volume
P_{sub}	subgrain size driving force
P_G	grain size driving force
P^{rex}	recrystallized grain driving force
P_Z	Zener drag
$P_{Z_{GB}}$	Zener drag on grain boundaries
$P_{Z_{sub}}$	Zener drag on subgrain boundaries
$P^{rex}_{\delta^{rex}}$	recrystallized grain driving force from recrystallized grain size
$P^{rex}_{\delta^{sub}}$	recrystallized grain driving force from dislocation density
P^{rex}_{ρ}	recrystallized grain driving force from dislocation density
Q_D	diffusion activation energy
Q_{vac}	vacancy formation energy
Q^{rex}	recrystallization nucleation energy
r_{pr}	precipitate radius
dt	time increment
T	Temperature
X^{rex}	recrystallized volume fraction

χ	precipitation parameter
δ^G	grain size
δ^{sub}	subgrain size
δ^{sub}_{eq}	equilibrium subgrain size
δ^{rex}	recrystallized grain size
δ^{rex}_{crit}	critical recrystallization size
ϵ_p	plastic strain
γ_{GB}	grain boundary surface energy
γ_{sub}	subgrain boundary surface enrgy
ν	Poisson number
ν_D	Debye frequency of the metal atoms
ρ	dislocation density
ρ_{eq}	equilibrium dislocation density
ρ_{mobile}	density of mobile dislocations
$d\rho_{climb}$	dislocation density decrease through climb
$d\rho_{glide}$	dislocation density decrease through glide
$d\rho_{hard}$	dislocation density increase through work hardening
σ_F	flow stress
σ_0	microstructure independent flow stress

2.2 Precipitation Kinetics

Within a material with solute elements, these elements sometimes tend to coalesce into small nuclei instead of being homogeneously distributed throughout the material. The nuclei may contain different concentrations of elements or consist of a different phase than the surrounding parent phase. This process is called *precipitation*.
Precipitation occurs if the energy of these small nuclei is lower than the energy of the parent phase which they replace. When this is the case, the solute atoms can spontaneously form such nuclei with thermal fluctuations. These nuclei can then grow through diffusion from the parent phase. In the end, the homogeneous material is replaced by a material with a slightly different chemical composition containing small precipitates of a different phase and composition. This means that the overall amount of each element has stayed the same, but through the formation of precipitates the energy was reduced by spatially clustering some of these elements. The process is schematically shown in figure 2.7.

The following sections are based on [14].

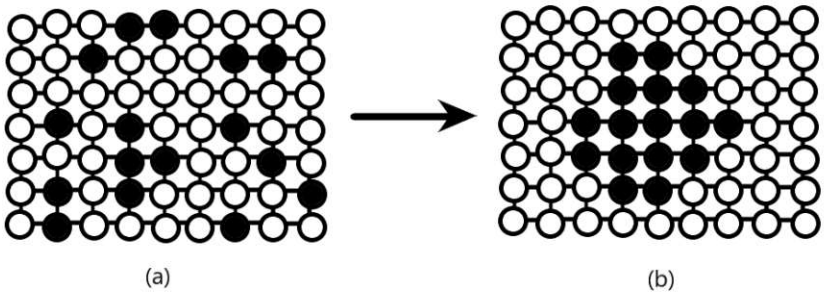

Fig. 2.7: A two dimensional lattice with two types of atoms (black & white). Their distribution is shown (a) before and (b) after precipitation.

2.2.1 Classical Nucleation Theory

Overview A useful method for calculating the amount of precipitate nuclei is Classical Nucleation Theory (CNT). It couples the creation of nuclei to the chemical composition, the temperature and the free energy of the precipitates, and it also helps estimate the influence of grains and dislocations on nucleation.

The following discussion is based on chapter 2 of [14].

Steady-State Flux The first point of interest is to determine the rate of newly created nuclei in a steady-state situation. We start by considering a simple super-saturated system consisting of two elements. The concentration of the minority element is above the saturation point, and thus little nuclei start to coalesce. We can model this process by looking at nuclei O_n of size n with corresponding surface area A_n. The nucleus can grow by attaching one element and it shrinks by detaching one element. We take the attachment rate per surface area to be a a_0, and the detachment rate per surface area to be q_n. This gives us the following relations:

$$\begin{aligned} O_n + O_1 &\to O_{n+1} \quad \text{at rate } = a_0 A_n \\ O_n - O_1 &\to O_{n-1} \quad \text{at rate } = q_n A_n \end{aligned} \tag{2.33}$$

This seemingly simple model can be made continuous with respect to the nucleus size n, so that it leads to a partial differential equation. With the assumption of a thermal equilibrium, this equation has a solution for the steady-state nucleus flux $j_s(n)$ of nuclei of size n, that is the number of nuclei going from size $n-1$ to n per differential timestep. Steady-state means that the flux remains constant over time. After reaching a critical size n^*, the nuclei don't follow a thermal distribution anymore but continue to grow on their own.[6] The flux at this critical size is given by the following equations:

$$\begin{aligned} j_s(n^*) &= Z\beta(n^*) \exp\left(-\frac{\Delta G_{nucl}(n^*)}{k_B T}\right) \\ Z &= \left[-\frac{1}{2\pi k_B T}\left(\frac{\partial^2 \Delta G_{nucl}}{\partial n^2}\right)_{n^*}\right]^{\frac{1}{2}} \end{aligned} \tag{2.34}$$

Here, Z is the *Zeldovich factor*, $\beta(n) = a_0 A_n$ is the attachment rate to a nucleus of size n and $\Delta G_{nucl}(n)$ is the nucleation energy of a nucleus of size n. Since nuclei of size n^* and larger grow into full precipitates, the flux $j_s(n^*)$ indicates the rate at which new precipitates are created within a steady-state system.

Time Lag After determining the steady-state flux, we can now look at the time it takes to reach the steady-state conditions for a system out of equilibrium. This time lag can by calculated via the following equation:

$$j = j_s \exp\left(-\frac{\tau}{t}\right)$$

[6] The reason for this will be given further down.

$$\tau = \frac{1}{4\pi\beta(n^*)Z^2} \tag{2.35}$$

Here, τ is called the *incubation time* and it measures how long a system takes to reach a steady-state nucleation.

The Nucleation Energy After describing the time dependent nucleation flux, we take a look at the free energy $\Delta G_{nucl}(n)$ of a nucleus of size n. As seen in equations 2.34, this energy is the major influence on the rate of nucleation and the nucleation size.
It has two important contributions, namely the volume energy density of the nucleus Δg_{vol} and the surface energy density of the interface of nucleus and the surrounding matrix given by Δg_{surf}. We now take $\frac{4\pi}{3}\Omega$ to be the volume of one element of the precipitate. Assuming spherical precipitates, this leads to the following equation for $\Delta G_{nucl}(n)$:

$$\Delta G_{nucl}(n) = \frac{4\pi}{3} n\Omega\Delta g_{vol} + 4\pi(n\Omega)^{\frac{2}{3}}\Delta g_{surf} \tag{2.36}$$

Here, the small g denotes the energy per volume and per surface area respectively. In the case of supersaturation and precipitation, we find that Δg_{vol} is negative. This explains the precipitation dynamics, since it is now energetically more favorable for precipitates to form. However, the surface term results in an energy barrier that the nucleus has to overcome first (see figure 2.8) with $\Delta G_{nucl}(n)$ having a maximum at some critical size n^*.

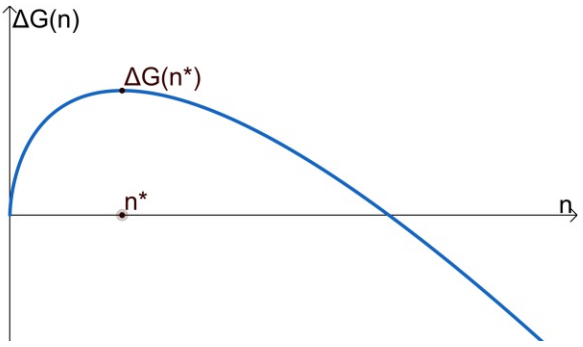

Fig. 2.8: Plot of the function $\Delta G_{nucl}(n)$ with arbitrary parameters Ω and $\Delta g_{vol} < 0 < \Delta g_{surf}$. The energy barrier for nucleation is given by $\Delta G_{nucl}(n^*)$ at the critical size n^*.

Heterogeneous Nucleation Just like small dust particles can help water in the air to form droplets and rain down, impurities and deformations within the crystal can enable nucleation. We multiply the steady-state flux in equation 2.34 by the number of potential

nucleation sites N to incorporate their influence. This together with equations 2.34 and 2.35 gives:

$$j(n^*) = NZ\beta(n^*) \exp\left(-\frac{\Delta G_{nucl}(n^*)}{k_B T}\right) \exp\left(-\frac{\tau}{t}\right) \tag{2.37}$$

We can now regard homogeneous nucleation and heterogeneous nucleation as two separate processes, each with their own N, ΔG_{nucl} and β.

The number of homogeneous nucleation N_{hom} sites is just the number of lattice atoms, and equation 2.36 gives a model for the nucleation energy. When looking at heterogeneous nucleation, we have two important nucleation sites: dislocations and grain boundaries. Both concepts are discussed in section 2.1.

First, we look at the number of nucleation sites. The dislocation density ρ is described in subsection 2.1.1 and gives the length of dislocations per volume. A line of dislocations with length L and interatomic distance a will have $\frac{L}{a}$ dislocation sites. Thus we can compute the number of dislocation nucleation sites per volume as:

$$N_{disl} = \frac{\rho}{a} \tag{2.38}$$

For a grain boundary area A within a unit volume and an interatomic distance a, we have approximately $\frac{A}{a^2}$ nucleation sites. Grains of grain size δ^G have a surface of $A = \Gamma_G \cdot x_G \cdot (\delta^G)^2$. Here, Γ_G is a shape factor dependant on the geometry of the grains[7] and x_G is the number of grains per unit volume. This gives the following number of nucleation sites at grain boundaries per volume:

$$N_{GB} = \Gamma_G x_G \frac{(\delta^G)^2}{a^2} \tag{2.39}$$

We now discuss the nucleation energies ΔG_{nucl} for heterogeneous nucleation, which has additional terms compared to equation 2.36. When a nucleus grows along a dislocation line or a grain boundary, it replaces these defects within the crystal and therefore reduces the excess energy from these defects. For dislocation lines with an energy per length γ_ρ, we find that a nucleus of size n with a diameter $d = f_\rho n^{\frac{1}{3}}$ thus effectively gains an additional nucleation energy term:

$$\Delta G_{disl}(n) = -f_\rho n^{\frac{1}{3}} \gamma_\rho \tag{2.40}$$

Similarly we can look at a nucleus along a grain boundary with diameter $d = f_{GB} n^{\frac{1}{3}}$. with grain boundary energy per area γ_{GB} we have an additional nucleation energy term:

$$\Delta G_{disl}(n) = -f_{GB}^2 n^{\frac{2}{3}} \gamma_{GB} \tag{2.41}$$

[7] Example geometries are a sphere or a tetrakaidekahedron.

In both cases of heterogeneous nucleation the additional energy is negative, since the deformations are removed by the growing nucleus.
The factors f_ρ and f_{GB} depend on the shape of the nuclei. Since the nucleus prefers to grow along the dislocations or grain boundaries, it becomes flattened in direction of the crystal defects.

Multi-component Nucleation Up until now we have only taken a two-element system into consideration. Because of the almost purely thermodynamical approach and the generalized expressions of the nucleation energy, we can come to the same conclusions for multi-component nucleation with different types of precipitates with potentially different crystal structures. The numerical values of ΔG_{nucl} change, but the general equations stay the same.

2.2.2 Evolution Equations for Precipitate Growth

Overview After obtaining a certain number of precipitate nuclei in a given system - for example through CNT - the issue of their time-evolution arises. This problem can be solved by using a thermodynamical approach, namely the Thermodynamic Extremal Principle (TEP).

The following discussion is based on chapter 3.5 of [14].

Thermodynamic Extremal Principle The evolution of the nuclei can be calculated with the TEP. We describe the system with the Gibbs energy G, which depends only on a set of state variables q_i such as the temperature, pressure and the chemical compositions and sizes of different nuclei, that means $G = G(q_i)$. The thermodynamic extremal principle now tells us that the time-evolution of G will go along a path where the energy dissipation $Q := -\dot{G}$ is maximised. In contrast to the Gibbs energy, the dissipation rate may also depend on the velocities $\dot{q}_i$, that is $Q = Q(q_i, \dot{q}_i)$. This gives the following set of equations:

$$Q = -\frac{dG}{dt} = -\sum_i \frac{\partial G}{\partial q_i}\dot{q}_i$$

$$\frac{\partial Q}{\partial q_i} = 0 \quad ; \quad \frac{\partial Q}{\partial \dot{q}_i} = 0 \tag{2.42}$$

A heuristic approach to understanding the thermodynamic extremal principle is as follows: For an isothermal and isobaric process, Q is equivalent to the change in entropy $\dot{S}$, which should be maximal.

This set of equations can be solved through method of Lagrange multipliers by solving the following equations for all $q_i, \dot{q}_i$:

$$\frac{\partial}{\partial q_i}\left[Q + \lambda\left(\frac{dG}{dt} + Q\right)\right] = 0$$

$$\frac{\partial}{\partial \dot{q}_i}\left[Q + \lambda\left(\frac{dG}{dt} + Q\right)\right] = 0 \tag{2.43}$$

Gibbs Energy We take a system of m precipitates and their surrounding matrix with volume V, made up of n different types of chemical elements. The total Gibbs energy is then given by three terms: the volume energy of the precipitates G_{vol}, the surface energy of the precipitates G_{surf} and the chemical energy of elements in the surrounding matrix G_{matrix}. We can write the equation as follows:

$$G = G_{vol} + G_{surf} + G_{matrix}$$

$$G = \sum_{k=1}^{m} \frac{4\pi r_k^3}{3}\left(\Delta g_{vol,k} + \sum_{i=1}^{n} c_{ki}\mu_{ki}\right) + \sum_{k=1}^{m} 4\pi r_k^2 \Delta g_{surf,k} + \sum_{i=1}^{n} V c_{0i}\mu_{0i} \tag{2.44}$$

The values $c_{1i}, ..., c_{mi}$ and $\mu_{1i}, ..., \mu_{mi}$ denote the concentrations and the chemical potential of element i in the precipitate $1, ..., m$, with c_{0i} and μ_{0i} denoting its concentration and chemical potential in the surrounding matrix. Each precipitate k has its own radius r_k, its volume energy $\Delta g_{vol,k}$ and its surface energy $\Delta g_{surf,k}$.
As an additional constraint for the Gibbs energy, we have mass conservation:

$$const = V c_{0_i} + \sum_{k=1}^{m} \frac{4\pi r_k^3}{3} c_{ki} \tag{2.45}$$

This equation gives an additional Lagrange multiplier for equations 2.43.

Dissipation Rate There are once again three terms which need to be considered for the energy dissipation Q: the interface migration (growth and shrinkage of the precipitate) with Q_{mig}, the diffusion inside of the precipitate with Q_{Dint} and the diffusion within the surrounding matrix with Q_{Dext}. Equations for these terms are rather complex and can be found in [14]. The important fact is that each depends only on chemical constants, the concentrations c_{ki}, the radii r_k and both their time derivatives $\dot{c}_{ki}$ and $\dot{r}_k$. Using these equations and setting c_{ki}, r_k as the state variables q_j, we can insert into equations 2.43 and explicitly calculate the precipitation evolution.

2.2.3 List of Symbols

a	interatomic distance
a_0	attachment rate to a nucleus per surface area
A_n	surface area of a nucleus of size n
c_i	concentration of element i
c_{0i}	concentration of element i in the surrounding matrix
c_{ki}	concentration of element i in the precipitate k
f_ρ	nucleus shape factor for nucleation along dislocations
f_{GB}	nucleus shape factor for nucleation along grain boundary
G	total Gibbs energy of the system
$\Delta g_{vol,k}$	free energy per volume of the precipitate k
Δg_{vol}	free energy per volume of a nucleus
Δg_{surf}	free energy per surface of a nucleus
$\Delta g_{surf,k}$	free energy per surface of the precipitate k
$\Delta G_{nucl}(n)$	nucleation energy for a nucleus of size n
j_s	steady-state flux of newly created nuclei
k_B	Boltzmann constant
n^*	critical nucleus size
N	number of nucleation sites
N_{disl}	number of nucleation sites at dislocations
N_{GB}	number of nucleation sites at grain boundaries
O_n	number of nuclei with size n
q_i	state variables for thermodynamical approach
q_n	detachment rate from a nucleus of size n per surface area
Q	energy dissipation of the system
Q_{mig}	interface migration dissipation
Q_{Dint}	energy dissipation due to diffusion inside of the precipitates
Q_{Dext}	energy dissipation due to diffusion in the matrix close to the precipitates
r_k	precipitate radius
t	time
T	temperature
V	volume of the material
x_G	number of grains per volume
Z	Zeldovich factor

$\beta(n)$	attachment factor for a nucleus of size n
δ^G	grain size
γ_ρ	energy per length of dislocation line
γ_{GB}	surface energy of grain boundary per area
Γ_G	grain surface geometry factor
λ	Lagrange multiplier
μ_{0i}	chemical potential of element element i in the surrounding matrix
μ_{ki}	chemical potential of element i in the precipitate k
τ	incubation time for a precipitation system to reach steady-state
ρ	dislocation density
Ω	volume factor of a single precipitate element

2.3 Macroscopic Materials Behaviour

Although many equations governing macroscopic material behaviour such as classical mechanics or heat conduction are straightforward, the complexity of problems often rules out an analytical solution by hand. The development of computers in the last century, however, has opened up a plethora of possibilities for physicists. Now, differential equations do not need to be solved analytically any more, since they can be approached numerically. Thus we will focus in this chapter on one such numerical algorithm named the *Finite Element Method (FEM)*. With it we will be able to calculate mechanical deformations, as well as thermal conduction within a solid body.

The subject of FEM is large enough to warrant a thesis of its own, so we will not go into too much detail. Our goal in this section will be to get an initial understanding of the theory and see how it is applied. Therefore it is important to note that the following section is by no means a rigorous mathematical derivation, but it should give a good understanding of the theoretical aspects.

2.3.1 Formulation of FEM

Overview We will now take a look at the mathematical foundation of the FEM approach. It is characterized by a more abstract way of tackling the problem, as it uses variational formulations instead of straightforward differential equations. Further, it relies on mathematical notions of a vector space of functions and a basis thereof.

The following subsection is based on chapter 4 of [4] and two courses at TU Wien, [9] and [8].

Variational Formulation The *variational formulation* of problems has seen successful application in many areas of physics, especially mechanics. As basic principle, we have the minimizing of some functional $\Pi[\boldsymbol{x}]$[8]. This functional depends on a set of functions, which are here indicated by the vector $\boldsymbol{x}$, and $\boldsymbol{x}$ belongs to some function vector space X. This functional Π often represents the total energy of the system, while $\boldsymbol{x}$ stands for the trajectories of particles or fields varying in space and time, and the vector space X contains all sufficiently smooth functions that fulfil certain boundary conditions. The physical solution is now the set of functions $\boldsymbol{x}^*$ which minimizes the functional $\Pi[\boldsymbol{x}]$,

[8] The square brackets indicate that it is not a function evaluated at specific values, but a functional evaluated at a whole function.

that is $\Pi[\boldsymbol{x}^*] \leq \Pi[\boldsymbol{x}]$ for all eligible sets of functions $\boldsymbol{x} \in X$. In the classical variational formulation, this is described by the vanishing of the variational derivative at $\boldsymbol{x}^*$:

$$\delta\Pi|_{\boldsymbol{x}^*} = 0 \tag{2.46}$$

Generally, X has infinite dimension, as it encompasses all sufficiently smooth functions that fulfil the respective boundary conditions. However, we can take a finite dimensional subset $Y \subset X$ and reformulate the problem within this subset. Once again we search for the minimal value of $\Pi[\boldsymbol{x}]$, but now we can write the set of functions $\boldsymbol{x}$ with respect to the base functions $\boldsymbol{b}_k$ of Y:

$$\boldsymbol{x} = \sum_k \lambda^k \boldsymbol{b}_k \tag{2.47}$$

This means we can rewrite the functional as being purely dependant on the λ^k and thus as a function instead of a functional:

$$\Pi[\boldsymbol{x}] = \Pi\left[\sum_k \lambda^k \boldsymbol{b}_k\right] = \tilde{\Pi}(\lambda^k) \tag{2.48}$$

Now we can substitute the variational derivative in equation 2.46 by the standard derivative to get:

$$\frac{\partial\tilde{\Pi}}{\partial\lambda^k}|_{(\lambda^*)^k} = 0 \tag{2.49}$$

If we choose the subset Y carefully, the subset solution $\tilde{\boldsymbol{x}}^* = \sum_{k=1}(\lambda^*)^k \boldsymbol{b}_k$ will approximate the real solution $\boldsymbol{x}^*$ well enough. Thus we have transformed a variational problem into a classical calculus problem.

Approximation Functions, Elements and Mesh We remind ourselves that the aforementioned approach works for any subset, but we will only reach a sensible approximation with well chosen subsets Y. Functions within the subset must be able to "imitate" all other functions within the general solution set X for this to work.
In FEM, these functions are obtained by partitioning the problem area into the titular finite elements. For most physical problems, we consider some sort of one-, two- or three-dimensional domain like a line, surface or a solid object. We can now cut this object into small blocks, which, together with their corner points, form the so-called *mesh*. The blocks may have one of many different geometrical forms, but cuboids and tetraeders are most commonly used in three dimensions, whereas quadrilaterals and triangles are most commonly used in two dimensions. An example of such a mesh is given in figure 2.9. These blocks are the finite elements, and their corner points are called the *nodes*.
With this mesh we can now define an example set of functions for the approximation. The

functions should be localized in some way as this makes it possible to locally approximate very well without interfering elsewhere. For this, we define one continuous function associated with each node, which has value 1 at this node and value 0 at all others. In addition, the function should only be non-zero within the elements that have this node as a corner point. These requirements can be achieved by defining a polynomial within each element that interpolates between the nodal points and has value 0 at those element boundaries that do not touch the node. An example for this with triangles in a two-dimensional mesh is given in figure 2.9.

These functions are not analytical, but they are continuous and almost everywhere one-time differentiable.[9] They form a basis of a vector space of functions, and we can approximate any boundary conditions by simply pre-defining the values of the boundary nodes. As with the elements themselves, these functions are not the only ones that can be used, but they are a very common example.

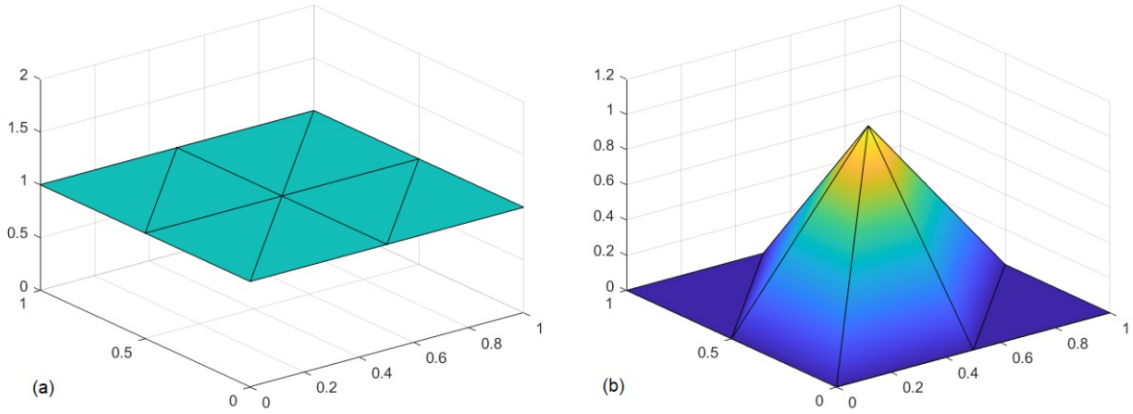

Fig. 2.9: Finite element method for a 1x1 square with triangular elements; (a) the mesh; (b) node function for the central node
(*The scaling of the vertical axis on subfigure (a) is irrelevant. For visualization purposes, subfigure (a) has been tilted to match subfigure (b).*)

Stiffness Matrices Very often, the variational formulation in physics has quadratic terms for the inherent energy and linear terms for the coupling of external forces. The mechanical deformation equation and the thermal diffusion equation given in subsection 2.3.2 are an example for this. We can then split the functional Π into the inherent-energy part Π_E

[9] By careful choice of higher dimensional polynomials, we can actually achieve higher differentiability.

and the force part Π_F, where the first is quadratic and the second linear in the arguments. This gives the following equalities:

$$\Pi\left[\sum_k \lambda^k \boldsymbol{b}_k\right] = \Pi_E\left[\sum_k \lambda^k \boldsymbol{b}_k, \sum_k \lambda^k \boldsymbol{b}_k\right] + \Pi_F\left[\sum_k \lambda^k \boldsymbol{b}_k\right]$$

$$= \sum_{k,l} \lambda^k \lambda^l \Pi_E[\boldsymbol{b}_k, \boldsymbol{b}_l] + \sum_k \lambda^k \Pi_F[\boldsymbol{b}_k] \tag{2.50}$$

The expressions $\Pi_E[\boldsymbol{b}_k, \boldsymbol{b}_l]$ and $\Pi_F[\boldsymbol{b}_k]$ only depend on the base functions $\boldsymbol{b}_k$ and thus can be computed before solving the minimizing equation 2.49. We can define the *stiffness matrix* $M_{k,l}$ and the *load vector* l_k the following way:

$$M_{kl} := \frac{1}{2}\Pi_E[\boldsymbol{b}_k, \boldsymbol{b}_l] \qquad l_k := -\Pi_F[\boldsymbol{b}_k] \tag{2.51}$$

Inserting equation 2.50 into equation 2.49, we see that the result is the following linear equation:

$$\sum_l M_{kl}(\lambda^*)^l = l_k \tag{2.52}$$

This simplification is one of the reasons why the FEM is so successful in numerical simulations. When choosing well known base functions such as the nodal functions discussed above, the stiffness matrix $M_{k,l}$ can often be computed with relative ease. Since the forces vary from problem to problem, the load vector l_k needs to be calculated individually. However as $\Pi_F[\boldsymbol{b}_k]$ often is an integral of $\boldsymbol{b}_k$ multiplied by the forces, this is also done relatively easy. Afterwards, the computer just needs to solve equation 2.52 to get the solution.

The localization of the base functions $\boldsymbol{b}_k$ as described above is an immense benefit for this approach. Since only very few $\boldsymbol{b}_k$ overlap, the matrix $M_{k,l}$ becomes a sparse matrix with zeroes almost everywhere. This way, when choosing a very high number of elements which results in a very high-dimensional matrix $M_{k,l}$, the computer can still handle the matrix calculations.

2.3.2 Application

Overview The theory detailed in subsection 2.3.1 can be applied to many different problems. We are interested in two of them, namely the solid body deformations and the heat conduction equation.

The paragraph about solid body deformations is based on chapter 4 of [4], and the paragraph on plastic deformations is based on chapter 6 of [21]. The last paragraph about thermal conduction is based on chapter 2 of [8], but it can also be found in chapter 7 of [4].

Solid Body Deformations A class of problems which are most commonly solved by the FEM are the solid body deformations. They describe a solid body subjected to different stresses and the deformation thereof. The total energy E_{tot} of the problem is given by the following expression (see chapter 4 of [4]):

$$E_{tot} = \frac{1}{2}\int_V \boldsymbol{\varepsilon} : \boldsymbol{\sigma} dV - \int_V \boldsymbol{f}^B \cdot \boldsymbol{u} dV - \int_S \boldsymbol{f}^S \cdot \boldsymbol{u} dS - \sum_{m=1}^{M} \boldsymbol{F}^m \cdot \boldsymbol{u}^m \tag{2.53}$$

At a certain point x within the deformed body, $\boldsymbol{\varepsilon}(x)$ is the strain tensor, $\boldsymbol{\sigma}(x)$ the stress tensor, $\boldsymbol{f}^B(x)$ the body force density, $\boldsymbol{f}^S(x)$ the surface force density and $\boldsymbol{u}(x)$ the displacement caused by the forces. The integration domain V stands for the full volume of the body, while S stands for its surface. The last term is composed of the sum over all M singular point forces $\boldsymbol{F}^m$ multiplied by the displacement $\boldsymbol{u}^m$ at that point. We can see that the first term represents the inherent work, while the remaining terms represent the influence of exterior forces.
Before we can use equation 2.53 as a starting point for the FEM, we have to relate the stress tensor $\boldsymbol{\sigma}$ with the strain tensor $\boldsymbol{\varepsilon}$ and the strain tensor $\boldsymbol{\varepsilon}$ with the displacements $\boldsymbol{u}$, so that we can reduce the energy to a functional for the displacement functions $\boldsymbol{u}$. The first relationship is given by Hooke's Law:

$$\boldsymbol{\sigma} = \boldsymbol{C}\boldsymbol{\varepsilon} \quad \Leftrightarrow \quad \sigma_{ij} = \sum_{kl} C_{ijkl}\varepsilon_{kl} \tag{2.54}$$

The four-tensor $\boldsymbol{C} \sim C_{ijkl}$ is symmetric and a material property, and it can in the more simple cases be expressed by the elasticity modulus E and the Poisson number ν. The second relationship is given by the definition of the strain:

$$\boldsymbol{\varepsilon} = \nabla \otimes \boldsymbol{u} \quad \Leftrightarrow \quad \varepsilon_{ij} = \frac{\partial u_i}{\partial x_j} \tag{2.55}$$

Together, equations 2.53, 2.54 and 2.55 let us define the energy functional $\Pi[\boldsymbol{u}]$:

$$\Pi[\boldsymbol{u}] = \frac{1}{2}\int_V (\nabla \otimes \boldsymbol{u}) : \boldsymbol{C}(\nabla \otimes \boldsymbol{u}) dV - \int_V \boldsymbol{f}^B \cdot \boldsymbol{u} dV - \int_S \boldsymbol{f}^S \cdot \boldsymbol{u} dS - \sum_{m=1}^{M} \boldsymbol{F}^m \cdot \boldsymbol{u}^m \tag{2.56}$$

From there on, we can use the FEM approach discussed in subsection 2.3.1 to approximate the solution.

Plastic Deformation The calculations above assume a perfectly elastic material, where the deformations are linearly dependent on the applied forces. However, many relevant processes involve plastic straining of the involved materials. The general property of plastic straining is that after reaching some flow stress σ_F, the relationship between stress and strain will not follow the linear Hooke's Law in equation 2.54 anymore. A one-dimensional example for this is given in figure 2.10.
In general, we can describe plastic deformation in the following way (see chapter 6 of [21]). First, we compute the elastic deformation by using the functional given in equation 2.53. From the deformation $\boldsymbol{u}$ we can then compute the strain $\boldsymbol{\varepsilon}$ and the stress $\boldsymbol{\sigma}$, as indicated in equations 2.55 and 2.54. Then we define some scalar *equivalent stress* $\bar{\sigma}$ that depends on the stress tensor $\boldsymbol{\sigma}$ and compare it to the flow stress σ_F of the material:

$$\bar{\sigma}(\boldsymbol{\sigma}) - \sigma_F \geq 0 \tag{2.57}$$

If this condition is met, we assume that plastic deformation takes place, and we reduce the previously purely elastic strain tensor $\boldsymbol{\varepsilon}$ by the plastic strain tensor increment $d\boldsymbol{\varepsilon}_p$:

$$\tilde{\boldsymbol{\varepsilon}} = \boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}_p \tag{2.58}$$

This gives a new stress $\tilde{\boldsymbol{\sigma}}$ according to equation 2.54. We then also consider the plastic hardening by increasing the flow stress proportional to the total amount of plastic strain added:

$$\tilde{\sigma}_F = \sigma_F + H d\varepsilon_p \quad ; \quad d\varepsilon_p = ||d\boldsymbol{\varepsilon}_p|| \tag{2.59}$$

Here, H is the hardening parameter of the material and $||.||$ is the matrix norm. The values of $d\boldsymbol{\varepsilon}_p$ have to be chosen in a way that afterwards, the equivalent stress is equal to the flow stress, that is:

$$\bar{\sigma}(\tilde{\boldsymbol{\sigma}}) - \tilde{\sigma}_F = 0 \tag{2.60}$$

After this equality is reached, the plastic strain $d\boldsymbol{\varepsilon}_p$ is the final result for the plastic deformation of the system.
There are many different models for the definition of the equivalent stress $\bar{\sigma}(.)$ or the choice of the plastic strain increment $d\boldsymbol{\varepsilon}_p$, and we will not go into detail here. More complex models can incorporate a variety of different effects, like anisotropic plasticity or different behaviour under elongation and compression. For example the FEM simulation program *LS-DYNA* has over 200 different models, each briefly described in [16].

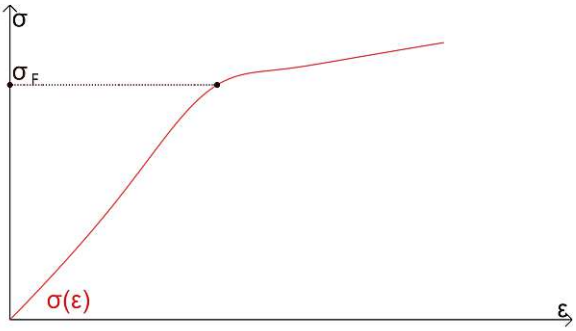

Fig. 2.10: Stress-strain curve of a typical metal. Up until σ_F, the deformation is elastic and approximately linear. Afterwards, there is plastic straining.

Thermal Conduction Another class of common problems for FEM application are the thermal conduction problems. The governing functional $\Pi[T]$ of the temperature T is given by the following equation (see chapter 7 of [4]):

$$\Pi[T] = \int_V \frac{1}{2} \sum_i K_i \left(\frac{\partial T}{\partial x_i} \right)^2 dV - \int_V cT \frac{dT}{dt} dV - \int_V T q^B dV - \int_S T q^S dS \tag{2.61}$$

At the point x we have $T(x)$ as the temperature, $K_i(x)$ as the heat conductivity in i-direction, $c(x)$ as the heat capacity, $^{dT}/_{dt}(x)$ as the time derivation of the temperature, $q^B(x)$ as the heat sources within the body and $q^S(x)$ as the heat sources on the surface. The integration domain V stands for the full volume of the body, while S stands for its surface. Both the heat conductivity and the heat capacity are material parameters.
The main difference between the thermal conduction equation 2.61 and the deformation equation 2.53 is the second term with the time derivative. We can solve this problem by using the FEM approach spatially and an ordinary differential equation in time (see chapter 2 of [8]). We assume that we already have a mesh with base functions b_k, as described in subsection 2.3.1. Now we make the Ansatz that our final solution $T(t)$ will be given by the following:

$$T(t) = \sum_k \lambda^k(t) b_k \tag{2.62}$$

This means that we assume only the coefficients λ^k depend on the time. For each time t, we can now use the FEM approach. We define the following matrices and vectors:

$$M^1_{kl} := \int_V \sum_i K_i \frac{\partial b_k}{\partial x_i} \frac{\partial b_l}{\partial x_i} dV$$

$$M^2_{kl} := \int_V c b_k b_l dV$$

$$l_k^B(t) := \int_V b_k q^B(t) dV \qquad l_k^S(t) := \int_S b_k q^S(t) dS \tag{2.63}$$

The load vectors $l_k^B(t)$ and $l_k^S(t)$ depend on the time t, since the heat sources $q^B(t)$ and $q^S(t)$ are boundary conditions that may depend on the time. Altogether we can plug the Ansatz from equation 2.62 into equation 2.61 to get the following equation:

$$\tilde{\Pi}(\lambda^j) = \frac{1}{2} \sum_{kl} M_{kl}^1 \lambda^k \lambda^l - \sum_{kl} M_{kl}^2 \lambda^k \frac{d\lambda^l}{dt} - \sum_k (l_k^B + l_k^S) \lambda^k \tag{2.64}$$

In the last step, we maximize $\tilde{\Pi}$ for the λ^k at a given time t. The time derivatives $\frac{d\lambda^l}{dt}$ are independent of the λ^k and thus we get:

$$\sum_k M_{ik}^2 \frac{d\lambda^k}{dt} = \sum_k M_{ki}^1 \lambda^k - (l^B + l^S)_i \tag{2.65}$$

This is an ordinary differential equation in time which we can solve either explicitly or through a standard numerical difference scheme.
There are other ways as well to model thermal conduction equations with FEM, but the approach detailed above should give a good insight on how this problem can be tackled, and is fitting for the scope of this work.

2.3.3 List of Symbols

$\boldsymbol{b}_k$, b_k	(vector/scalar) base functions of subset S
c	heat capacity
$\boldsymbol{C}, C_{ijkl}$	Hookes law tensor
$\boldsymbol{f}^B$, $\boldsymbol{f}^S$	body force density, surface force density
$\boldsymbol{F}^m$	point force
H	hardening parameter
K_i	heat conductivity in i direction
M_{kl}	stiffness matrix
M^1_{kl}	heat flow part of thermal stiffness matrix
M^2_{kl}	heat dissipation part of thermal stiffness matrix
l_k	load vector
l^B_k, l^S_k	body load vector, surface load vector
q^B, q^S	body heat source, surface heat source
$\boldsymbol{x}$	potential solution functions of variational problem
$\boldsymbol{x}^*$	solution of variational problem
X	space of potential solution functions
Y	finite dimensional subspace of X
t	time
T	temperature
$\boldsymbol{u}, u_i$	displacement vector
$\boldsymbol{\epsilon}, \epsilon_{ij}$	strain tensor
$d\epsilon_p$	plastic strain increment
$d\boldsymbol{\epsilon}_p$	plastic strain tensor increment
λ^k	coordinates with respect to base functions $\boldsymbol{b}_k$
$(\lambda^*)^k$	coordinates of the solution
$\Pi[\boldsymbol{x}]$	(energy) functional for variational formulation
$\Pi_E[\boldsymbol{x}_1, \boldsymbol{x}_2]$	bilinear inherent energy functional
$\Pi_F[\boldsymbol{x}]$	linear force functional
$\tilde{\Pi}(\lambda^k)$	(energy) functional $\Pi[\boldsymbol{x}]$ evaluated at $\boldsymbol{x} = \sum_k \lambda^k \boldsymbol{b}_k$
σ_F	flow stress
$\bar{\sigma}(\boldsymbol{\sigma})$	equivalent stress

Chapter 3

Experiment

The theories and algorithms discussed within this thesis need validation. To this end, a set of relatively simple compression tests that had already been performed at LKR Leichtmetallkompetenzzentrum Ranshofen GmbH (LKR) was taken as a reference for the numerical simulation. The simulation incorporated the different theories discussed in chapter 2 by using the FEM solver *LS-DYNA*, the precipitation kinetics program *MatCalc* and some user written FORTRAN code.
The overall setup of programs had already been used at LKR to simulate similar experiments, but this time a coupling between *LS-DYNA* and *MatCalc* was implemented.

3.1 Compression Test

In 7 experiments, small aluminium cylinders were compressed and heat-treated at LKR with the DIL 805A/D/T, a dilatometer with integrated compression and heating capabilities (see [1] for more information). The process consisted of three stages: the pre heat treatment, the deformation and the post heat treatment, which are schematically shown in figure 3.1. Originally, the goal of this experiment was to measure the influence of heat treatments on recrystallization after deformation, but within the context of this work it is used as reference for the numerical simulation.
The compressed cylinders had a diameter of 5 mm and a height of 10 mm and consisted of the alloy AA2024 detailed in table 3.1. All process parameters are given in table 3.2. They were first subjected to a pre heat treatment, where they were heated with a rate $(dT/dt)_a$ and then remained at a temperature of T_a for a specific time t_a. Afterwards, the cylinders were compressed at this temperature with a strain rate of $\dot{\varepsilon}_b$ up to a total strain of ε_b. Following the compression, the specimens were subject to two different post heat treatments. In experiment No. 1, the cylinder was kept at the deformation temperature for a time t_{c_1} and rapidly cooled down afterwards at a rate $(dT/dt)_{c_1}$. In the experiments No. 2-7, the cylinders were rapidly cooled down at a rate $(dT/dt)_{c_1}$ immediately after

the deformation and then subsequently heated again up to a temperature of T_{c_2} with a heating rate of $(dT/dt)_{c_2}$. In experiment No. 7, the specimen was left at constant high temperature after it reached T_{c_2} for a time t_{c_2}. After their respective treatment ended, all cylinders were cut in half, anodized with Barker's reagent to make the microstructure visible and then analyzed under the microscope (see figures 5.4 and 5.8). The whole process is schematically depicted in figure 3.1.

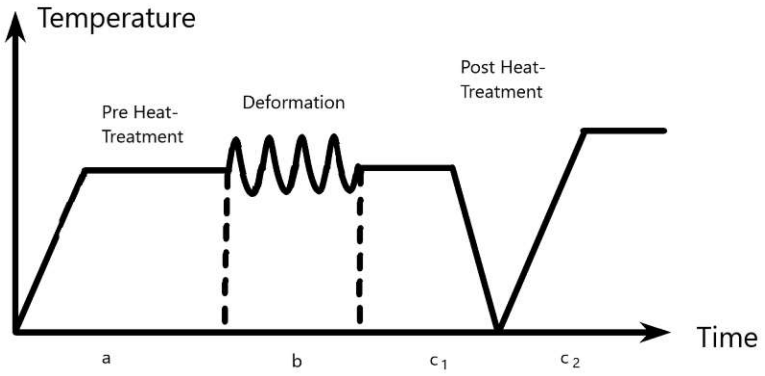

Fig. 3.1: Schematic overview of the compression test consisting of (a) the pre heat treatment, (b) the deformation, (c1) the first step of the post heat treatment (for all cylinders), (c2) the second step of the post heat treatment (the corresponding samples were quenched directly after deformation)

Tab. 3.1: Chemical composition of the aluminium alloy AA2024 in %.

	Al	Cu	Mg	Mn	Fe	Si
AA2024	93,69	4,27	1,42	0,4	0,12	0,1

Tab. 3.2: Parameters of the different compression tests

N	$(dT/dt)_a$ [°C/s]	T_a [°C]	t_a [s]	ε_b [-]	$\dot{\varepsilon}_b$ [s^{-1}]	t_{c_1} [s]	$(dT/dt)_{c_1}$ [°C/s]	$(dT/dt)_{c_2}$ [°C/min]	T_{c_2} [°C]	t_{c_2} [min]
1	15	470	10	1	1	600	35	-	-	-
2	15	470	900	1	1	0	35	7,1	450	-
3	15	470	900	1	1	0	35	7,1	460	-
4	15	470	900	1	1	0	35	7,1	470	-
5	15	470	900	1	1	0	35	7,1	480	-
6	15	470	900	1	1	0	35	7,1	490	-
7	15	470	900	1	1	0	35	7,1	490	33

3.2 Simulation Programs

The numerical implementation of the experiments described in section 3.1 uses two different programs, namely *MatCalc* and *LS-DYNA*. They handle the precipitation (*MatCalc*) and the FEM-computation (*LS-DYNA*). The microstructure model MD^{2}M described in the section 2.1 and the appendix is implemented as user code in *LS-DYNA*. A Flowchart of their inputs and outputs is given in figure 3.2.

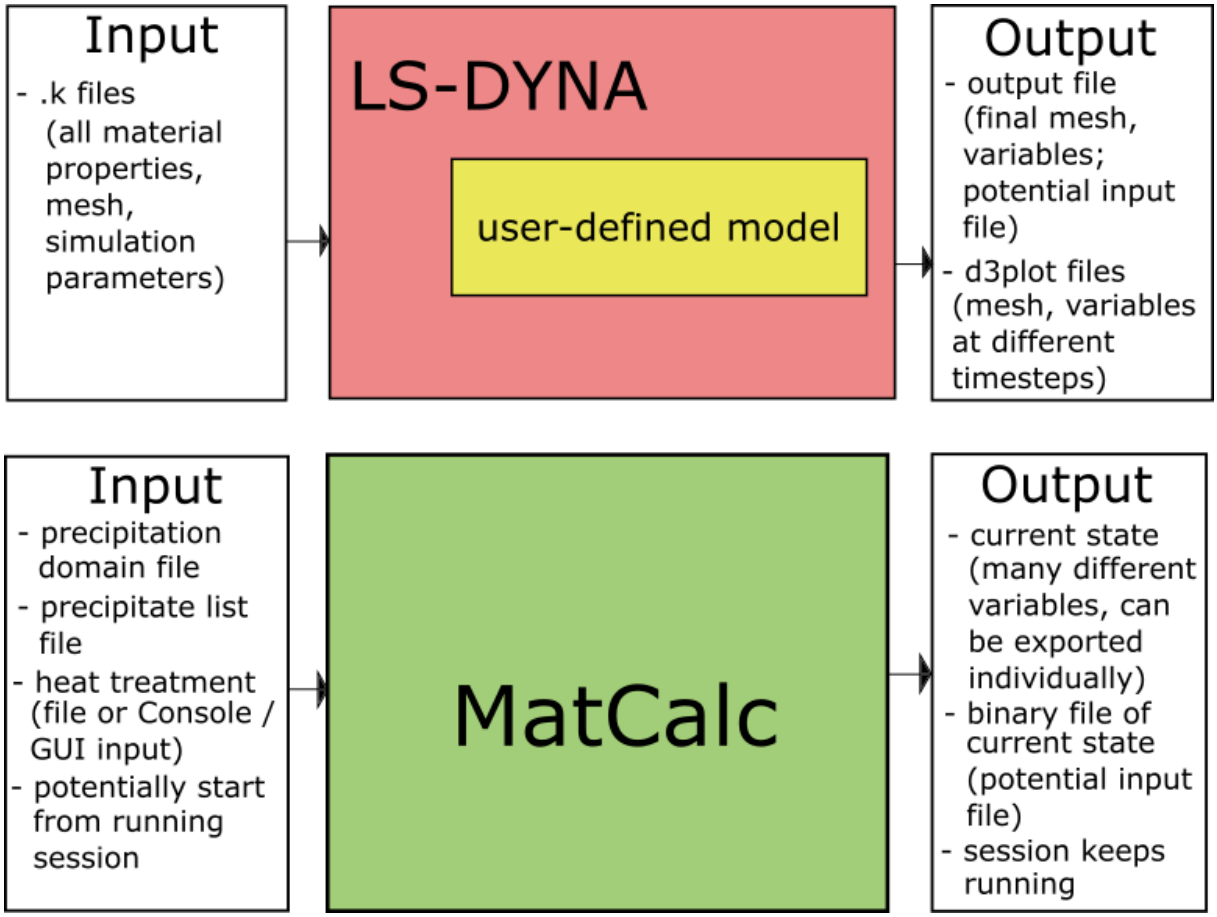

Fig. 3.2: Overview of the inputs and outputs of the programs discussed in this section. The user-defined models described in subsection 3.2.2 are part of the *LS-DYNA* program and thus have no explicit input / output themselves.

3.2.1 FEM Solver

LS-DYNA For the overall simulation of compression and heat treatment, the program *LS-DYNA* version R11 was used. *LS-DYNA* is a general purpose FEM program based on the approach detailed in section 2.3 and it can run thermal, mechanical and thermo-mechanical simulations. It began its development in the 70's as a way to simulate in

3D the impact of low-altitude release nuclear warheads at Lawrence Livermore National Laboratory (see [5]), but it has evolved into an all-purpose FEM commercial software used around the globe.
A major asset of *LS-DYNA* is the large library of material models (see [16]), which can be defined in the input. There are both thermal and mechanical models, which describe the nonlinear element behaviour of the FEM equations. The input / output handling of *LS-DYNA* can either be done by manually writing into the text-files, or by using *LS-PrePost* or similar pre-processing programs for *LS-DYNA*.

Input *LS-DYNA* takes the material properties (e.g. density, elastic modulus, ...), the finite element mesh and the simulation procedure as input. They are written in specific text files with the *.k* ending.

Output The simulation saves the current mesh at predefined intervals (which may be larger than the actual timesteps) to *d3plot* files containing the values of most variables. These files can be visualized with *LS-PrePost* (see the figures in chapter 5). In addition, the last state can be written to an ASCII file, which can be used as input file with a *.k* ending for a subsequent simulation.

3.2.2 Microstructure Models

User-Defined Material Models The program *LS-DYNA* allows a certain amount of user code to be implemented. In place of the material models mentioned in subsection 3.2.1, the user can choose to write their own code in Fortran. The code will then be called by *LS-DYNA* in each timestep once for each FEM element. Originally, this was intended for the user to write more exotic nonlinear models or get a better control over individual calculations, but the open-ended nature of this function allows to implement additional models, such as the microstructure dynamics detailed in section 2.1, on top of the FEM. Such a model comes with a set of free variables that can be used to store and calculate values for each element. The program *LS-DYNA* then saves these variables for each timestep, which can be used for further calculations, but also for visualization in post-processing. The plots in figures 5.13 and 5.7 show the distribution of variables that were stored in this way.

Mean Dislocation Density Model The MD^2M detailed in subsection 6.2.3 was implemented within this user code. For the work hardening part, a mechanical model was used, while the recovery and recrystallization part was written in a thermal model. This means

for each timestep and for each FEM element, the microstructure is calculated and the variables are stored using the aforementioned free variables.
For the mechanical model, a simple calculation for plastic straining was added in the beginning to calculate the deformation and obtain the plastic strain increment $d\epsilon_p$.
For the thermal model, an option for clustering and interfacing with *MatCalc* was also implemented, which is described in more detail in chapter 4.

Input / Output The inputs and outputs of the MD^{2}M are described in the appendix. Additionally, the work hardening part takes mechanical input provided by *LS-DYNA*. A schematical representation is given in figure 3.3.

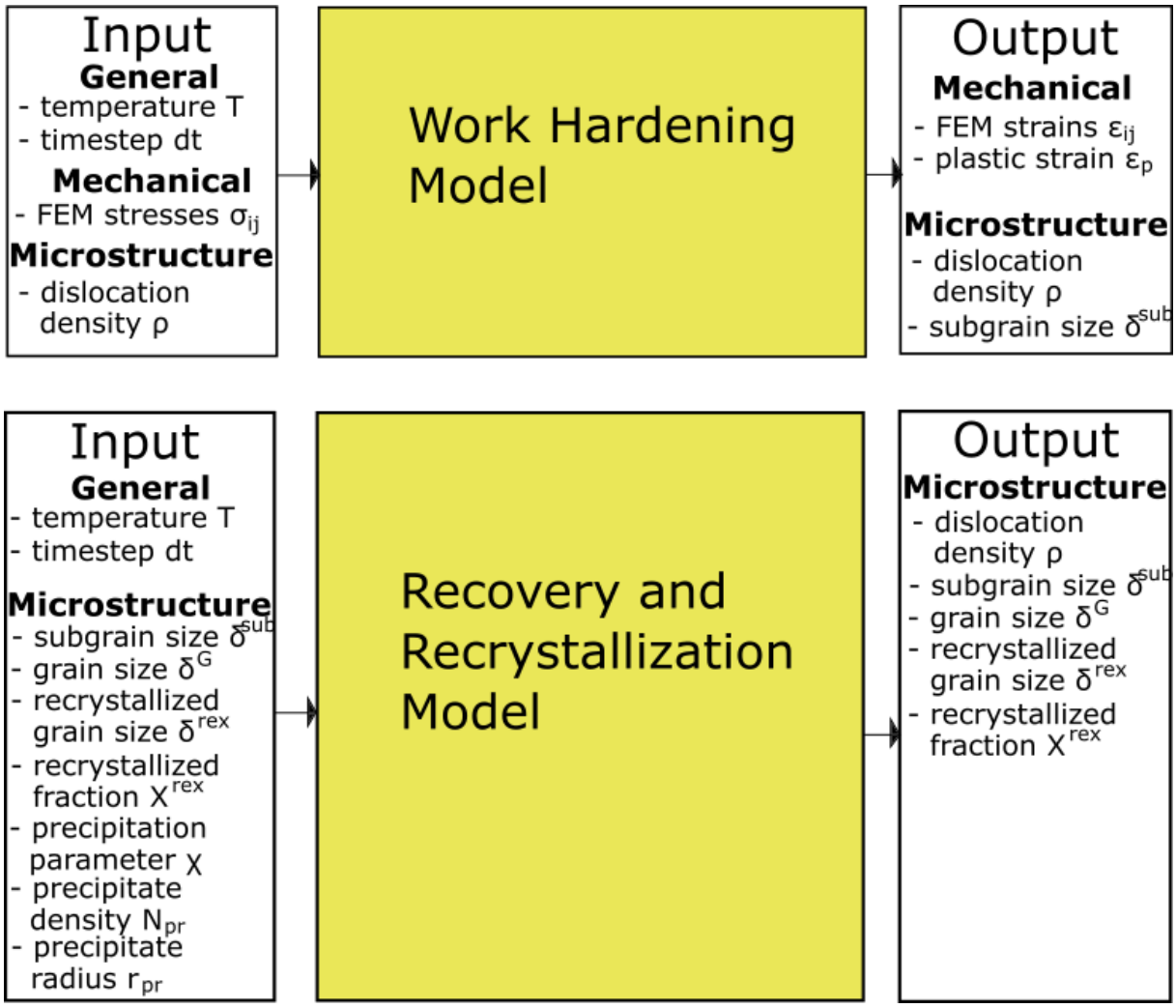

Fig. 3.3: Overview of the inputs and outputs of the user defined models discussed in this section.

3.2.3 Precipiation Solver

For the implementation of the precipitation kinetics discussed in section 2.2, the program *MatCalc* version 6.04 was used. *MatCalc* started as a PhD thesis project at the Graz University of Technology, Austria, in 1993. Initially, it modelled precipitation in interstitial-free and bake-hardening steel, but over the years, it has been greatly refined.
In its current state, *MatCalc* uses the nucleation theory described in subsection 2.2.1 and the evolution equations derived from the thermodynamic extremal principle described in subsection 2.2.2 to simulate multicomponent precipitation in a given metal. A major advantage of *MatCalc* are the vast databases which it accesses for these calculations. It can either be operated through the console or via GUI (graphical user interface). More information about the version and program can be found in [2].

Input The program is given three types of input. First, it needs the properties of the overall domain where the precipitation happens, such as the dislocation density, the grain size, the chemical composition and the phase of the surrounding matrix and others. Second, the program gets a list of all different possible precipitates within the precipitation domain. Each entry in this list entails the chemical composition of the precipitate, the lattice misfit, the interface mobility and more. The last input is the heat treatment, meaning the simulation time and the corresponding temperature progression. The inputs can either be read from a file or be given directly to *MatCalc* through the console or GUI.

Output *MatCalc* calculates a plethora of variables during the precipitation simulation, such as individual precipitates and their sizes. Any of those variables can be requested directly through the console or GUI. *MatCalc* can also save the current state into a binary file, which may be used as input at a later time. Finally, a *MatCalc* session keeps running until explicitly closed and it always stores the last computed state. This means a session can be called again to continue the computation where it last ended.

3.3 Numerical Implementation

The simulation of the different experiments described in section 3.1 was done in three steps, as shown in figure 3.4. Each step is a distinct simulation in itself, but these different simulations pass their results over to the next step, so that a coherent simulation of the whole process emerges.

To examine the influence of the clustering algorithm described in 4, all 7 experiments were simulated in 3 different variants (see table 3.3). The first two stages of the experiment (see figure 3.1), namely the pre heat treatment and the deformation were simulated the same way for all 3 variants, since they either use *LS-DYNA* or *MatCalc* and don't require any coupling between the two programs. The last stage of the simulation, namely the post heat treatment, is where the 3 variants differ in whether they coupled to *MatCalc* and how this coupling was implemented.

Tab. 3.3: Simulation Variants

	MatCalc	Clustering
1	no	no
2	yes	no
3	yes	yes

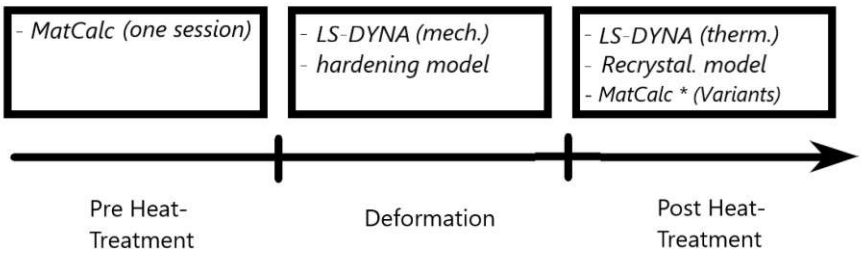

Fig. 3.4: Schematic overview of the numerical implementation and the programs used in each simulation step. The * in the last step indicates the different variants, as seen in table 3.3.

3.3.1 Pre Heat Treatment

For the pre heat treatment simulation, *MatCalc* was used to simulate the precipitation kinetics (described in section 2.2) during the pre heat treatment. The program was started once for the whole specimen, since the initial microstructure of the material was assumed to be uniform.

Input The starting value for the grain size δ^G was measured in advance, the starting value for the dislocation density was taken as $10^{11}\ m^{-2}$ (an estimated value in line with typical values of $10^{10}\ m^{-2} - 10^{11}\ m^{-2}$, see [11]), and the material parameters given in table 3.1 were used for the precipitation domain, as well as for the list of possible precipitates.

Output As output, the mean radius of the precipitates r_{pr}, the precipitate density N_{pr} and the precipitation parameter χ (see equation 2.9) at the end of the heat treatment were obtained, as well as a binary file that saved the precipitation state for use in the post heat treatment simulation.

3.3.2 Deformation

The mechanical FEM solver of *LS-DYNA* (see subsection 3.2.1) was used to simulate the compression of the specimen. For the implementation of the work hardening dynamics of the microstructure, the MD^{2}M was implemented in the previously described mechanical user-defined material in *LS-DYNA*.

Input In previous simulations at LKR, fitting parameters such as A, B, C from equation 2.10 had already been fitted through extensive trial simulations. Material parameters such as the elastic modulus E had been measured through experiments or looked up in literature (see [13] for a detailed list of values and sources). The initial grain size was taken as 65 μm (also determined through experiments). All precipitation results were taken from the output of the pre heat treatment simulation in *MatCalc*. The modelling and meshing of the cylinder was done in *LS-PrePost*.

Output As output, the deformed specimen was obtained, as well as for each element the values of the dislocation density ρ, the grain size δ^G and the plastic strain ε_p.[1]

3.3.3 Post Heat Treatment

The thermal FEM solver of *LS-DYNA* was used to simulate the post heat treatment. The recovery and recrystallization part of MD^{2}M was implemented in the previously described thermal user-defined material in *LS-DYNA*.
For experiments No. 2-7 in table 3.2, only experiment 7 was simulated. This is because the simulation of experiment No. 7 incorporates all others, as they were conducted in the same way but stopped at lower temperatures.

[1] Passing the subgrain size δ^{sub} was not necessary, since it is related to the dislocation density ρ via equation 2.11.

There was also a clustering option implemented into the user code within *LS-DYNA* to couple it to the simulation performed by *MatCalc*. It is described in more detail in chapter 4. To evaluate the impact of this clustering, three variants of the post heat treatment simulation were conducted for each experiment. They differed in whether they used the clustering algorithm and *MatCalc* for the precipitation kinetics. Table 3.3 gives an overview of the different variants.

Input As with the deformation simulation, all material parameters and fitting parameters had already been determined for previous simulations at LKR. The output of the mechanical *LS-DYNA* simulation provided the input for the thermal *LS-DYNA* simulation, with the mesh, the dislocation density ρ, the grain size δ^G and the plastic strain ε_p being preserved. For those variants with coupling to *MatCalc*, the binary output file from the end of the pre heat treatment simulation in *MatCalc* was used as a starting point. In the other variant, the precipitation parameter χ, the precipitate density N_{pr} and the precipitate radius r_{pr} were taken from the pre heat treatment as constant values.

Output The output of this simulation was the time dependent distribution of the microstructure and precipitation values. For variant 3, the time dependent cluster distribution was also part of the output.

Variant 1 The first variant (see table 3.3) was the most simple simulation, where *MatCalc* was not used in the post heat treatment stage of the simulation. Instead, the final precipitation values obtained during the pre heat treatment were used as constant values during the post heat treatment simulation.

Variant 2 The second variant (see table 3.3) coupled one session of *MatCalc* to the FEM simulation in *LS-DYNA*. In each timestep within the simulation, the values of the grain size δ^G and the dislocation density ρ were averaged over all FEM elements, and the results were passed to the *MatCalc* session. This method allowed the precipitation values calculated by *MatCalc* to play a more dynamic role, but it didn't take into account the various regions with different microstructure within the specimen.

Variant 3 The third variant (see table 3.3) used the clustering algorithm described in chapter 4 to couple *LS-DYNA* to *MatCalc*. As many *MatCalc* sessions as there were clusters ran concurrently to the simulation, which lead to different precipitation values in different regions of the simulation.

Chapter 4

Coupling Grain Structure and Precipitation

One goal of this master thesis was the coupling of the grain size evolution calculated by the MD^{2}M and implemented within *LS-DYNA* to the precipitation kinetics simulated by *MatCalc* during the post heat treatment. In previous simulations at LKR, it was assumed that the number and size of precipitations doesn't change after the pre heat treatment. However, depending on the time and duration of the post heat treatment, this assumption can be wrong. Thus, the aforementioned coupling becomes necessary.
This proved to be a difficult task, as it is not feasible to simply call a *MatCalc* session for each FEM element, since the interfacing with *MatCalc*, as well as its computations, are very time consuming. Therefore, an alternative method needed to be employed, namely the clustering algorithm described within this chapter.

The suggested algorithm is of general purpose and not confined to solving the problem discussed above. It might be applied to similar problems in multi-scale computations when simulations at different length scales are performed.

4.1 Clustering of Large Datasets

We look at large datasets (e.g. all elements within a FEM simulation) which require a complex evaluation function (e.g. the calculation of precipitation kinetics with *MatCalc*). The idea, then, is to split the data points into clusters with similar values and then compute the evaluation function once for each cluster instead of once for each data point. If the data points are similar enough and if the evaluation function is smooth enough, the error between the calculation for the cluster mean value and each data point within that cluster will be negligible. Thus, instead of calling the evaluation function once for each

data point, it will only be called once for each cluster. We can estimate the time reduction for this computation in the following way:

$$t_{cluster} \approx t_0 \frac{Z}{N} \tag{4.1}$$

Here, $t_{cluster}$ is the computation time with the clustering algorithm implemented, t_0 is the computation time without the algorithm, Z is the estimated number of clusters and N is the number of data points. A schematic representation of this idea is given in figure 4.1.

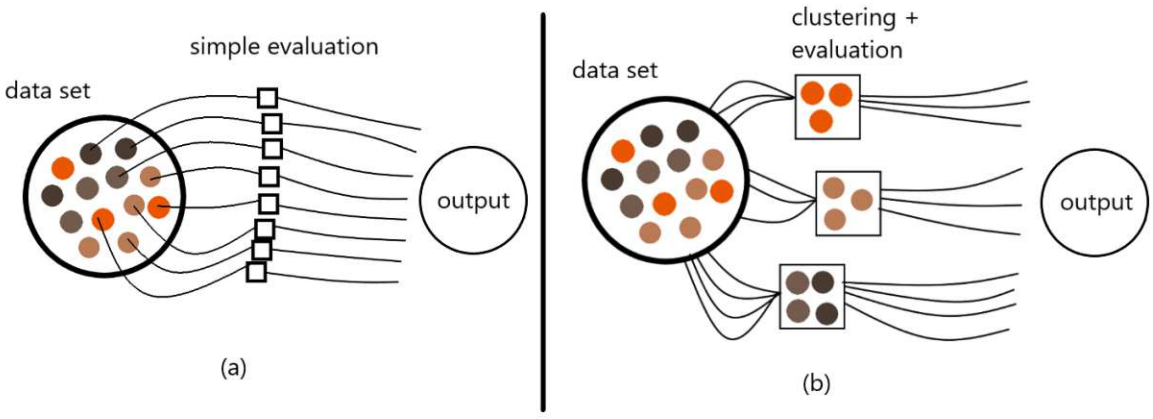

Fig. 4.1: Overview of the clustering algorithm. Similar data points are represented in the same colour and the square boxes represent the computation of an evaluation function; (a) without clustering; (b) with clustering

4.1.1 Clustering

Overview The clustering algorithm should take a set of N data points as input and give a set of Z clusters as output, where Z will be determined during the clustering. To achieve this, we will have to face two challenges. First, we need to define a measure of "closeness", so that we can determine when two data points possibly belong to the same cluster. Second, we need to think about the algorithm with which the clusters are created.[1] We will tackle both problems in the following subsection. The algorithm is shown for an example of 2 dimensional data points in figure 4.2.

[1]This problem of clustering has been discussed extensively in recent years (see [22]), but the main focus there has been data evaluation. This means, given a certain dataset, the goal is to cluster it into sensible clusters of data for better human evaluation (an example would be the mobile phone user data within a country, where the intent is to detect user patterns). However, our goal is not pattern recognition but efficient computation, so we will develop our own approach.

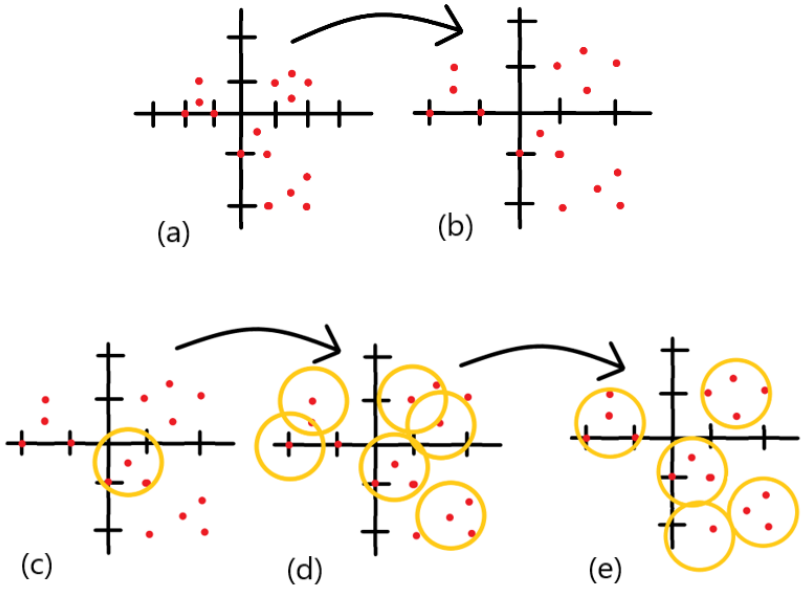

Fig. 4.2: Overview of the clustering algorithm. The two dimensional data points are represented as red dots; (a) unclustered dataset; (b) rescaled unclustered dataset; (c) start of clustering; (d) during clustering; (e) reclustering finished

Weighted Norm We start by denoting the data points by $\boldsymbol{x}^n$. Each data point is an array of arbitrary dimension K, such as Euclidean coordinates, the pressure and temperature at a certain place or the grain size and dislocation density of an FEM element, and a two dimensional example for a dataset is given in figure 4.2 (a). Each data point $\boldsymbol{x}^n$ has entries x_k^n, and, since we want to compare and cluster them, all data points need to be of the same dimension K. As we want to cluster data points with similar values, we need to define a notion of "closeness" for them. The different data dimensions might vary drastically in scale, for example when looking at grain size and dislocation density, where the values for grain size are of magnitude $\sim 10^{-6}\ m$ but the values for dislocation density are of magnitude $\sim 10^{12}\ m^{-2}$. Therefore we introduce global limits λ_k for each data dimension. The limits denote the maximum distance in each dimension that we want to have in our clusters. To continue with the previous example, the limit for the grain size could be $2 \cdot 10^{-7}\ m$, but the limit for the dislocation density could be $5 \cdot 10^{11}\ m^{-2}$. We can now introduce a distance measure that will help us with the clustering, the *weighted norm* d_w^λ. It is defined for two data points $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$ as follows:

$$d_w^\lambda(\boldsymbol{x}^i, \boldsymbol{x}^j) := \left(\sum_{k=1}^{K} \left(\frac{x_k^i - x_k^j}{\lambda_k} \right)^2 \right)^{\frac{1}{2}} \tag{4.2}$$

As we can see, the weighted norm is just the Euclidean norm, but with the k-th dimension divided by λ_k, and this dependence on λ is denoted by the superscript. It gives a general approach to "closeness", as we now have two data points $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$ eligible for the same cluster if and only if $d_w^\lambda(\boldsymbol{x}^i, \boldsymbol{x}^j) \leq 1$.
From a mathematical point of view, equation 4.2 is equivalent to rescaling the k-th coordinate axes by a factor $1/\lambda_k$, which is shown as step (b) in figure 4.2.

Cluster Creation From now on we will assign each cluster a number z, and Z will be the total number of clusters. Each cluster z has a special point $\boldsymbol{c}^z$ with coordinates c_k^z, which we call the *cluster center*[2]. Whenever we want to measure the "closeness" between a cluster z and a data point $\boldsymbol{x}^n$, we will use this cluster center as a reference and compute the weighted norm $d_w^\lambda(\boldsymbol{c}^z, \boldsymbol{x}^n)$.
We now start the clustering by assigning $\boldsymbol{x}^1$ to the first cluster and designating $\boldsymbol{x}^1 = \boldsymbol{c}^1$ as the cluster center, which gives us a total cluster number of $Z = 1$. From now on, we go through all data points $\boldsymbol{x}^n$ in sequence, and for each do the following:

- Check for the distance $d_w^\lambda(\boldsymbol{x}^n, \boldsymbol{c}^z)$ to all existing clusters and record the cluster z_{min} with the minimal distance $d_w^\lambda(\boldsymbol{x}^n, \boldsymbol{c}^{z_{min}})$.
- If $d_w^\lambda(\boldsymbol{x}^n, \boldsymbol{c}^{z_{min}}) \leq 1$, then assign $\boldsymbol{x}^n$ to the cluster z_{min}.
- If $d_w^\lambda(\boldsymbol{x}^n, \boldsymbol{c}^{z_{min}}) > 1$, then create a new cluster numbered $Z + 1$, assign $\boldsymbol{x}^n$ to this new cluster and designate $\boldsymbol{x}^n = \boldsymbol{c}^{Z+1}$ as the new cluster center.

After running this algorithm for all data points, every data point will belong to a cluster, and within a certain cluster every data point will have a weighted norm difference of at most 1 to the cluster center.[3] The whole process is schematically shown in steps (c) and (d) of figure 4.2, where the clusters are indicated by yellow circles. In step (c), the first cluster is chosen and in step (d), the subsequent clusters are chosen according to the discussed algorithm.

Reclustering After the clusters have been created, we can try to improve them by using an approach very similar to the K-means clustering approach (see [22]).
For each cluster, we calculate the average value of all data points belonging to that cluster and set these averages as the new cluster centers $\boldsymbol{c}^z$. Then we once again go through all data points and assign each data point $\boldsymbol{x}^n$ to the cluster where $d_w^\lambda(\boldsymbol{x}^n, \boldsymbol{c}^z)$ is minimal. The

[2] It is important to note that $\boldsymbol{c}^z$ is not the average value of all data points in the cluster z, since we a priori don't know which data points will belong to it.
[3] When we look at the definition of d_w^λ in equation 4.2, we see that this conforms to our initial wish that elements within one cluster should differ at most by roughly λ_k in the k-th coordinate.

difference to the previous algorithm is that the total number of clusters Z and their centers $\boldsymbol{c}^z$ are already known, and that no new clusters are created in this step. We can repeat this step as often as we like to further refine the clustering. After each reclustering, there is a possibility that a cluster is empty, since all elements actually fit better into different clusters. If this is the case, we simply remove the cluster, relabel the other clusters to $1, ..., Z-1$ and do not consider the removed cluster any further. The effect of reclustering is schematically shown in step (e) of figure 4.2.

4.1.2 Evaluation

Overview After clustering the data points, we can now turn our attention to computing the evaluation function, which we will call $\boldsymbol{f}(\boldsymbol{x})$ from now on. This evaluation function can be any continuous function, such as the precipitation kinetics of a local microstructure. We will quickly discuss the evaluation algorithm and then take a look into the underlying mathematics to justify its potential accuracy. As we will see, this accuracy depends greatly on the continuity of the evaluation function.

Evaluation Algorithm For a cluster z, we calculate the average value of all data points within that cluster, which we denote by $\bar{\boldsymbol{x}}^z$. Afterwards, we evaluate the function at this average value to get $\boldsymbol{f}(\bar{\boldsymbol{x}}^z)$. Then we distribute this result back to all data points within that cluster. If we denote $\boldsymbol{f}_{cluster}(\boldsymbol{x})$ as the evaluation by the clustering algorithm, we get for data points $\boldsymbol{x}^n$ in the cluster z:

$$\boldsymbol{f}(\boldsymbol{x}^n) \sim \boldsymbol{f}_{cluster}(\boldsymbol{x}^n) = \boldsymbol{f}(\bar{\boldsymbol{x}}^z) \tag{4.3}$$

The $\sim$ indicates that the clustering result $\boldsymbol{f}_{cluster}(\boldsymbol{x}^n)$ is (hopefully) similar to the actual result $\boldsymbol{f}(\boldsymbol{x}^n)$ for each data point $\boldsymbol{x}^n$.

Accuracy To justify the approximation in equation 4.3, we will now look at a function $\boldsymbol{f}$ which is uniformly continuous in the relevant domain.[4] Further, instead of the standard Euclidean norm $|.|_2$ we use d_w^λ. Since d_w^λ is just a global rescaling coupled with the standard Euclidean norm, it is equivalent to $|.|_2$, and therefore the uniform continuity holds for d_w^λ. The advantage now is that we can fine-tune the rescaling in different directions by

[4] This assumption is plausible, since most physical processes are continuous, apart from a finite number of discontinuities (such as phase transformations). For domains that contain no discontinuities, as long as the input data points are bounded, we will always have uniform continuity. For domains that contain a discontinuity, the approximation will hold everywhere except for a small area around the discontinuity.

changing the limits λ_k to achieve the optimal δ in the inequality. For a given ϵ this gives the following inequality:

$$d_w^{\lambda}(\boldsymbol{x}, \boldsymbol{y}) \leq \delta \quad \Rightarrow \quad |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})| \leq \epsilon \quad \forall\, x, y \tag{4.4}$$

We now define new limits $\lambda_k^* := \lambda_k \delta$. The new limits produce a new weighted norm $d_w^{\lambda^*}$ for which we can deduce the following relation from equation 4.2:

$$d_w^{\lambda^*}(\boldsymbol{x}, \boldsymbol{y}) = \delta d_w^{\lambda}(\boldsymbol{x}, \boldsymbol{y}) \tag{4.5}$$

Substituting this into equation 4.4, we get:

$$d_w^{\lambda^*}(\boldsymbol{x}, \boldsymbol{y}) \leq 1 \quad \Rightarrow \quad ||\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})|| \leq \epsilon \quad \forall\, \boldsymbol{x}, \boldsymbol{y} \tag{4.6}$$

When we now look at the cluster creation, we see that within a cluster, data points have a weighted norm difference of $\lesssim 1$ to the cluster average. This gives for a data point $\boldsymbol{x}^n$ within a cluster z with the cluster average $\bar{\boldsymbol{x}}^z$ the following inequality:

$$||\boldsymbol{f}(\boldsymbol{x}^n) - \boldsymbol{f}_{cluster}(\boldsymbol{x}^n)|| = ||\boldsymbol{f}(\boldsymbol{x}^n) - \boldsymbol{f}(\bar{\boldsymbol{x}}^z)|| \lesssim \epsilon \tag{4.7}$$

We can achieve a sufficiently small ϵ by choosing the limits λ_k well.
Altogether this shows the importance of the continuity of the evaluation function $\boldsymbol{f}$, which has to be considered before each application, as well as the strong dependence on the chosen limits λ_k.

4.1.3 History-Dependent Clustering

Time Evolution Problems When the output of the evaluation function describes the time evolution of a system (e.g. the precipitation kinetics of a metal), its output is often dependent on external parameters (e.g. temperature, dislocation density, grain size), but also on its internal state and its previous internal variables (e.g. the current sizes and types of all different precipitates). We assume that the evaluation function has the following form:

$$\boldsymbol{f}(\boldsymbol{x}, S) = (\boldsymbol{y}, S^*) \tag{4.8}$$

The inputs are the external variables $\boldsymbol{x}$ and the internal state S. The outputs are some variables $\boldsymbol{y}$ which we will use externally, and the evolved internal state S^*.
We could apply the previously discussed clustering algorithm to this problem. However, we would need to cluster over multiple timesteps for all input variables, that is for $\boldsymbol{x}$ and

for S. As S potentially contains a high number of variables (e.g. the current sizes and types of all different precipitates), this is not always feasible.

History-Dependent Clustering We can solve the problem by preserving the clusters over different timesteps and storing the internal state S once for each cluster, since all elements within one cluster have the same state. For this, we assume that we have the following pairs of input to our problem:

$$\begin{aligned} (z_{old}, S^{z_{old}}) \qquad & 1 \leq z_{old} \leq Z_{old} \\ (\boldsymbol{x}^n, z^n_{old}) \qquad & 1 \leq n \leq N \end{aligned} \tag{4.9}$$

The pairs $(z_{old}, S^{z_{old}})$ denote the old clusters z_{old} and their respective internal state $S^{z_{old}}$, while the pairs (x^n, z^n_{old}) denote the data points and the clusters they currently belong to. The subscript *old* emphasizes the previous clustering.[5]
We then group the data points into sets according to their old clusters:

$$X_{z^*_{old}} := \{\boldsymbol{x}^n : z^n_{old} = z^*_{old}\} \tag{4.10}$$

Afterwards, we cluster the $X_{z^*_{old}}$ individually according to the previously discussed clustering algorithm, but without evaluating them yet.
To evaluate the newly obtained cluster z, we calculate the average of the data points $\bar{\boldsymbol{x}}^z$ as before. For the evaluation function, we also need the internal state of the cluster. Since we clustered the $X_{z^*_{old}}$ individually, all $\boldsymbol{x}^n$ within the cluster come from the same old cluster z^*_{old}. All data points within the cluster z thus have the internal state $S^{z^*_{old}}$. We can now evaluate a data point $\boldsymbol{x}^n$ within the cluster:

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{x}^n, S^{z^n_{old}}) \sim \boldsymbol{f}_{cluster}(\boldsymbol{x}^n, S^{z^n_{old}}) = \\ = \boldsymbol{f}(\bar{\boldsymbol{x}}^z, S^{z^*_{old}}) = (\boldsymbol{y}^z, S^z) \end{aligned} \tag{4.11}$$

here, $\boldsymbol{y}^z$ is the output in which we are interested, and S^z is the new internal state of the cluster z.

Effect of History-Dependend Clustering With this algorithm, the previous clustering is taken into account, and it can only be refined. Since different clusters have different internal states and histories, their elements must be considered separately and they cannot

[5] In the case of the first clustering, the algorithm needs the existing clusters z_{old} and internal states $S^{z_{old}}$ as input. An option is to have just one starting cluster, which means that at the beginning all data points have the same cluster and thus the same internal state S.

be put into the same new cluster. The effect is that clusters can never merge together, even if, at a later time, their data points have similar values again. They can only stay the same or split up into smaller clusters, and figure 4.3 shows such a process.

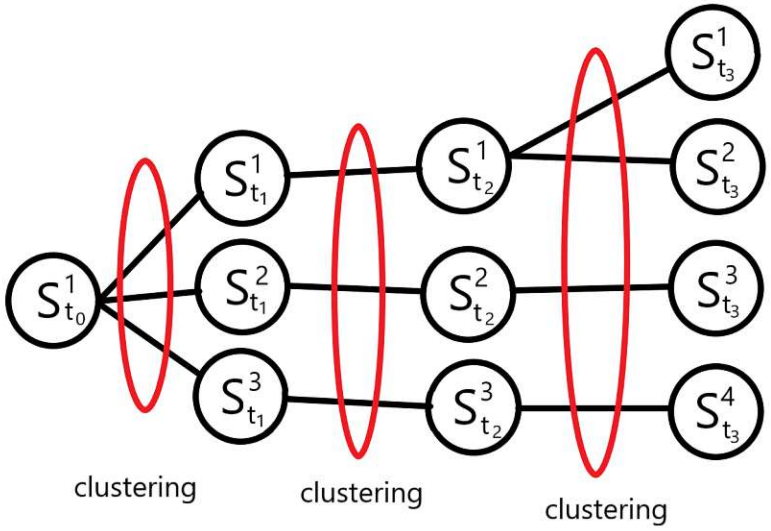

Fig. 4.3: Schematic representation of history-dependent clustering. Each circle represents a cluster with the internal state S^i_t at time t. In the first timestep, the initial cluster splits up into three distinct clusters. In the second timestep, all values are similar enough so that the clusters don't split up further. In the third timestep, the first cluster splits in two, because its data points have diverged too much since the last clustering. This is however not the case for the other two clusters and they stay the same.

4.2 Implementation

As seen by the the equations describing the precipitation kinetics in section 2.2, *MatCalc* is a continuous evaluation function, so we can use the clustering algorithm described in section 4.1 to couple *LS-DYNA* and *MatCalc*. This algorithm first had to be put into code, and this code then needed to be implemented within *LS-DYNA*.

4.2.1 flexiCluP

The clustering module[6] *Flexible Clustering of Paramters (flexiCluP)* was developed to implement the clustering algorithm discussed in section 4.1. It is written in FORTRAN and can be compiled together with the main program which uses the clustering algorithm.

Code The history dependent clustering is called via the subroutine **hiDe_flexiCluP**. This subroutine takes as input an array of data points, the clustering limits and an array of integers that contains the current cluster numbers as needed for the history dependent approach. The outputs are an array filled with the evaluation result for each data point and an array of integers that contain the new cluster numbers.
The code also has a dummy function called **evaluation_function** where the evaluation function needs to be implemented. *flexiCluP* is implemented so that the user just needs to write the evaluation function for a single data point. To this end, it provides the input value, as well as the previous cluster integer, which corresponds to the previous internal state. As output it expects the results from the evaluation function. The new internal state of each cluster must be saved separately, as *flexiCluP* doesn't concern itself with the internal states of the clusters. Figure 4.4 gives a schematic overview of the subroutine.

4.2.2 LS-DYNA and MatCalc

As explained in subsection 3.2.2, *LS-DYNA* allows for a certain amount of FORTRAN user code to be implemented. Two subroutines[7], **thumat11** and **uctrl**, as well as the *flexiCluP* module, were used to implement the clustering and evalutation algorithm. The clustered data points were the temperature T, the grain size δ^G and the dislocation density ρ within the different FEM elements, and the evaluation function was the *MatCalc* computation. The *MatCalc* sessions were called by FORTRAN through console commands, and the internal states S were automatically saved within these persistent sessions. Figure 4.5 shows a flowchart of the implementation.

[6] A module in FORTRAN is similar to a library in other programming languages.
[7] A subroutine is a function with no explicit return value in FORTRAN, similar to a *void* function in C.

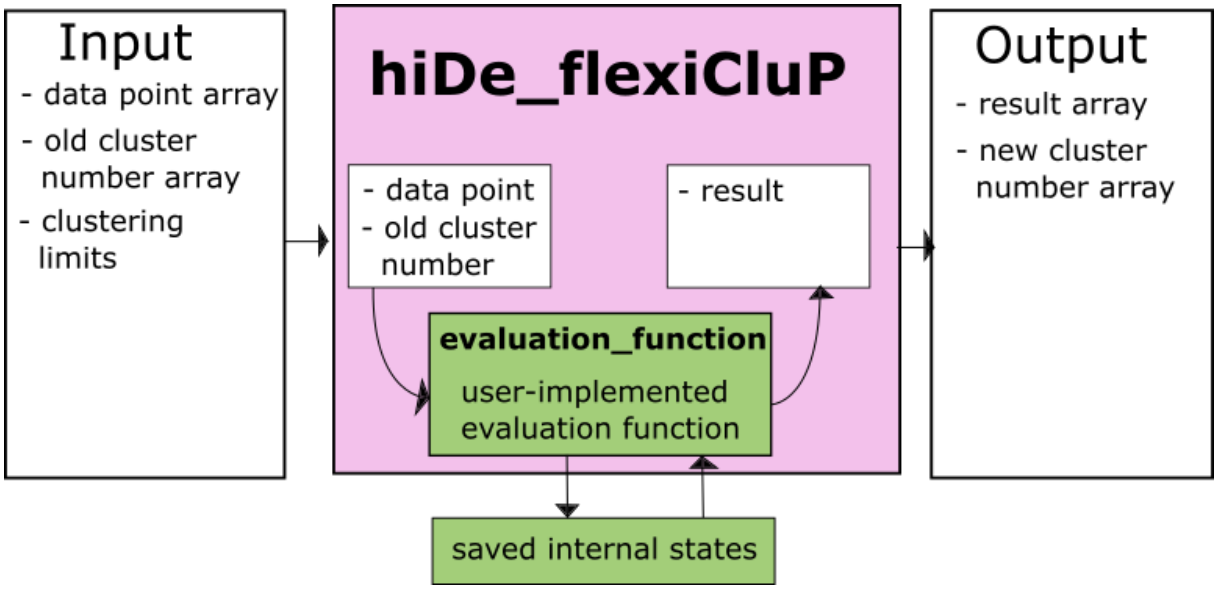

Fig. 4.4: Schematic overview of the subroutine **hiDe_flexiCluP** and the function **evaluation_function** within.

thumat11 The subroutine **thumat11** (*thermal user material 11*) is one option to implement user-defined material models in *LS-DYNA*. It was used for the recovery and recrystallization models, as described in subsection 3.2.2. After calculating the microstructure, the dislocation density ρ, the grain size δ^G and the temperature T were saved onto a global array, which had an entry for each element. This array had a global scope, so its values persisted between function calls, which made it possible to store and access the individual element variables outside of **thumat11**.

uctrl1 The subroutine **uctrl1** (*user control 1*) is called once per timestep and can be modified by the user. Within **uctrl1**, the subroutine **hiDe_flexiCluP** was called. The inputs were the values ρ, δ^G and T from the global array, then another global array containing the previous cluster numbers and finally the clustering limits, which were constant throughout the simulation. The outputs were the precipitation parameter χ, the precipitate density N_{pr} and the precipitate radius r_{pr}. These were saved into a global array, so that they could be accessed by the individual elements in **thumat11**.

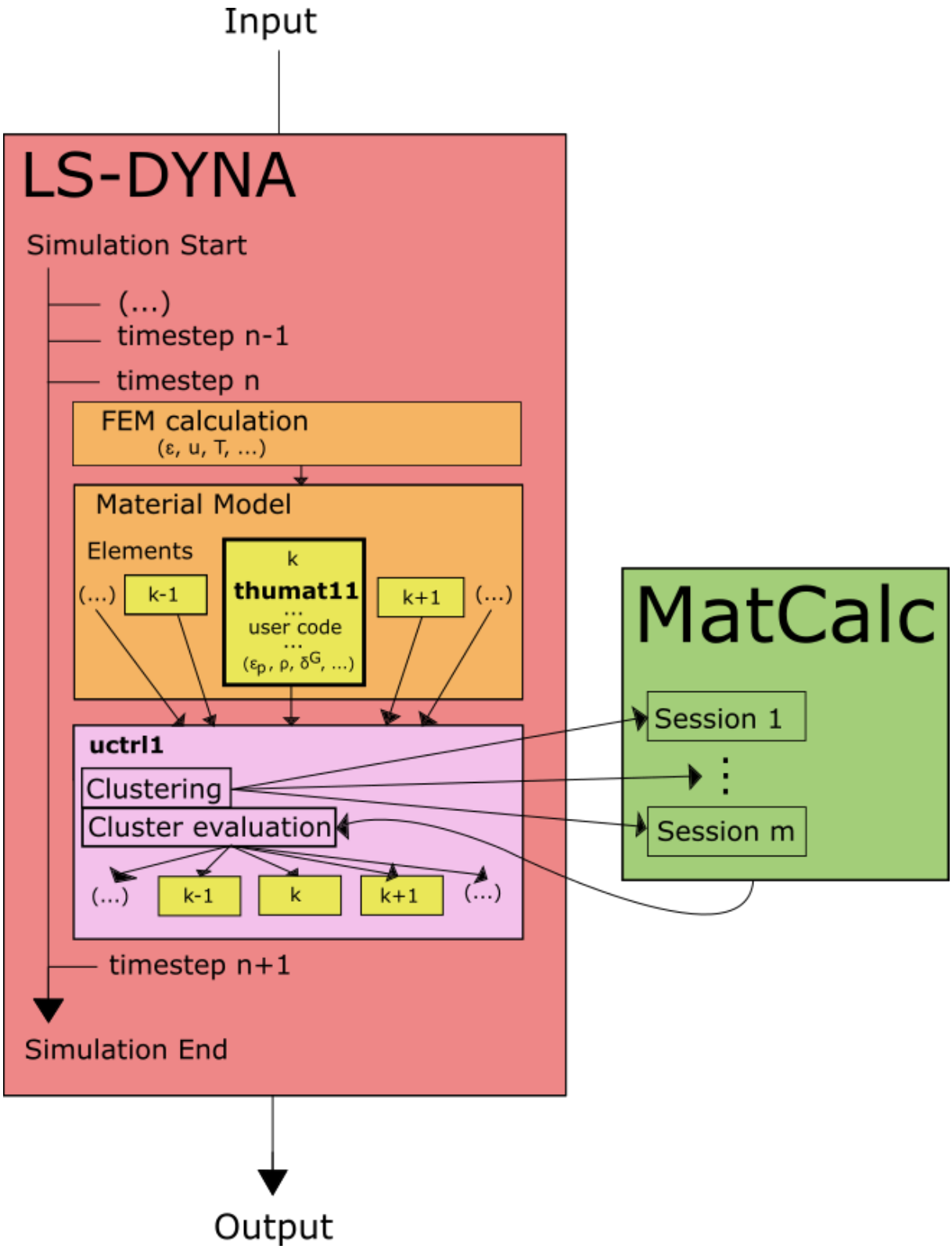

Fig. 4.5: Flowchart of the thermal *LS-DYNA* simulation with clustering: For each timestep, the program applies the FEM approach described in section 2.3. The material model is implemented in the subroutine **thumat11**, which is called once for each element. Afterwards, the subroutine **uctrl1** is called once per timestep by *LS-DYNA*. It clusters the previously gathered data and sends it to the *MatCalc* sessions. After returning the results, they are distributed back to the elements.

Chapter 5

Results

In this chapter the results of the experiment, as well as the simulations are presented. To recap, there were 7 experiments conducted, which are detailed in table 3.2, but only 2 simulations, one for experiment No. 1 and one for experiments No. 2-7. The reason was that experiments No. 2-7 have the same parameters, but were just stopped at different temperatures, so one simulation going through all temperatures is sufficient to model them. Experiment and simulation both were divided into three stages, which consist of the pre heat treatment, deformation and post heat treatment. This is depicted in figures 3.1 and 3.4. However, to gauge the impact of the clustering and coupling, the post heat treatment was simulated with 3 variants for each simulation, as shown in table 3.3.
The pre heat treatment consisted only of a single calculation in *MatCalc* for experiment No. 1 and experiments No. 2-7 each. Since there was no original code or model involved, it has been omitted here.
The program *LS-PrePost* was used to visualize the simulation results, and the specimen was (virtually) cut in half to get a better sense of the distribution of values. The only exception is the compression in figure 5.1, which was visualized in shaded 3D.

5.1 Deformation

Macroscopic Deformation The aluminium cylinder was compressed at a strain rate of 1 s^{-1}, which can be seen in figure 5.1. This deformation was the same for experiments No. 1-7. The corresponding plastic strain of a cross section at the end of the deformation in experiment No. 1 is shown in figure 5.2. It is highest in the center and at the buckled edges, while next to the impactors the part barely deforms.
The apparent cross shape is a typical shape for this kind of experiment, and it is called the forging cross (see chapter 6.3 in [15]). Since the plastic strain has a huge influence on the dislocations, this forging cross is like a fingerprint of the deformation process and it can be recognized in the microstructure further down the line.

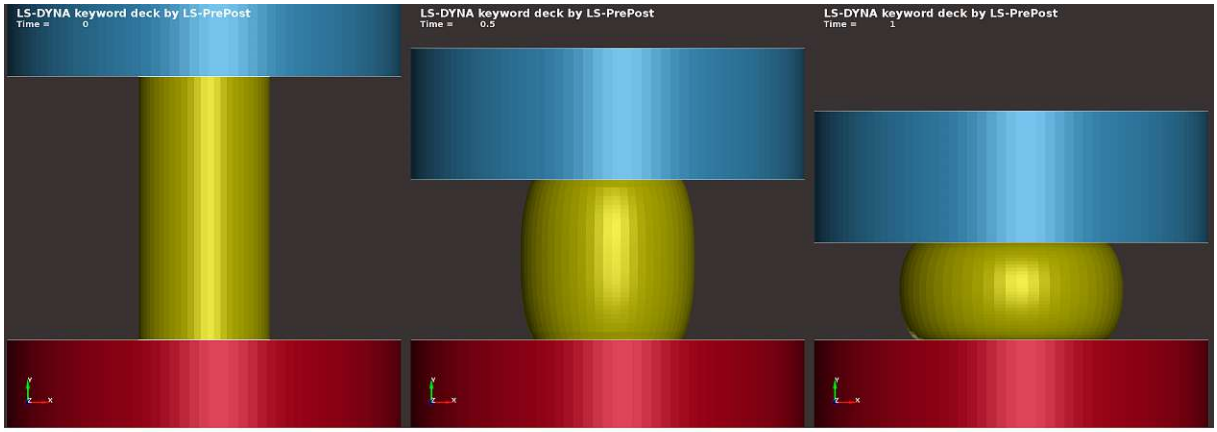

Fig. 5.1: Timesteps $t = 0\ s$ (a), $t = 0.5\ s$ (b) and $t = 1\ s$ (c) in the compression test simulation at strain rate $1s^{-1}$. The impactors are depicted in blue and red, and the specimen is depicted in yellow. All are visualized in shaded 3D.

Microstructure The dislocation density of a cross section at the end of the compression test simulation is given for experiment No. 1 and experiments No. 2-7 in figure 5.3 (see table 3.2 for the process parameters of the experiments).

Even though the deformation was identical, the different experiments already exhibit a different dislocation density. Experiment No. 1 had a pre heating time of 10 s, while experiments No. 2-7 had 600s. This means that for experiment No. 1, fewer solute atoms could dissolve in the matrix and thus their concentrations were lower. Therefore the dislocation relaxed faster through climb recovery, since the solute drag was lower. This effect is incorporated in the addition of the precipitation parameter χ in equations 2.8 and 2.9 which affects the recovery in equation 2.10.

The dislocation density of the deformation step is passed on to the post heat treatment simulation, so we can see that the difference in pre heat treatment times has a significant impact on the rest of the simulation.

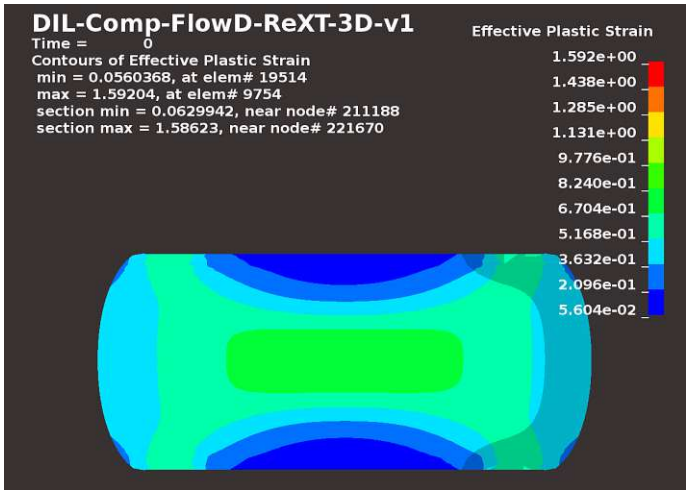

Fig. 5.2: The dimensionless plastic strain after the deformation for experiment No. 1.

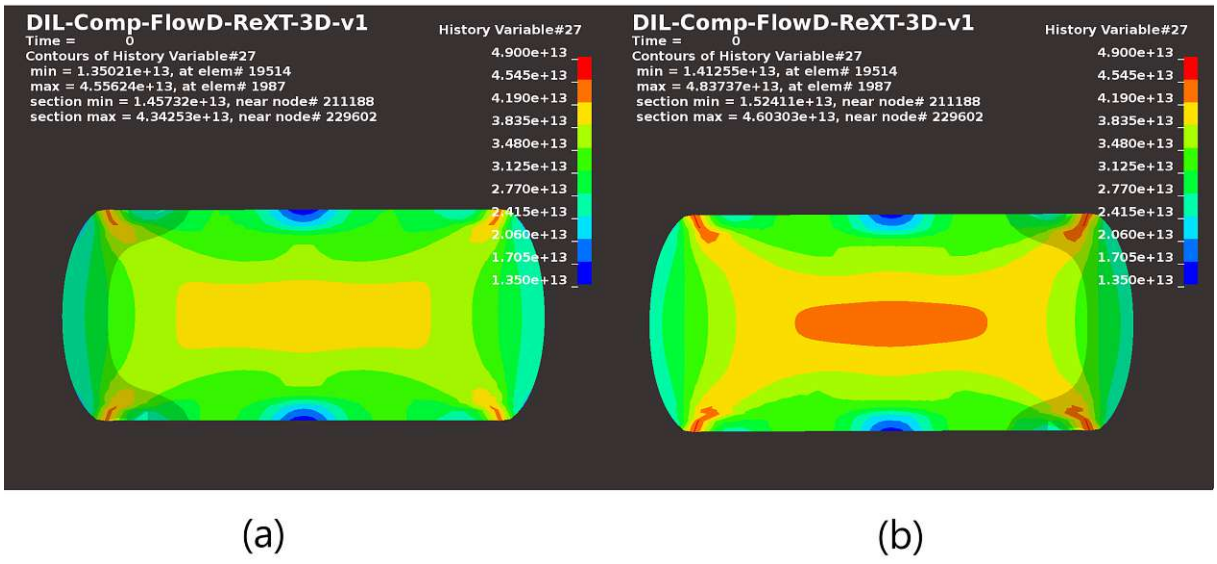

Fig. 5.3: Dislocation densities in $[m^{-2}]$ at the end of the compression step before the post heat treatment for (a) experiment No. 1 and (b) experiments No. 2-7.

5.2 Post Heat Treatment Experiment No. 1

Experiment Parameters As detailed in table 3.2, during the pre heat treatment the specimen was heated to 470 $°C$ at a rate of 15 $°C/s$ and kept there for 10 s. Afterwards it was deformed at a strain rate of 1 s^{-1} at the same temperature. Finally it was kept at 470 $°C$ for 600 s during the post heat treatment and then quickly cooled down to 20 $°C$ at 35 $°C/s$.
The simulation was done in three variants, which are given in table 3.3. They consist of variant 1 with constant precipitation variables, variant 2 where one precipitation calculation is done for the average of the whole specimen and variant 3 with full coupling and clustering as described in chapter 4.

Variant 2 Figures 5.5 and 5.6 show a strange behaviour for the simulation variant 2. Since variant 2 is a mix between variants 1 and 3, the unique features of variant 2 seemed like an artifact of the simulation. This is especially true since variants 1 and 3 have almost identical values. After some investigation, it turned out that the *MatCalc* session for the precipitation kinetics failed to launch. This resulted in erroneous precipitation values and thus caused the false results.

Grain Size The actual experiment specimen can be seen in figure 5.4. It has been cut open and the grain structure has been highlighted with Barker's reagent.
For comparison, the simulated grain size is given in figure 5.5. At the center, the grain size is larger than at the top and bottom, which is in line with the experimental measurement. However, variants 1 and 3 suggest even larger grains towards the edges of the specimen, which are not observed in reality. One reason for this discrepancy could be that the simulation is not finely tuned enough and the edges recrystallize without doing so in reality. There could also be a problem with the recrystallization model, for example the fact that it treats grains as spherical, which is not the case in reality (see figure 5.4).
Coming back to variants 1 and 3, their values are very similar. This suggests that the constant precipitation variables in variant 1 were a good approximation. Therefore the precipitation kinetics during the post heat treatment did not have a large influence on the grain structure.

Recrystallized Volume Fraction The evolution of the recrystallized fraction within the simulation can be seen in figure 5.6 for all three simulation variants.
We can see that recrystallization starts relatively early, since the specimen is already heavily recrystallized after 85 s heat treatment and fully recrystallized at the end of the

treatment. This result is hard to validate, since the specimen in figure 5.4 only gives the grain size but not whether these grains are already recrystallized or not.
Further, we see again that variants 1 and 3 are very similar. This supports the suspicion that the precipitation kinetics did not play a huge role in this simulation.

Clustering In figure 5.7 we see for variant 3 the cluster distribution within the specimen at the end of the simulation, as well as the distribution of the precipitate density N_{pr} and the precipitate radius r_{pr}. Both values were calculated within the specimen according to the clustering, as described in chapter 4.
The cluster distribution shows that the clustering algorithm has worked as intended, as the clusters are distributed along regions with roughly the same microstructure. This can be checked when comparing the pattern to figures 5.5 and 5.6.
However, the distributions of N_{pr} and r_{pr} after 600 s show that the precipitation kinetics were the same almost everywhere. Only after the cooling at 613 s there emerges a difference in the precipitation variables. This is another strong indicator the precipitation kinetics during this experiment were neglegible.

Discussion Altogether we see that the simulation can recreate the forging cross pattern and have a corresponding grain structure evolution which in the center is similar to the observed grain structure in the experiment. On the other hand, the accuracy of the grain structure towards the edges can still be improved. The simulation also shows recrystallization almost everywhere, which doesn't seem to conform with the experiments at the edges.
Finally, the clustering algorithm seems to work as intended, but there is much evidence that the precipitation kinetics did not play a large role during this experiments post heat treatment.
Variant 2 of the simulation failed, but since it is a mix of variants 1 and 3 and their results were similar, we can safely assume that it would have produced matching values.

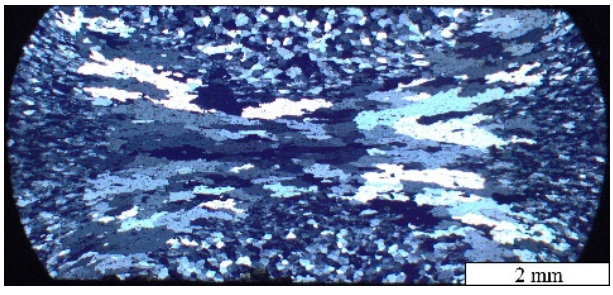

Fig. 5.4: Section of the specimen anodized with Barker's reagent to make the microstructure visible.

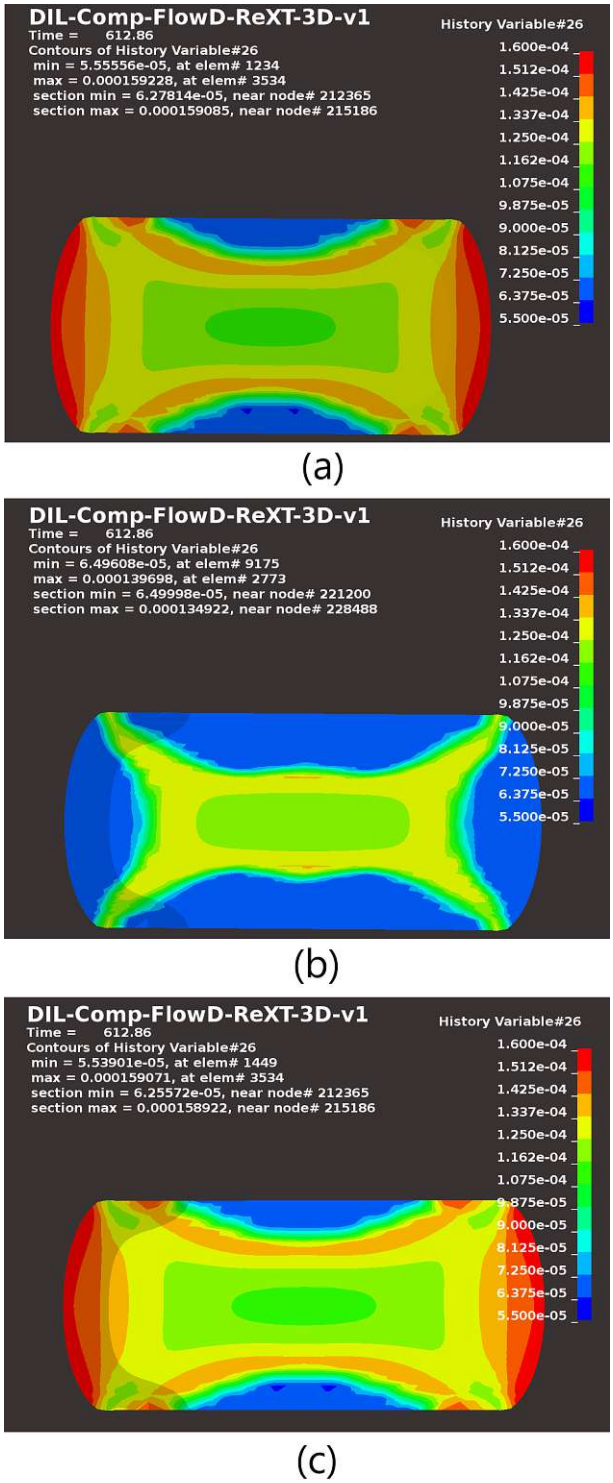

Fig. 5.5: Grain size within the specimen for (a) simulation variant 1, (b) simulation variant 2 and (c) simulation variant 3. The values are taken at the end of the simulation after cooling down to 20 $°C$.

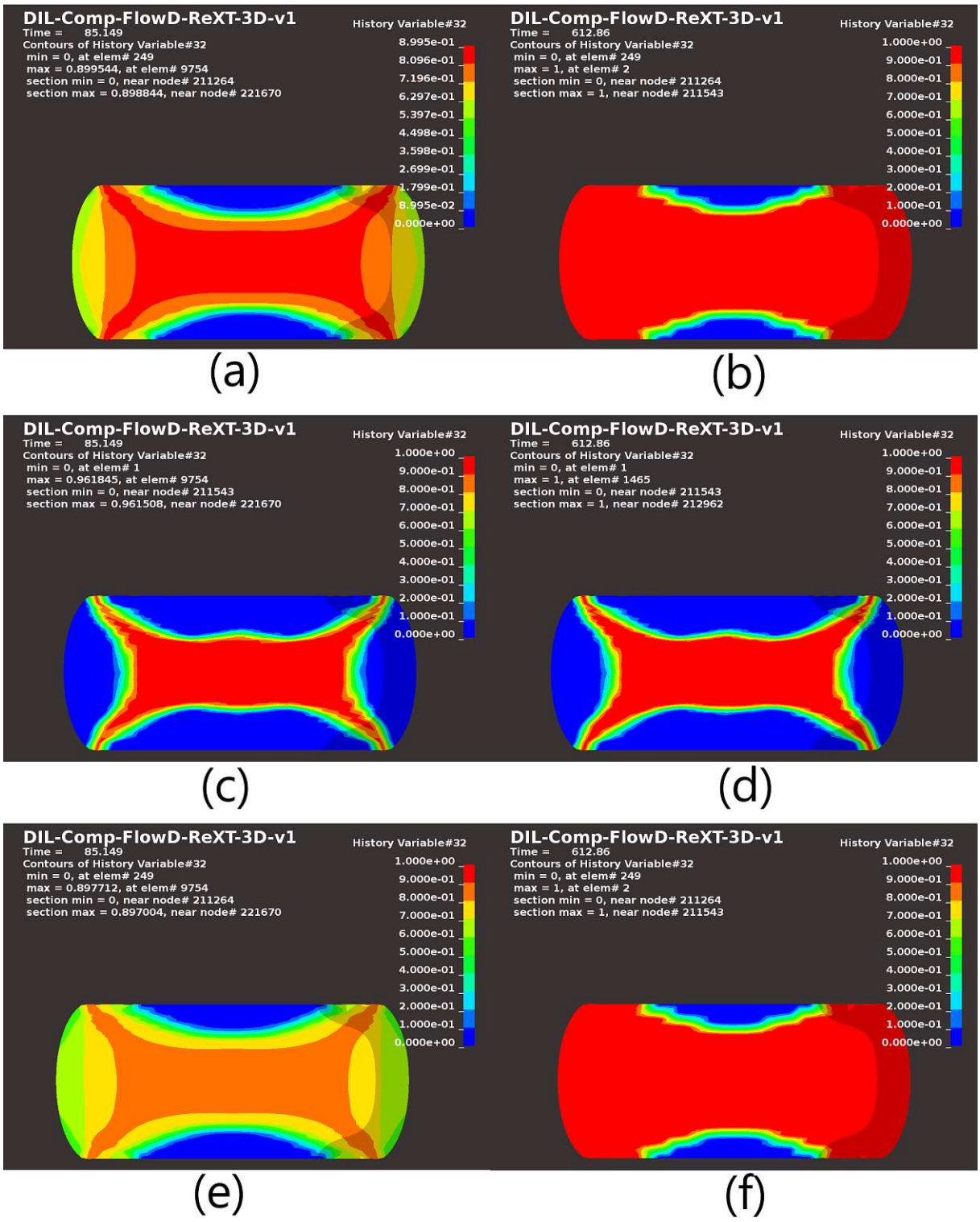

Fig. 5.6: Recrystallized volume fraction for (a),(b) simulation variant 1, (c),(d), simulation variant 2 and (e),(f) simulation variant 3. The left column (a),(c),(e) is taken after 85 *s* and the right column (b),(d),(f) is taken at the end after 613 *s*.

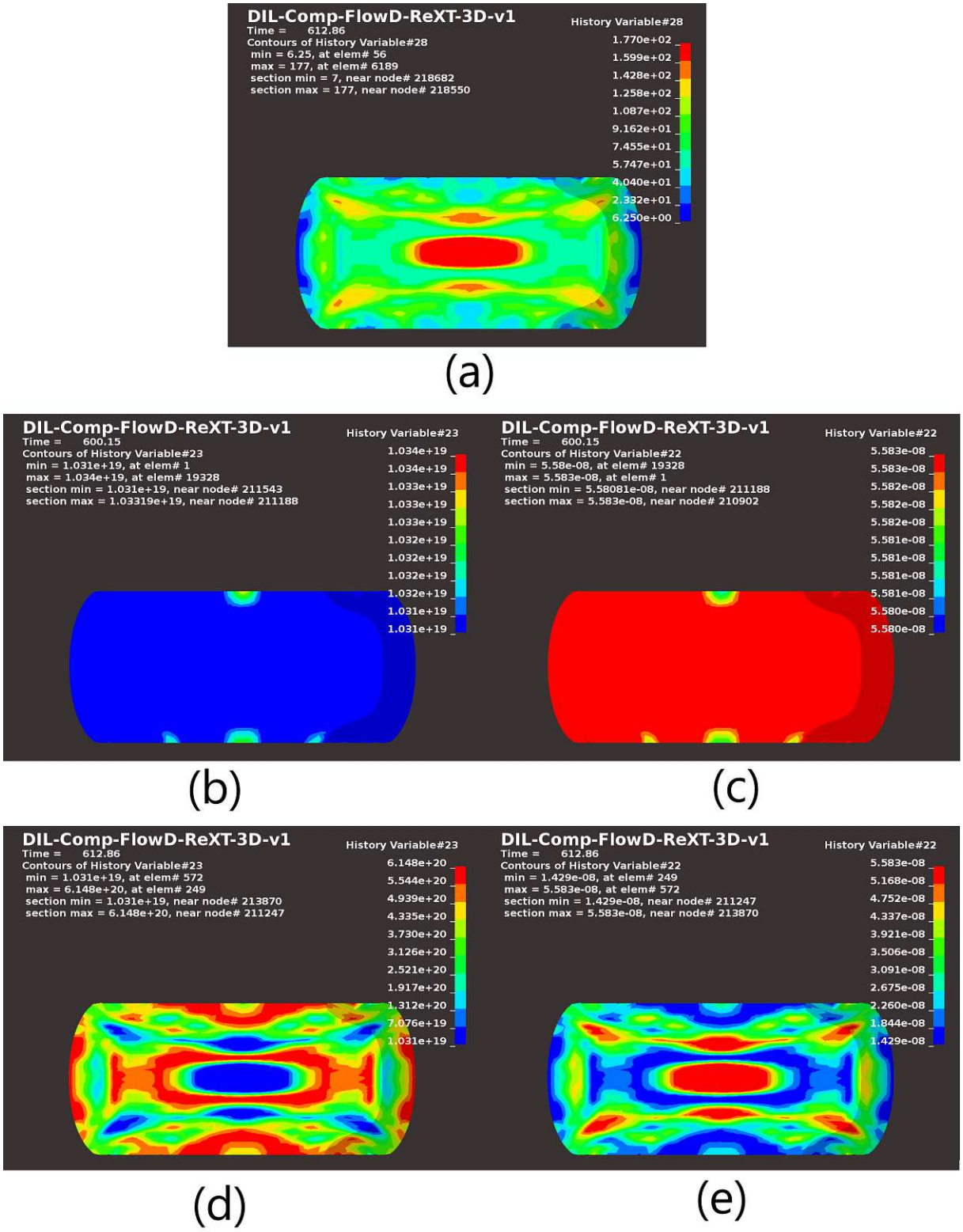

Fig. 5.7: Variant 3 of the post heat treatment simulation. The top picture (a) shows the cluster distribution in the end at 613 s.
In the middle row are (b) the precipitate density N_{pr} and (c) the mean precipitate radius r_{pr} at 600 s and 470 $°C$, just before cooling down.
In the bottom row are again (d) N_{pr} and (e) r_{pr}, but this time at 613 s and 20 $°C$, after cooling down.

5.3 Post Heat Treatment Experiments No. 2-7

Experiment Parameters As detailed in table 3.2, during the pre heat treatment the specimen was heated to 470 $°C$ at a rate of 15 $°C/s$ and kept there for 600 s. Afterwards it was deformed at a strain rate of 1 s^{-1} at the same temperature. Then it quickly cooled down to 20 $°C$ at 35 $°C/s$ and afterwards reheated at a rate of $7,1$ $°C/min$. The specimens No. 2-6 were then subsequently removed and inspected at temperatures 450 $°C$, 460 $°C$, 470 $°C$, 480 $°C$ and 490 $°C$ respectively. The final specimen No. 7 was kept at 490 $°C$ for another 33 min.
All these experiments were captured with one simulation, which was just evaluated multiple times at the relevant temperatures.
Like for experiment No. 1, the simulation was done in three variants, which are given in table 3.3. They consist of variant 1 with constant precipitation variables, variant 2 where one precipitation calculation is done for the average of the whole specimen and variant 3 with full coupling and clustering as described in chapter 4.

Grain Size The actual experiment specimens can be seen in figure 5.8. They have been cut open and the grain structure has been highlighted with Barker's reagent. In addition, the grains at the center have been measured for experiment No. 7 with an average radius of about 103 μm.
The simulated grain size of experiment No. 7 is given in figure 5.9 for all three variants. Each has a grain size of about 100 μm at the center, which is in line with the experimental measurement. Like with experiment No. 1, the simulated grain size gets larger further to the edges, whereas it stays relatively the same or gets even smaller in the actual experiments. Again, this suggests room for improvement with either the fitting parameters or the model itself.
We can see a significant difference in the grain size distribution between the three variants. Variant 1 generally has the largest grains, while variant 2 has slightly smaller ones, and variant 3 has the smallest grains. Thus, variant 3 is closest to reality, although its predictions are still not accurate at the edges. This difference between the variants behaviour suggests that the precipitation kinetics played an important role during the post heat treatment of experiments No. 2-7. It also demonstrates that variant 2 can be seen as a mixture of variants 1 and 3, since its values lie between the other two.

Recrystallized Volume Fraction The evolution of the recrystallized fraction within the simulation can be seen in figures 5.10 for variant 1, 5.11 for variant 2 and 5.12 for variant 3. All figures have a depiction for each of the experiments No. 2-7.

We can see that for variant 1, the specimen is already fully recrystallized in experiment No. 2 which corresponds to a final temperature of 450 $°C$. This is completely wrong when compared to figure 5.8, where we can see that recrystallization started between 460 $°C$ and 470 $°C$.
For variant 2, the recrystallization starts somewhere between 460 $°C$ and 470 $°C$, which is correct when compared to figure 5.8. However, the simulation of experiment No. 5 has already fully recrystallized, which is not the case for the real specimen. As with the grain size, this discrepancy shows that there is room for improvement in the model.
Variant 3 shows a similar behaviour to variant 2, with the difference being that the recrystallization happens slightly earlier. This means that variant 3 also gets the recrystallization timing approximately right, but afterwards there are some problems.
The differences in recrystallization timings between the variants is further evidence that for experiments No. 2-7, the precipitation kinetics played an important role. It is, however, hard to determine whether variant 3 was an improvement over variant 2 since both results differ from the actual experiments.

Clustering Figure 5.13 shows the clustering at the end of the simulation for variant 3, as well as the precipitate density N_{pr} and the precipitate radius r_{pr} at 470 $°C$, which corresponds to experiment No. 4. Both values were calculated within the specimen according to the clustering, as described in chapter 4.
The cluster distribution indicates that the clustering worked as intended, just like with experiment No. 1, and the distribution of N_{pr} and r_{pr} shows that the calculation of *MatCalc* for each cluster worked as well.
In addition the values for N_{pr} and r_{pr} provide further proof that the precipitation kinetics were varying within the specimen and thus the precipitation coupling was necessary.

Discussion Again, the simulations manage to produce a relatively accurate value for the grain size in the center of the specimen. However, as with experiment No. 1, the grain size towards the edges is not accurate.
We can also see that for experiments No. 2-7, a coupling to the precipitation kinetics is necessary to produce sensible recrystallization results. This is evidenced by the fact that variant 1 recrystallizes much too early. It is, however, hard to say whether variant 3 is a significant upgrade over variant 2, since it is complicated to measure the recrystallized fraction in real specimens.
Regarding the clustering alorithm, the simulations again provide good evidence that it is working as intended and calculating the precpitation kinetics with *MatCalc* for each cluster.

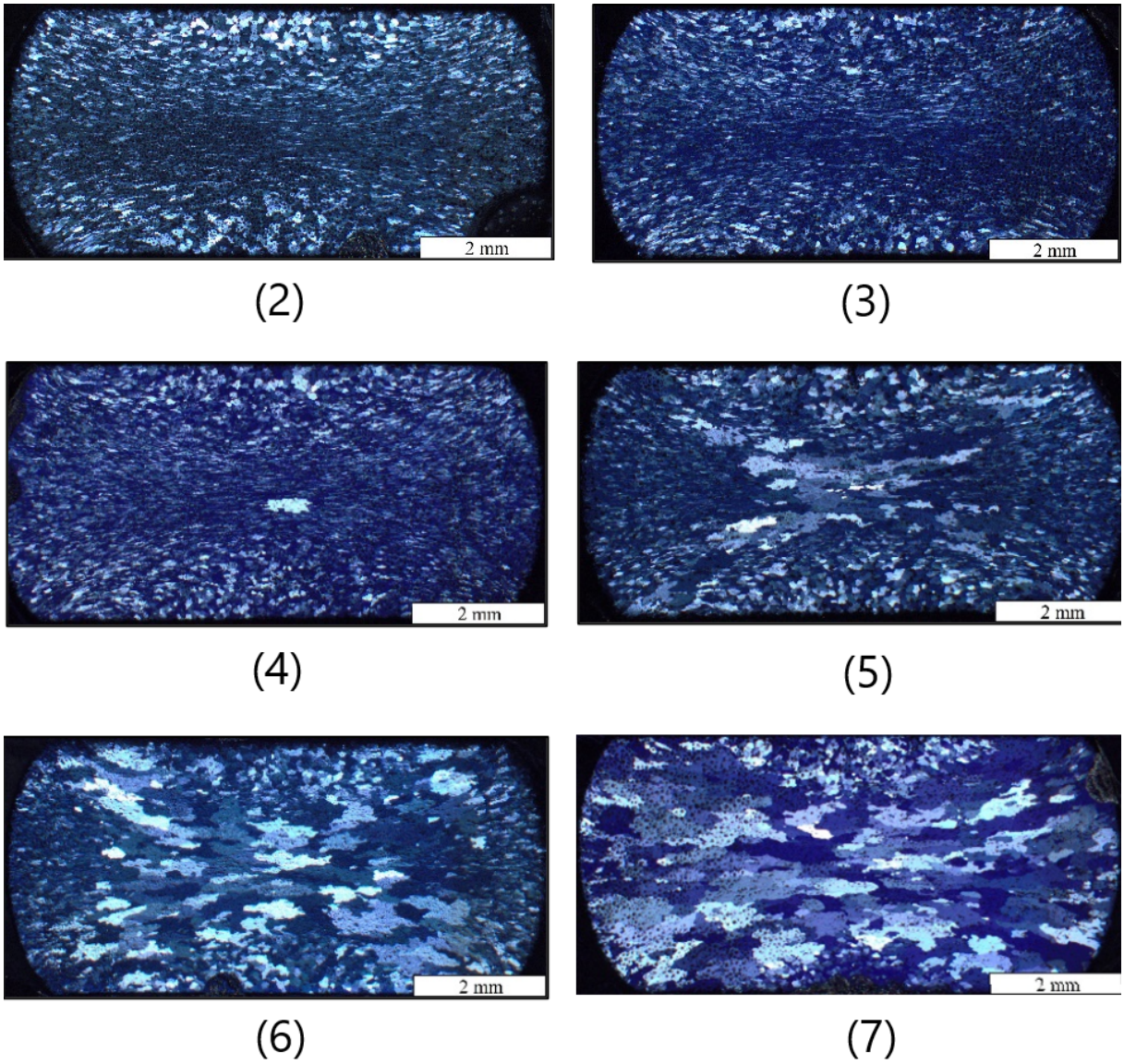

Fig. 5.8: Sections of specimens anodized with Barker's reagent to make the microstructure visible. The specimens were cooled down after heating up to (2) 450 °C, (3) 460 °C, (4) 470 °C, (5) 480 °C, (6) 490 °C and (7) after an additional 33 min of heat treatment at 490 °C.

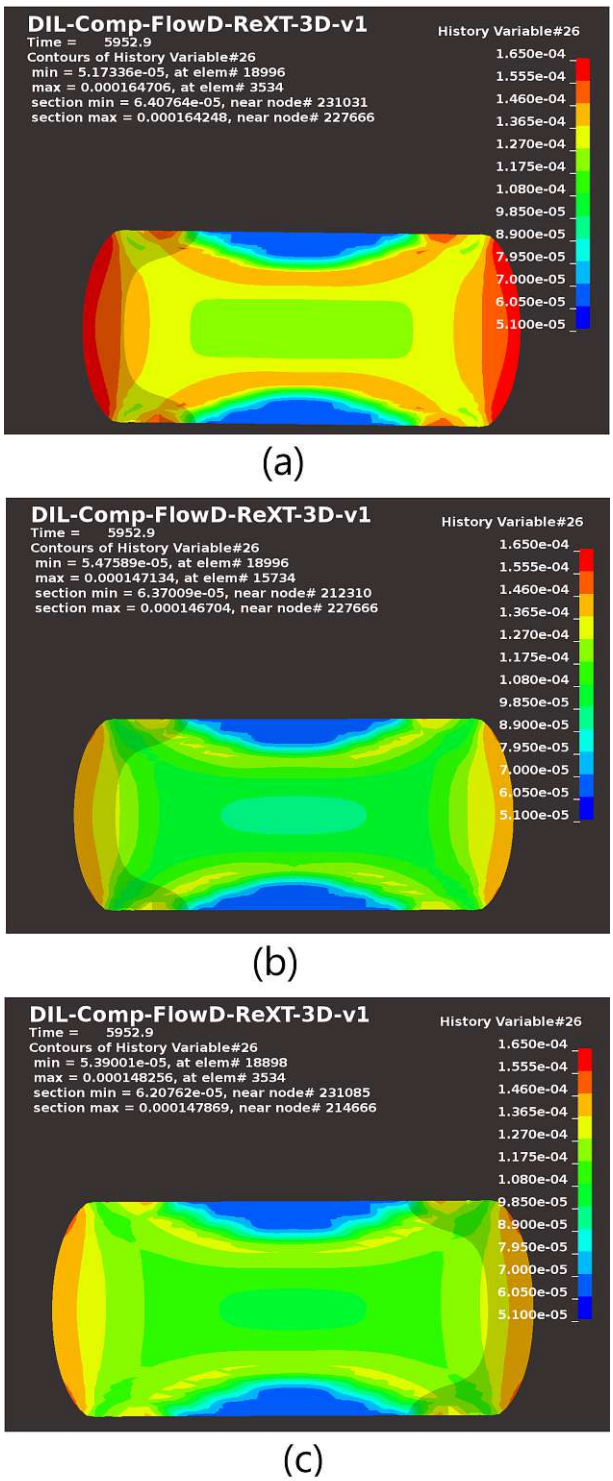

Fig. 5.9: Grain size within the specimen for experiment No. 7 and (a) simulation variant 1, (b) simulation variant 2 and (c) simulation variant 3. The values are taken at the end (after holding 490 $°C$ for 33 $mins$) of the simulation.

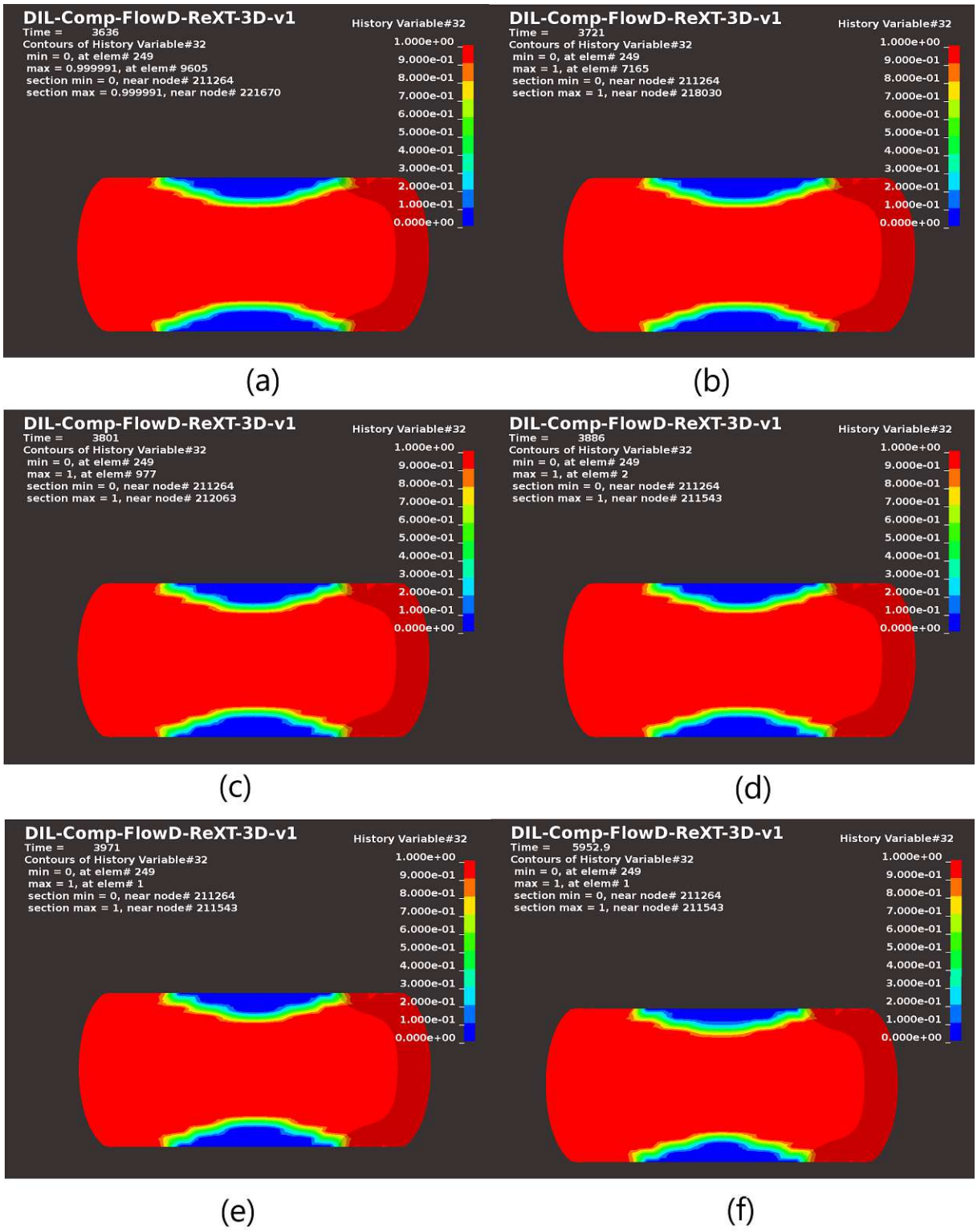

Fig. 5.10: Variant 1 of the post heat treatment simulation. The recrystallized volume fraction is shown at (a) 450 °C, (b) 460 °C, (c) 470 °C, (d) 480 °C, (e) 490 °C and (f) after an additional 33 min of heat treatment at 490 °C.

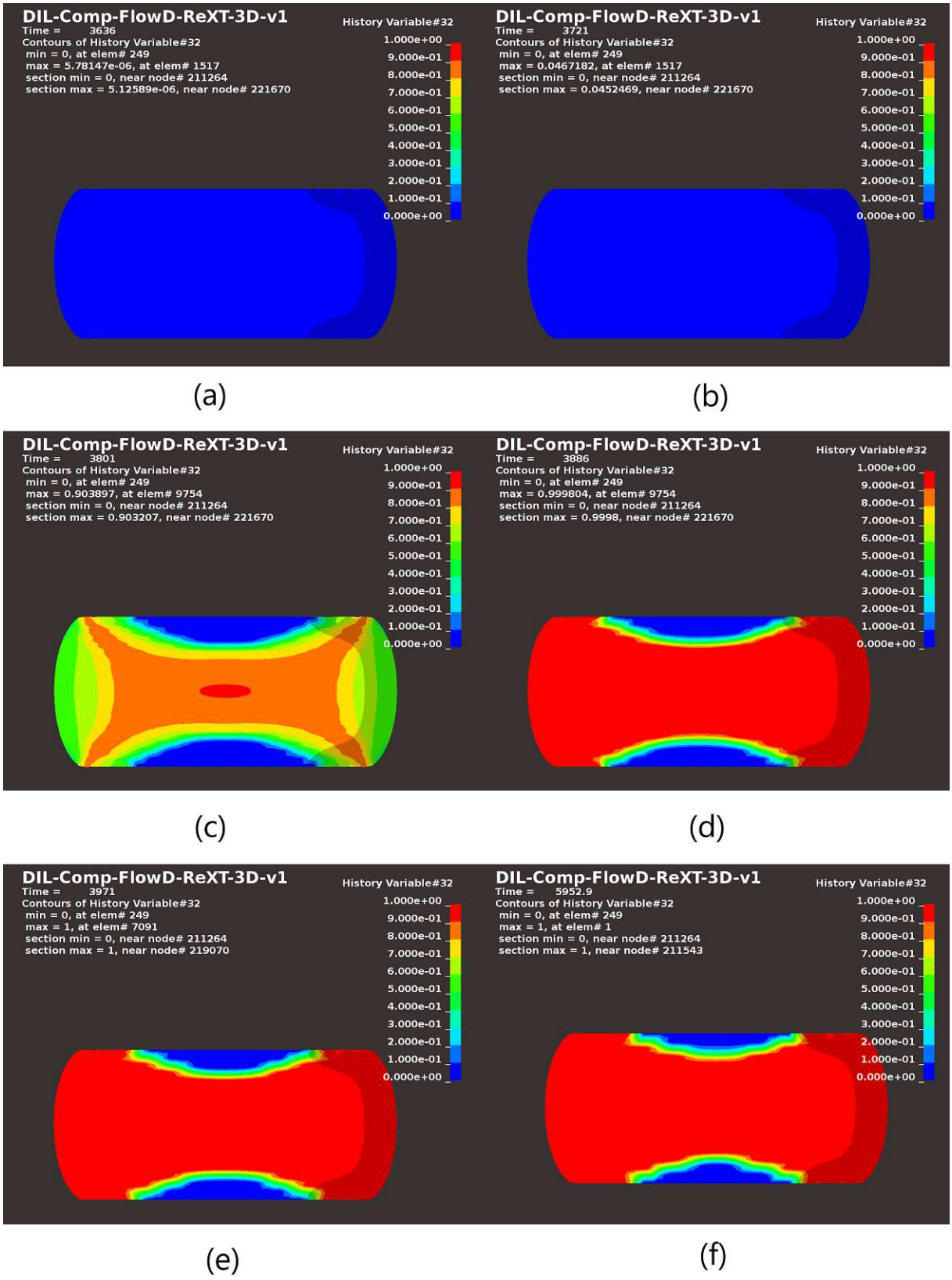

Fig. 5.11: Variant 2 of the post heat treatment simulation. The recrystallized volume fraction is shown at (a) 450 °C, (b) 460 °C, (c) 470 °C, (d) 480 °C, (e) 490 °C and (f) after an additional 33 min of heat treatment at 490 °C.

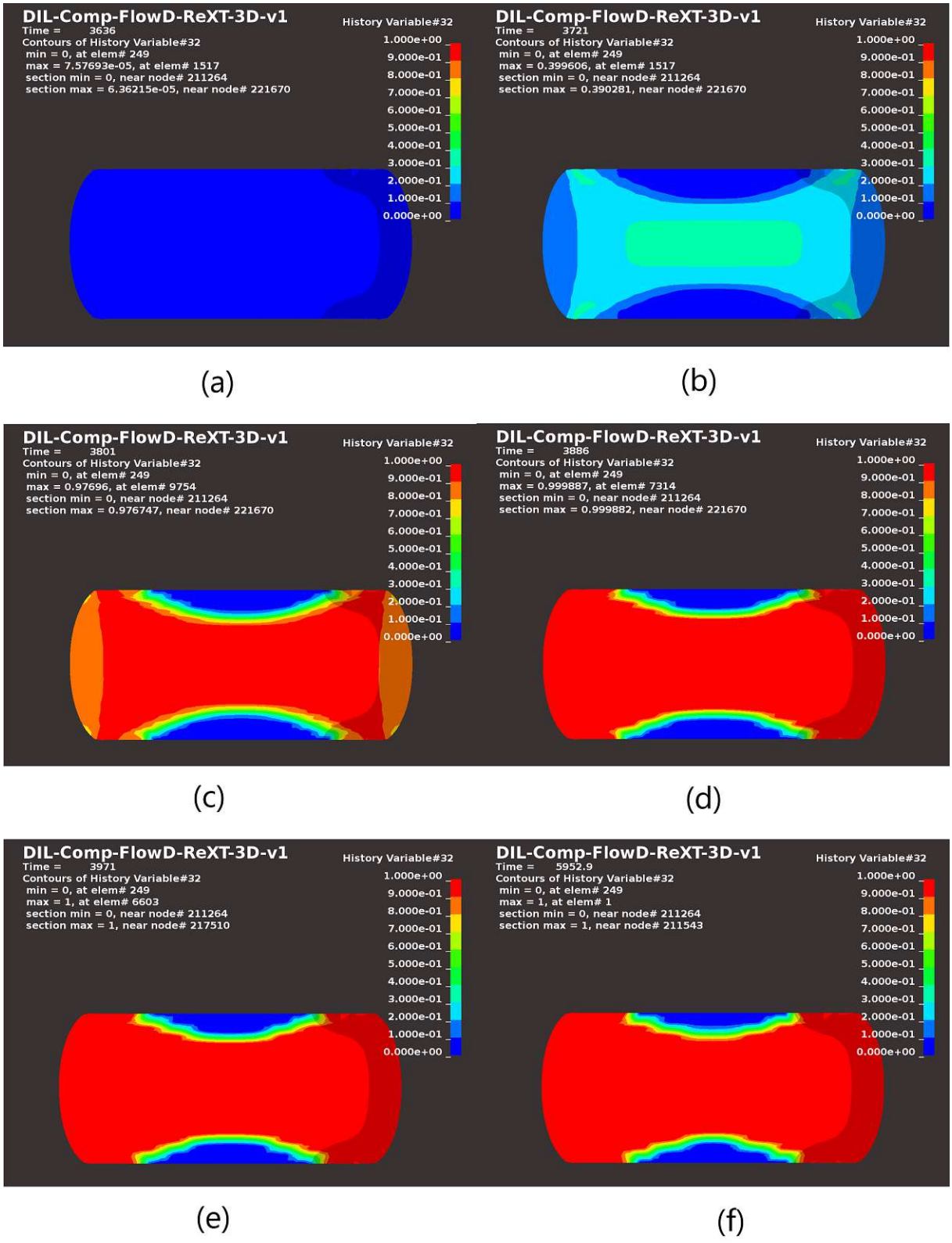

Fig. 5.12: Variant 3 of the post heat treatment simulation. The recrystallized volume fraction is shown at (a) 450 °C, (b) 460 °C, (c) 470 °C, (d) 480 °C, (e) 490 °C and (f) after an additional 33 min of heat treatment at 490 °C.

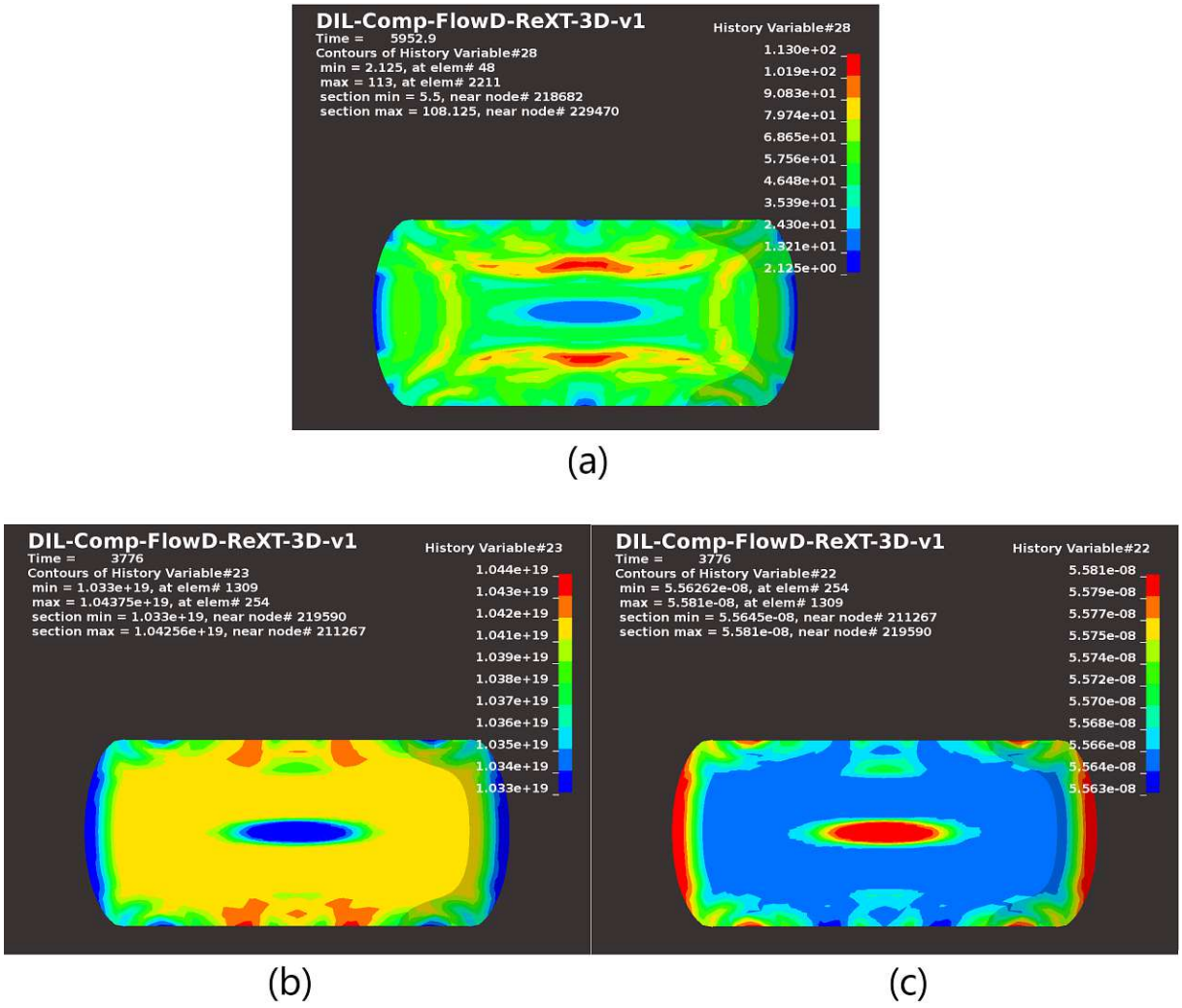

Fig. 5.13: Variant 3 of the post heat treatment simulation. The figures show (a) the cluster distribution and the resulting distributions of (b) the precipitate density N_{pr} in $[m^{-3}]$ and (c) the mean precipitate radius r_{pr} in $[m]$ at 470 $°C$.

Chapter 6

Summary and Outlook

6.1 Summary

This thesis aimed to improve the simulation of aluminium during deformation and heat treatment. The previous multiscale model already combined the grain structure evolution and recrystallization with macroscopic parameters such as temperature and the deformation, and it was expanded by a computation of the precipitation kinetics. In chapter 2, the physical theory behind the various models was explained. The MD^{2}M model in particular was studied and sourced in section 2.1, and a guideline to its numerical implementation was given in the appendix. This documentation will facilitate maintenance and further development to the model and the code.

The expanded model was implemented numerically through a special algorithm which coupled the existing implementation to a precipitation kinetics program. For this, the simulation used the FEM solver *LS-DYNA* and the precipitation kinetics calculation program *MatCalc*. Both are detailed in section 3.2, while their implementation is described in section 3.3.

The coupling algorithm was developed during this thesis and it uses a clustering approach to improve the computation speed of the coupling. During the FEM simulation, in each timestep the algorithm gathers all elements with similar grain structure into clusters and computes the precipitation kinetics only once for each cluster. Without the algorithm, the simulation would probably have taken multiple weeks or even more than a month to complete. With the algorithm, the calculation time was a few days. Further, the clustering algorithm was designed to be general enough so that it is not bound to this specific problem. It can be used for other multiscale applications to reduce computation time. The clustering algorithm and its implementation are given in chapter 4.

In the end, various multiscale simulations of hot compression tests were performed and compared to available experiments to validate the multiscale model and the clustering

algorithm. The 7 experiments all consisted of a pre heat treatment, a uniaxial compression step and a post heat treatment. Every experiment was simulated in three variants to gauge the impact of the precipitation coupling and the clustering algorithm. The first variant used no coupling and just had static precipitation variables throughout the simulation. The second variant averaged the values of all elements within the specimen and calculated the precipitation kinetics once in each timestep. Finally, the third variant used the full coupling available through the clustering algorithm. The experiments are described in section 3.1, while the results and discussion thereof is given in chapter 5.

The results in section 5.3 show that there is a significant difference in the recrystallization behaviour between variant 1 and variants 2 and 3. The fully coupled and clustered variant 3 fits reasonably well for all experiments, while variant 1 recrystallized too early for experiments No. 2-7 (see figure 5.6). Variant 2 had failed for experiment No. 1 (see figure 5.6), but worked well for experiments No. 2-7. Altogether, the results show that the clustering algorithm works as intended. They also suggest that a coupled simulation is necessary, since the precipitation kinetics during the heat treatment influenced the recrystallization behaviour for the simulation of experiments No. 2-7. It is important to note here that the impact of the successful coupling is likely even greater for more complex geometries and processes. However, their simulation would have exceeded the bounds of this thesis.

6.2 Outlook

6.2.1 Model Refinement

Current Situation The simulated experiment was very simple and is more of a proof of concept. The model and the parameters could be refined to obtain more accurate results.

Simulation Parameters There needs to be a parameter study for choosing the optimal clustering limits and reaffirm the various fitting parameters used. This helps to improve the predictive accuracy of the model, e.g. for the exact values of the dislocation density. After that, there could be other types of experiments with either larger specimens or with different procedures such as hot-rolling. These could show the range of validity for the model.

Improved Recrystallization Models A major shortcoming of the recovery and recrystallization models described in subsections 2.1.3 and 2.1.4 is the missing of a grain growth equation after recrystallization. This can be seen in equation 2.31, which stops the grain growth of the recrystallized grains after full recrystallization, that is $X^{rex} = 1$. However, in reality we observe continuous grain growth, even after recrystallization, as depicted in figure 2.5 and as seen in the experiments in figure 5.8. This effect could be implemented similar to equation 2.17, but only coming into play after a sufficiently high recrystallization, e.g. $0.9 < X^{rex}$.

6.2.2 Code Performance

Current Situation The clustering algorithm managed to make the coupled simulations possible. However, the simulation times were still too high, taking as long as 138 hours for one simulation of experiments No. 2-7 variant 3. For application in more complex simulations, these times have to be cut down drastically. This could be achieved by the following two methods.

Reworked LS-DYNA Code The current *LS-DYNA* implementation is a single-threaded, sequential simulation, which doesn't utilize potential parallel processing capabilities. Thus, a great way to reduce the simulation time is to rewrite the code for Message Passing Interface (MPI).
MPI is a parallel processing standard for FORTRAN and C with premade libraries and easy portability for different systems. It allows the user to divide the numerical work onto

different cores that run parallel and pass data between each other. Also, this standard is supported by *LS-DYNA*.

Reworked MatCalc Coupling A major timesink of the simulation is the reading and writing of *MatCalc* outputs to files. Potential solutions suggested by the developers of *MatCalc* are to either incorporate it directly into the code via library, or to pass data via a socket[1]. Both methods would drastically reduce call times for *MatCalc*, but their implementation is rather complicated.

6.2.3 Clustering Algorithm

Current Situation The clustering algorithm is a potent way to gather similar data points and reduce the computational complexity of their evaluation. Still, it could be improved by adding functionalities or refining existing ones. The improvements described in this subsection were not necessary for the coupling of *LS-DYNA* and *MatCalc*, but they would be useful for other applications of the clustering algorithm.

Dynamic Limits When looking at the evaluation function, there may be areas of input values where even big differences in the data points have very similar outputs. On the other hand, there may be areas where even the slightest change in input can result in a drastically different output. At the moment, there is one set of limits λ that stay constant over the whole input area. Therefore it is necessary to fit the limits to the most sensitive input area, which means smaller limits and more clusters, which increases computational complexity.
However, if the sensitive areas can be estimated beforehand, the limits might become dependent on the inputs of the weighted norm $d_w^\lambda(\boldsymbol{x}, \boldsymbol{y})$, that is $\lambda = \lambda(\boldsymbol{x}, \boldsymbol{y})$. This means that in areas where the output of the evaluation function rapidly changes, the limits can be sufficiently small, while in other areas where the output function is nearly constant, the limits can be larger to accommodate this fact.

Interpolated Evaluation The current algorithm evaluates each cluster once to get the final results. This is equivalent to keeping the evaluation function constant within that cluster. Another possibility is to interpolate the evaluation function within one cluster by evaluating at more than one data point per cluster. This allows for larger clusters, since the evaluation within each cluster is more accurate.

[1] Sockets are a way for programs to communicate directly via RAM, which is a lot faster than communicating via files on the hard drive.

Appendix

The Mean Dislocation Density Model (MD^{2}M)

The MD^{2}M uses the theory discussed above to simulate the microstructure evolution during metal working as well as during heat treatments. Its state variables are the (mean) dislocation density ρ, the subgrain size δ^{sub}, the grain size δ^G and the recrystallized fraction X^{rex}. The model consists of two parts, namely the work hardening model described in subsection 2.1.2 for simulating metal working, and the recovery and recrystallization models described in subsections 2.1.3 and 2.1.4 for simulating heat treatments.

On the following pages, the governing equations are detailed together with a reference to the relevant theory. They are written sequentially, so the models can be read like instructions for a computer, with earlier lines updating variables that are necessary later on. Whenever the increase of some variable x is denoted by $dx = ...$, the next step is always $x = x + dx$, and it is ommited for convenience. This = sign is to be interpreted in a programming sense, where $x = x + 2$ means that the variable x is increased by 2.

The MD^{2}M was developed at LKR based on the various sources mentioned above. It is worth to note here that the used microstructure model MD^{2}M has certain similarities with the microstructure model implemented directly in *MatCalc*. The basics for both models are common and were previously developed in collaboration between the LKR and TU Wien (see e.g. [20] and [6]). Further development took place in these groups independently (see e.g. [13]).

WORK HARDENING

INITIALIZATION

Constants

σ_0, α_1, α_2, G, χ, b, ν_D, Q_D, f_T, A, B, C, Q_{vac}, ρ_{eq}, K	taken from literature / trial simulations

State Variables

$\rho = \rho_0$	initial values of dislocation density (usually ρ_{eq})
$\delta^G = \delta_0^G$	and grain size

CALCULATION IN EACH STEP

Input Variables

T, $d\epsilon_p$, dt	inputs from FEM model, see subsection 2.3.2

Diffusion Coefficient

$D = \chi \cdot \nu_D b^2 \exp\left(\frac{-Q_D}{k_B T}\right)$	equation 2.9

Annihilation Distance

$d_{ann} = Gb^4 \frac{1}{2\pi(1-\nu)Q_{vac}}$	equation 2.5

Flow Stress Model for Dislocation Density

$d\rho =$ $= \frac{f_T\sqrt{\rho}}{Ab} d\varepsilon_p - B\frac{2d_{ann}f_T}{b}\rho d\varepsilon_p - 2C \cdot D\frac{Gb^2}{k_B T}(\rho^2 - \rho_{eq}^2)dt$	equation 2.10
$\delta^{sub} = \frac{K}{\sqrt{\rho}}$	equation 2.11

Hardening

$\sigma_F = \sigma_0 + \alpha_1 Gb\sqrt{\rho} + \alpha_2 Gb(\frac{1}{\delta^{sub}} + \frac{1}{\delta^G})$	equation 2.2 and subsection 2.3.2
$H = \frac{Gf_T}{2}\left(\alpha_1 + \frac{\alpha_2}{K}\right) \cdot \left(\frac{1}{A} - 2Bd_{ann}\sqrt{\rho}\right)$	taking $\frac{d\sigma_F}{d\epsilon_p}$ and using equations 2.2 and 2.10

RECOVERY AND RECRYSTALLIZATION

INITIALIZATION

Constants

C_ρ, ρ_{eq}, G, b, γ_{sub}, γ_{GB} ν_D, Q_D, C_M, δ_{eq}^{sub}, K, C^{rex}, Q^{rex}	taken from literature / trial simulations

State Variables

$\delta^{sub} = \delta^{sub}$	initial values of dislocation density,
$\delta^{G} = \delta_0^{G}$	grain size,
$\delta^{rex} = \delta_{crit}^{rex}$	recrystallized grain size
$N^{rex} = 0$	and recrystallization nuclei

CALCULATION IN EACH STEP

Input Variables

T, dt	inputs from FEM model, see subsection 2.3.2
χ, N_{pr}, r_{pr}	precipitation variables, either static or from external program (e.g. MatCalc, see sections 2.2 and 3.2)

Diffusion Coefficient

$D = \chi \cdot \nu_D b^2 \exp\left(\frac{-Q_D}{k_B T}\right)$	equation 2.9

Grain and Subgrain Boundary Mobility

$M_{GB} = C_M \frac{Db}{k_B T}$	equation 2.13
$M_{sub} = M_{GB} \frac{\gamma_{sub}}{\gamma_{GB}}$	equation 2.14

Zener Drag

$P_{Z_{GB}} = 2N_{pr}\pi\gamma_{GB}r_{pr}^2$	
$P_{Z_{sub}} = 2N_{pr}\pi\gamma_{sub}r_{pr}^2$	equation 2.15

Grain Growth

$d\delta^G = M_{GB}\left(\frac{6\gamma_{GB}}{\delta^G} - 2P_{Z_{GB}}\right)dt$	
$d\delta^G = 0 \quad \text{if} \quad \left(\frac{6\gamma_{GB}}{\delta^G} - 2P_{Z_{GB}}\right) < 0$	equation 2.17

Recovery

$d\delta^{sub} = M_{sub}\left(\frac{6\gamma_{sub}}{\delta^{sub}} - \frac{6\gamma_{sub}}{\delta^{sub}_{eq}} - 2P_{Z_{sub}}\right)dt$ $d\delta^{sub} = 0 \quad \text{if} \quad \left(\frac{6\gamma_{sub}}{\delta^{sub}} - \frac{6\gamma_{sub}}{\delta^{sub}_{eq}} - 2P_{Z_{sub}}\right) < 0$	equation 2.20
$\rho = \left(\frac{K}{\delta^{sub}}\right)^2$	equation 2.11

Recrystallization

$P^{rex} = C_\rho(\rho - \rho_{eq})Gb^2 + \frac{3\gamma_{sub}}{\delta^{sub}}$	equation 2.28
$\delta^{rex}_{crit} = \frac{4\gamma_{GB}}{P^{rex} - P_{Z_{GB}}}$	equation 2.27
$dN^{rex} = C^{rex}\exp\left(-\frac{Q^{rex}}{k_B T}\right)(1 - X^{rex})dt$ $dN^{rex} = 0 \quad \text{if} \quad \delta^{sub} < \delta^{rex}_{crit}$	equation 2.32
$d\delta^{rex} = \begin{cases} M_{GB}\left(P^{rex} - P_{Z_{GB}}\right)(1 - X^{rex})dt \\ 0 \quad \text{if} \quad \left(P^{rex} - P_{Z_{GB}}\right) < 0 \end{cases}$ $d\delta^{rex} = 0 \quad \text{if} \quad \delta^{sub} < \delta^{rex}_{crit}$	equation 2.31
$X^{rex} = N^{rex}\frac{\pi}{6}(\delta^{rex})^3$	equation 2.30

Bibliography

[1] URL: https://www.ait.ac.at/laboratories/abschreck-und-umformdilatometer-dil805adt.

[2] URL: https://www.matcalc-engineering.com/start/.

[3] M. F. Ashby and D. R. H. Jones. *Werkstoffe 1: Eigenschaften, Mechanismen und Anwendungen*. Spektrum-Akademischer Vlg, Aug. 10, 2006. ISBN: 3827417082. URL: https://www.ebook.de/de/product/5516669/michael_f_ashby_david_r_h_jones_werkstoffe_1_eigenschaften_mechanismen_und_anwendungen.html.

[4] Bathe. *Finite-Elemente-Methoden Matrizen und lineare Algebra; Die Methode der finiten Elemente; Lösung von Gleichgewichtsbedingungen und Bewegungsgleichungen*. BerlinHeidelbergNew YorkTokyo: Springer, 1986. ISBN: 354015602X.

[5] D. J. Benson. *The History of LS-DYNA*. June 2007. URL: https://blog.d3view.com/history-of-ls-dyna-by-dr-david-benson/.

[6] H. Buken, P. Sherstnev, and E. Kozeschnik. "A state parameter-based model for static recrystallization interacting with precipitation". In: *Modelling and Simulation in Materials Science and Engineering* 24.3 (Feb. 2016), p. 035006. ISSN: 0965-0393. DOI: 10.1088/0965-0393/24/3/035006.

[7] "DOE fundamentals handbook: Material science. Volume 1". In: (Jan. 1993). DOI: 10.2172/10154822. URL: https://www.osti.gov/biblio/10154822.

[8] M. Feischl and M. Melenk. *Numerik für partielle Differentialgleichungen: instationäre Probleme*. June 2020.

[9] M. Feischl and D. Praetorius. *Numerik für partielle Differentialgleichungen: stationäre Probleme*. Jan. 2020.

[10] M. Fleck. URL: https://commons.wikimedia.org/w/index.php?curid=96858277.

[11] G. Gottstein and D. G. für Metallkunde. *Rekristallisation metallischer Werkstoffe: Grundlagen, Analyse, Anwendung*. Oberursel, West Germany: Deutsche Gesellschaft für Metallkunde, 1984. ISBN: 9783883550626. URL: https://books.google.at/books?id=2UGAAAAACAAJ.

[12] F. J. Humphreys. *Recrystallization and related annealing phenomena*. Amsterdam Boston: Elsevier, 2004. ISBN: 0080441645.

[13] E. Kabliman, A. H. Kolody, J. Kronsteiner, M. Kommenda, and G. Kronberger. "Application of symbolic regression for constitutive modeling of plastic deformation". In: 6 (June 2021), p. 100052. DOI: 10.1016/j.apples.2021.100052.

[14] E. Kozeschnik. *Modeling Solid-State Precipitation*. Momentum Press, Nov. 30, 2012. 502 pp. ISBN: 1606500627. URL: https://www.ebook.de/de/product/19222657/ernst_kozeschnik_modeling_solid_state_precipitation.html.

[15] G. Kromoser. *CHARAKTERISIERUNG DER GEFÜGEENTWICKLUNG UND UMFORMBARKEIT BEI DER HALBWARMUMFORMUNG VON STÄHLEN*. 2017.

[16] *LS-DYNA KEYWORD USER'S MANUAL*. II. LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY, 2020. URL: https://www.dynasupport.com/manuals/ls-dyna-manuals.

[17] K. Marthinsen, J. Friis, B. Holmedal, I. Skauvik, and T. Furu. "Modelling the Recrystallization Behaviour during Industrial Processing of Aluminium Alloys". In: *Materials Science Forum* 715-716 (Apr. 2012), pp. 543–548. DOI: 10.4028/www.scientific.net/msf.715-716.543.

[18] K. Marthinsen and E. Nes. "The ALFLOW Model - A Microstructural Approach to Constitutive PlasticityModelling of Aluminium Alloys". In: *Materials Science Forum - MATER SCI FORUM* 331-337 (May 2000), pp. 1231–1242. DOI: 10.4028/www.scientific.net/MSF.331-337.1231.

[19] E. Nes. "Modelling of work hardening and stress saturation in FCC metals". In: *Progress in Materials Science* 41.3 (July 1997), pp. 129–193. DOI: https://doi.org/10.1016/S0079-6425(97)00032-7.

[20] P. Sherstnev, C. Melzer, and C. Sommitsch. "Prediction of precipitation kinetics during homogenisation and microstructure evolution during and after hot rolling of AA5083". In: *International Journal of Mechanical Sciences* 54.1 (Jan. 2012), pp. 12–19. DOI: 10.1016/j.ijmecsci.2011.09.001.

[21] E. A. de Souza Neto, D. Perić, and D. R. J. Owen. *Computational methods for plasticity : theory and applications*. Chichester, West Sussex, UK: Wiley, 2008. ISBN: 978-0-470-69452-7.

[22] B. Varghese, A. Unnikrishnan, and K. Jacob. “Spatial Clustering Algorithms- An Overview”. In: *Asian Journal of Computer Science And Information Technology* 3 (Jan. 2014).

[23] H. Vatne, T. Furu, R. Ørsund, and E. Nes. “Modelling recrystallization after hot deformation of aluminium”. In: *Acta Materialia* 44.11 (Nov. 1996), pp. 4463–4473. DOI: https://doi.org/10.1016/1359-6454(96)00078-X.