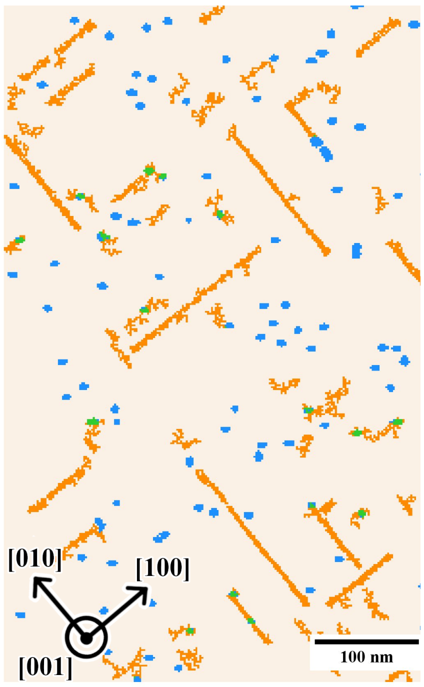
Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

Kristin Frøystein

Scanning precession electron diffraction data analysis of in-situ precipitate evolution in an Al-Mg-Si-Cu alloy

Master's thesis in Applied Physics and Mathematics
Supervisor: Randi Holmestad
Co-supervisor: Elisabeth Thronsen, SINTEF and Jørgen A. Sørhaug
January 2023



NTNU
Norwegian University of
Science and Technology

Kristin Frøystein

Scanning precession electron diffraction data analysis of in-situ precipitate evolution in an Al-Mg-Si-Cu alloy

Master's thesis in Applied Physics and Mathematics
Supervisor: Randi Holmestad
Co-supervisor: Elisabeth Thronsen, SINTEF and Jørgen A. Sørhaug
January 2023

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

Abstract

The material studied in this thesis is a Al-Mg-Si-Cu (6xxx) aluminium alloy. The precipitates that form in this alloy during ageing determines many of the material's mechanical properties, including strength. Understanding how different temperatures influence the composition and crystal structure of precipitates is therefore important to understand how heat treatment affects the materials physical properties.

A method of investigating the crystal structure of materials is scanning precession electron diffraction. Scanning precession electron diffraction is a transmission electron microscopy technique that yields datasets of high quality. One dataset can contain hundreds of thousands of diffraction patterns, and each can be be few gigabytes in size. These large quantities of data can be tedious to analyse manually. Therefore, machine learning is a natural approach to extract information from these datasets.

In this master's thesis, scanning precession electron diffraction datasets from an Al-Mg-Si-Cu (6xxx) aluminium alloy sample have been studied. The datasets were recorded as the sample was *in-situ* heated, with a constant heating rate of 0.01 K/s.

The data analysis used in the thesis are principal components analysis using singular value decomposition and non-negative matrix factorisation. The non-negative matrix factorisation revealed that there are 4 phases present in the datasets. These are β'', L, C and Q$'$. From the non-negative matrix factorisation, phase maps were created using a combination of masking, the fitting of Gaussian functions and threshold values.

The main goals of the master thesis are to examine how the precipitates evolve during the heat treatment, and find which phases exist at what temperatures. An investigating into how the phase transformations are happening is also carried out. Do the transformations occur through dissolution and nucleation or through continuous phase transformations? Possible phase transitions were observed, and these are β'' to L, β'' to C, and L to Q$'$.

Sammendrag

Materialet studert in denne masteroppgaven er en Al-Mg-Si-Cu (6xxx) aluminiumlegering. Precipitatene som dannes i denne legeringen under utherding bestemmer mange av materialets mekansike egenskaper, inkludert styrke. Å forstå hvordan forskjellige temperaturer påvirker sammensetningen av og krystallstrukturen til presipitatene er derfor viktig, for å forstå hvordan varmebehandling av materialet påvirker dets fysiske egenskaper.

En metode for å undersøke krystallstrukturen til materialer er ved skannende presesjonselektrondiffraksjon. Skannende presesjonselektrondiffraksjon er en transmisjonselektronmikroskopiteknikk som gir dataset av høy kvalitet. Et dataset can inneholde hundretusener av diffraksjonsmøntre, og hvert dataset kan være på størreslse med noen gigabyte. Disse store mengdene data kan være krevende å analysere manuelt. Derfor er maskinlæring en naturlig tilnærming for å utvinne informasjon fra disse datasettene.

I denne masteroppgaven er skannende presesjonselektrondiffraksjon-datasett fra en Al-Mg-Si-Cu (6xxx) aluminiumlegerering studert. Datasettene ble tatt opp samtidig som prøven ble *in-situ* oppvarmet med en konstant oppvarmingsrate på 0.01 K/s.

Dataanalysemetodene brukt i masteroppgaven er hovedkomponentanalyse og ikkenegativ matrisefaktorisering. Den ikke-negative matrisefaktoriseringen avlørte at det er 4 faser tilstede i datasettene. Disse er β'', L, C og Q$'$. Fra den ikke-negative matrisefaktoriseringen har det blitt laget fasekart ved å bruke en kombinasjon av maskering, kurvetilpassning av Gaussiske funksjoner og grenseverdier.

Målene med masteroppgaven var å undersøke hvordan presipitatene utvikler seg ved varmebehandling, og å finne ut hvilke faser som eksisterer ved hvilke temperaturer. En udersøkelse av fasetransformasjonene som skjer er også gjort, for å finne ut om transformasjonene skjer ved oppløsning og nukleering, eller ved kontinuerlige transformasjoner. Mulige fasetransformasjoner som ble observert var β'' til L, β'' til C, og L til Q$'$.

Preface

This master's thesis is written in my last semester at NTNU in Trondheim, as part of my degree in applied physics and mathematics. The thesis was written under the Department of Physics section Material Physics. My supervisor was professor Randi Holmestad, along with co-supervisors Elisabeth Thronsen (SINTEF) and PhD canditate Jørgen A. Sørhaug. This work is loosely built on my own project thesis written last semester.

I would like to thank my supervisors Randi Holmestad, Elisabeth Thronsen and Jørgen A. Sørhaug for their guidance, valuable suggestions and help. I have always left our weekly meetings with new ideas to explore. I would also like to thank Tina Bergh and Jørgen A. Sørhaug for providing the data used in this thesis. Tina has also made helpful suggestions and provided tips for the development of the data processing methods used. Additionally, I would like to thank Emil Frang Christiansen for introducing me to and teaching me how to use Pyxem and HyperSpy when I first started my project thesis in January 2022. The creation of phase maps in this thesis is also based on Emil's code.

Kristin Frøystein

Trondheim, 20th of January, 2023

List of Abbreviations

CBED	Convergent beam electron diffraction
GP Zone	Guinier-Preston zone
NMF	Non-negative matrix factorisation
PCA	Principal component analysis
SPED	Scanning precession electron diffraction
SSSS	Supersaturated solid solution
SVD	Singular value decomposition
(V)DF	(Virtual) dark-field
(V)BF	(Virtual) bright-field
ZA	Zone axis

Table of Contents

Chapter 1

Introduction

Aluminium is the most abundant metal on earth, and its crust contains 8.1 % aluminium. It is also a very widely used material, commonly used in everyday life in aluminium foil, cans, and frying pans. In transportation, aluminium is used in for example cars and airplanes. Electrical transmission lines can also be made of aluminium, as aluminium is a great electrical conductor [1]. Another advantage of aluminium use is its great ability to be recycled, using only 5 % of the energy required to produce new aluminium through electrolysis [2]. Additionally, aluminium is resistant to corrosion [1]. These qualities makes aluminium an attractive material, and justifies the increasing demand for aluminium in the world today [3].

Pure aluminium is soft and malleable and is thus unsuitable for applications where strength is a major design consideration [1]. This yields pure aluminium unsuitable for applications that require strength. Creating aluminium alloys, through adding silicon (Si), magnesium (Mg), manganese (Mn), copper (Cu), and other elements, introduces several new properties to aluminium, including substantial increase in strength [4]. These new properties arise as a consequence of a changed structure in the material at an atomic level. Various new phases called *precipitates* can appear in the material depending on temperature, time and composition. Precipitation is one way of introducing strength to an aluminium alloy. In order to understand how an alloy's properties can change under different conditions it is important to study which precipitates that appear and disappear during for example heating of the material.

A powerful tool for examining the structure of materials is transmission electron microscopy (TEM). In TEM, an electron beam is directed onto the sample and the transmitted signal is detected. Due to the strong interaction between the electrons and the sample, a range of signals can be detected simultaneously. One operation mode is diffraction mode, in which the diffraction pattern is captured. Using scanning precession electron diffraction (SPED), an enourmous amount of diffraction patterns can be recorded, easily a couple of GB. This makes the identification of all the patterns an insurmountable task to do manually. Therefore, a method to identify such large amounts of patterns is required.

Recently, SPED has been used in the phase mapping of precipitates in Al alloys [5, 6]. Additionally, *in-situ* heating experiments have been done on an Al-Mg-Si-Cu alloy. Non-negative matrix factorisation has also previously been used on SPED data [5, 7]. In the in-situ study by Sunde et al. [6], the material was cooled to room temperature between each scan. In this master's thesis, the scans are done over the sample as it is continuously heated. The heating rate of 0.01 K/s were selected in order to be able to compare results with differential scanning calorimetry (DSC).

In this master's thesis, 11 SPED datasets recorded from a 6xxx Al alloy during *in-situ* heating is studied using non-negative matrix factorisation (NMF). A method using the fitting of Guassians is developed to create phase maps from each dataset. The goal is to determine which precipitates exist in each dataset, and to examine how the phases evolve during the *in-situ* heating. It is also desirable to examine how phase transformations happen; do they dissolve and re-nucleate into a new phase, or do they undergo a continuous phase transformation?

In this thesis, theory explaining fundamental crystallography and diffraction is presented first, followed by introduction to SPED. Precipitation in 6xxx alloys is presented next. Then, the method is presented. This chapter contains information about the material used, the data acquisition using SPED and a description of the data processing methods used and developed. The results are presented next, consisting of phase maps over each sample, statistics from the precipitates, and examples of phase transformations. The results are followed by a discussion of both the method and the material, and a conclusion ends the report, along with further work.

Chapter 2

Theory

This section provides the necessary theory to explain the methods used in this project, as well as a fundament to understand the results presented. First, basic crystallography is presented, followed by diffraction. Then, transmission electron microscopy and scanning precession electron diffraction is introduced. Lastly, the 6xxx aluminium system is presented..

2.1 Crystallography

This chapter is in its entirety influenced by [8, 9].

When thinking of crystals, perhaps pictures of beautiful, colorful gems appear. However, crystals are much more common than that. Fundamentally, a crystal is a regular, periodic arrangement of repeated units; atoms, molecules or ions. The structure can be seen as a periodic lattice consiting of lattice points. The crystal structures of materials heavily influence the macroscopic properties of the material. This section presents basic crystallography, and is in its entirety influenced by references [8, 9].

A perfect crystal is a Bravias lattice. In this type of lattice, no matter which lattice point is chosen, the lattice looks the same. This can be described by the following equation:

$$\vec{R} = n_1\vec{a_1} + n_2\vec{a_2} + n_3\vec{a_3}, \tag{2.1}$$

where $\vec{a_i}$ are three non-parallel vectors existing in separate planes, and n_i are integers. The vectors $\vec{a_i}$ are called the *primitive vectors*, while the vector $\vec{R}$ is the *lattice vector.*

In three dimensions, one can define a volume of the lattice that when translated over the entire lattice contains all the volume of the lattice, without overlap. This is the *primitive unit cell.* The volume V of this cell is given by

$$V = \vec{a_3} \cdot (\vec{a_1} \times \vec{a_2}). \tag{2.2}$$

In order to define the various crystal systems, we need the lengths of the lattice vectors and angles between them. These are defined as follows:

$$a = |\vec{a_1}|, \quad b = |\vec{a_2}|, \quad c = |\vec{a_3}| \tag{2.3}$$

and the angles α, β, γ are the angles between the lattice vectors.

Within a crystal system, one can define planes and axes. A *lattice plane* is a plane which contain at least three lattice points. Each lattice plane can be identified by where the plane intersects with the axes. The Miller indices of such a plane h, k, l are given as the inverse of the points were the plane intersects the axes. A *zone axis* (ZA) is a lattice row parallel to the intersection of two or more families of lattice planes $\{hkl\}$. A ZA, denoted $[uvw]$ is parallel to a family of lattice planes if Weiss' law is fulfilled:

$$hu + kv + lw = 0. \tag{2.4}$$

In order to model crystals using Bravais lattices, a *basis* needs to be introduced. A real crystal is a Bravais lattice with atoms, ions, or molecules at the lattice points. These physical units are the basis of the crystal. Thus, a crystal is defined as the Bravais lattice and the basis. In all, there are 7 crystal systems and 14 Bravais lattices. These are shown in Table 2.1.

The 14 Bravais lattices can be additionally be categorised by the arrangement of the lattice points in the unit cell. There are 4 of these types of centering. Primitive (P) contains lattice points only at each corner of the unit cell. In body-centered (I), there is one lattice point in the center of the unit cell in addition to one at each corner. The third is face-centered. Here, there is one lattice point at each corner, and one lattice point in the middle of each face. Lastly, there is base-centered (A, B, C). At each corner there is a lattice point. In addition, there is one lattice point at the center of two parallel faces. Depending on which faces these are, they are denoted A, B or C. A corresponds to the face $(\vec{a_2}, \vec{a_3})$, B to $(\vec{a_1}, \vec{a_3})$ and C to $(\vec{a_1}, \vec{a_2})$. The different centerings of the Bravais lattices are also shown in table 2.1.

Table 2.1: The different crystal systems, their definitions, and the possible Bravais Lattices.

Crystal system	**Definition**	**Bravais Lattices**
Triclinic	$a \neq b \neq c$ $\alpha \neq \beta \neq \gamma$	P
Monoclinic	$a \neq b \neq c$ $\alpha = \beta = 90°$	P, I
Orthorombic	$a \neq b \neq c$ $\alpha = \beta = \gamma = 90°$	P, C, I, F
Tetragonal	$a = b$ $\alpha = \beta = \gamma = 90°$	P, I
Cubic	$a = b = c$ $\alpha = \beta = \gamma = 90°$	P, I, F
Trigonal	$a = b = c$ $\alpha = \beta = \gamma$	P
Hexagonal	$a = b$ $\alpha = \beta = 90°$ $\gamma = 120°$	P

2.2 Diffraction

In the previous section, the basics of crystals were introduced. When waves interact with the crystal structure, they get scattered. This section describes the basics of these interaction, the resulting diffraction. The following section is based on *kinematic diffraction*, where each wave is only scattered once. The scattering is also assumed to be elastic and coherent. Dynamic diffraction is briefly introduced at the end of this section.

Figure 2.1 shows a schematic of how waves with wavelength λ are scattered from parallel crystal planes with a distance d between them. The wavelength, λ, is comparable to the distance d.

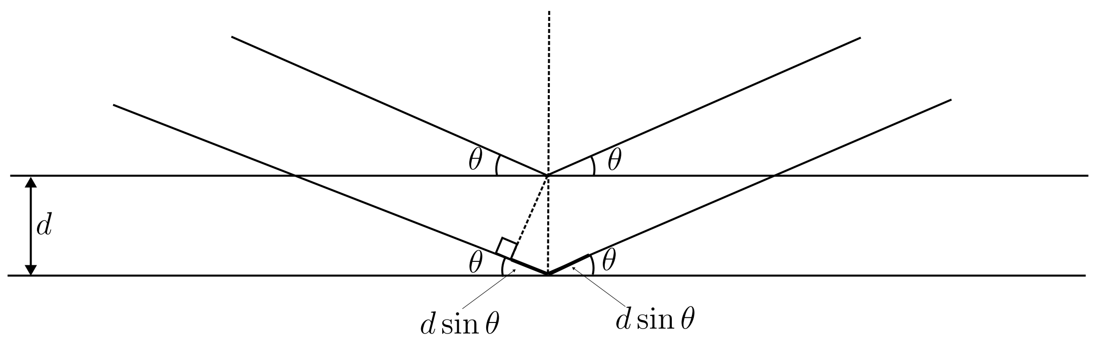


Figure 2.1: Waves scattered from two crystal planes with distance d. The incoming wave has a wavelength λ comparable to the distance d.

As the waves are reflected, the angles between the incoming and the reflected wave and the crystal planes are the same.

$$\theta = \theta_{\text{in}} = \theta_{\text{out}} \tag{2.5}$$

The difference in path length between the two waves is then [9]

$$2d\sin\theta = n\lambda, \tag{2.6}$$

where θ is the angle between the crystal planes and the incoming wave. Here, $2d\sin\theta$ is the path length between the waves scattered by the first and the second crystal plane. This is Bragg's law. The distance d_{hkl} between the crystal planes in a crystal with given Miller indices $\{hkl\}$ is given by [8]

$$d_{hkl} = \frac{2\pi}{|h\vec{b_1} + k\vec{b_2} + l\vec{b_3}|}, \tag{2.7}$$

where $\vec{b_1}$, $\vec{b_2}$ and $\vec{b_3}$ are the reciprocal lattice vectors in reciprocal space. These vectors are defined as follows,

$$\vec{b_1} = \frac{2\pi}{V}(\vec{a_2} \times \vec{a_3}), \quad \vec{b_2} = \frac{2\pi}{V}(\vec{a_3} \times \vec{a_1}), \quad \vec{b_3} = \frac{2\pi}{V}(\vec{a_1} \times \vec{a_2}), \tag{2.8}$$

where V is the volume of the unit cell defined above. Combined, these vectors constitute the reciprocal lattice vector, given by

$$\vec{g} = h\vec{b_1} + k\vec{b_2} + l\vec{b_3}. \tag{2.9}$$

In an aluminium crystal, which is cubic, each $\vec{b_i}$ are orthogonal with length b = $2\pi/a$, which reduces equation 2.7 to

$$d_{hkl} = \sqrt{\frac{a^2}{h^2 + k^2 + l^2}}, \tag{2.10}$$

where a is the lattice parameter of the crystal [8]. For aluminium, $a = 4.05\text{Å}$ [9].

Structure Factor

The reciprocal lattice points determine the positions of the reflections in the diffraction patter. How about the intensities? The intensity of a diffraction pattern from a crystal is proportional to crystal structure factor [8]:

$$I \propto |F(\vec{g})|^2 \tag{2.11}$$

Here, $F(\vec{g})$ is the structure factor. The structure factor is given by [8]

$$F(\vec{g}) = \sum_{j=1}^{N} f_j(\vec{g}) e^{i\vec{g}\cdot\vec{r}_j}. \tag{2.12}$$

Here, f_j is the atomic form factor for atom j, while $\vec{r}_j$ is the position atom j in the unit cell. The structure factor describes the intensities from each atom in the basis. For each combination of hkl, the intensities can now be found by

$$F_{hkl} = \sum_{j=1}^{N} f_j e^{-2\pi i(hx_j + ky_j + lz_j)}. \tag{2.13}$$

2.2.1 Two-beam approximation

In the two-beam approximation, there are assumed to be only two beams in the system; the central beam and one beam diffracted in the crystal [10].

The Laue condition, which is the reciprocal space equivalent of Braggs' law, is

$$\vec{k}' - \vec{k} = \vec{g} \tag{2.14}$$

The Ewald sphere is constructed from the incoming beam, denoted by $\vec{k}$. Let this vector end at a lattice point in the reciprocal lattice. Then, a sphere with radius $|k| = 2\pi/\lambda$ is drawn with the same origin as $\vec{k}$. This sphere is the Ewald sphere. A diffracted beam will be formed if the Ewald sphere intersects any other point in the reciprocal lattice as well [9]. Figure 2.2 shows the geometrical explanation of the Laue condition, as well as the Ewald sphere.

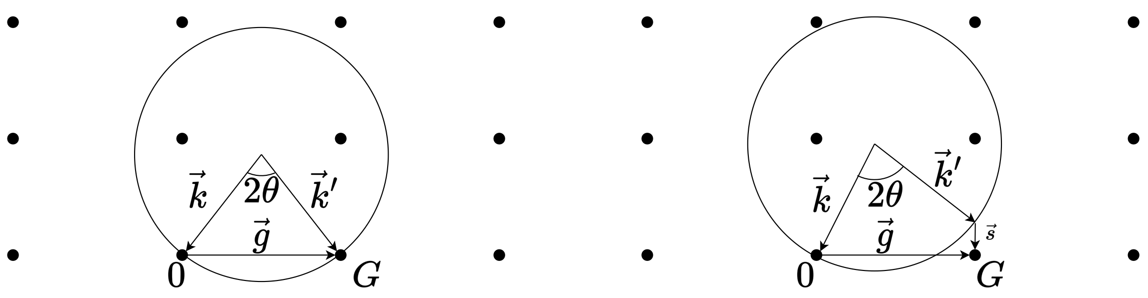


Figure 2.2: Laue diffraction in two cases; Bragg diffraction ($\vec{s} = 0$) on the left, and the general with $\vec{s} \neq 0$ on the right. Based on [10].

However, one can also define a system with a small deviation from Bragg's law, as shown in Figure 2.2.

$$\vec{k'} - \vec{k} = \vec{g} + \vec{s}, \tag{2.15}$$

where $\vec{k'}$ is the diffracted wave, $\vec{k}$ is the incoming wave, $\vec{g}$ is the reciprocal lattice vector and $\vec{s}$ is the deviation between the Ewald sphere and the diffraction spot. The deviation vector can be defined as

$$\vec{s} = s_a\hat{a}^* + s_b\hat{b}^* + s_c\hat{c}^*, \tag{2.16}$$

where $\hat{a}^*$, $\hat{b}^*$ and $\hat{c}^*$ are unit cell translation vectors.

Reciprocal lattice rods are reciprocal lattice points that are stretched in one direction. This is a result of the inverse relationship between real and reciprocal space. A thinner sample in real space results in taller rel-rods in reciprocal space [11]. Rel-rods can be seen in Figure 2.3. Diffracted beams occur when the rel-rods intersect the Ewald sphere. This means that diffraction spots that may not be allowed according to Bragg's law in Equation 2.6, can show up in the diffraction pattern anyway.

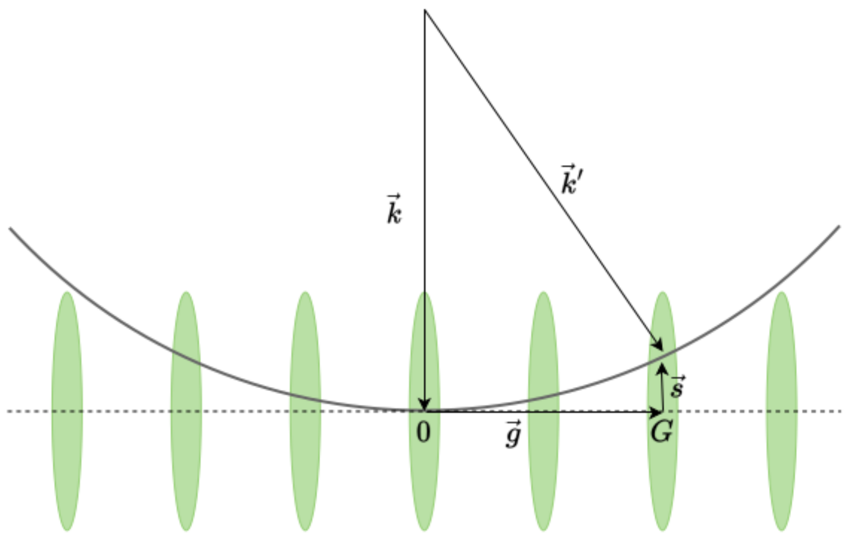


Figure 2.3: A schematic drawing of the Ewald sphere intersecting the stretched out diffraction spots; the reciprocal lattice rods.

From [10] we get the intensity from the gth diffracted beam, from a sample of thickness t:

$$I_g(t) = \left(\frac{\pi}{\xi_g}\right)^2 \frac{\sin^2(\pi t s)}{(\pi t s)^2}, \tag{2.17}$$

where the extinction distance

$$\xi_g = \frac{\pi k V_{cell} \cos\theta}{F(2\theta)}. \tag{2.18}$$

Here s_c is replaced with s, which is the component of $\vec{s}$ that is normal to the crystal plane, and F is the structure factor.

2.2.2 A Brief Note on Dynamic diffraction

So far kinematic diffraction theory has been discussed. In kinematic diffraction it is assumed that only one scattering event happens, and this is the case with a very thin sample [11]. However, samples are often thick enough to allow for more than one scattering event. This is dymanic diffraction. In this case, a beam that has already been diffracted serve as the incoming beam for the second diffraction [11]. This affects the intensity of the diffraction spots. Some diffraction spots that are kinematically forbidden may also appear because of the multiple scattering events [11].

In dynamical diffraction two-beam, s is replaced with s', which is $s' = (s^2 + \xi_q^{-2})^{1/2}$. This is called the effective excitation error [10].

2.3 Scanning Precession Electron Diffraction

In transmission electron microscopy (TEM), an electron beam, accelerated by a specific voltage, are focused with magnetic lenses [11]. Then, the electron beam is directed through various lenses and apertures, and sent through the sample. Here, most of the electron beam will pass through the sample without any interaction. However, a small amount of the electrons will interact, and be scattered. In diffraction mode in a TEM,both the central beam and the scattered beams are sent either to a fluorecent screen or a detector, revealing a diffraction pattern [11]. Diffraction can be done with either a parallel or convergent beam, and is only one of several modes in TEM. Other modes are imaging, electron energy loss spectroscopy (EELS) and convergent-beam electron diffraction (CBED) [11].

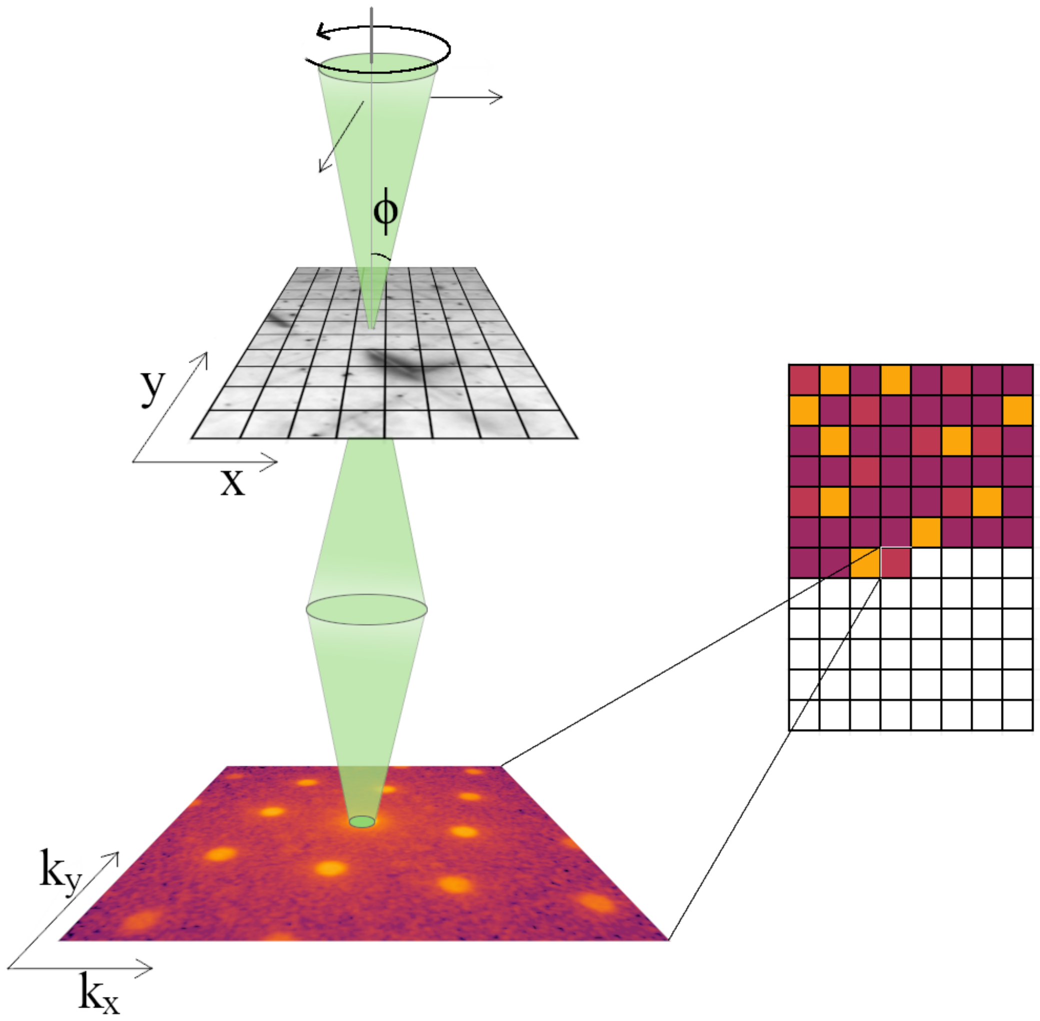


Figure 2.4: A schematic of how the SPED 4D dataset is recorded. The beam precesses around in the cone shape indicated. ϕ is the precession angle, which is greatly exaggerated.

In scanning electron diffraction (SED), the incoming probe is scanned over a region of interest. A diffraction pattern is recorded at each probe position. The beam is typically converged to a small probe. This can be combined with precession, where the beam precesses in a cone shape with angle ϕ. This is done in scanning precession electron precession (SPED). This can be seen in Figure 2.4. At each probe position, several patterns are recorded while the beam precesses, and combined. Thus, each two-dimensional diffraction pattern in reciprocal space is associated with its real space position in the region of interest in the sample. In this method, the Ewald sphere is tilted around with the electron beam, which increases the chances of the Ewald sphere intersecting a reciprocal lattice point. Therefore, SPED patterns can include more reflections than a TEM diffraction pattern [12]. In SPED, the intensities appear more kinematic-like than when using an unprecessing probe to record the patterns [12].

2.4 6xxx Aluminum Alloy

There are two main groups of Al alloys; wrought alloys and cast alloys. Cast alloys are produced directly in its final shape through moulding. Wrought alloys are deformed into its intended shape by extrusion, rolling or forging. The wrought alloys accounts for the largest part of aluminium products manufactured. The 6xxx Al alloy is a wrought alloy which has great extrudability, and is often used in the automobile industry because of its low density, high strength and corrosion resistance [13].

Aluminium in itself is not particularly strong. However, adding small amounts of other elements can greatly increase it. Deformation of crystals can be attributed to the movement of dislocations within the material. Mechanically blocking these dislocations will therefore increase the material's strength. One important strengthening mechanism is precipitate hardening. This includes the formation of semi-coherent precipitates in the material[9] during thermomechanical treatment. These precipitates act as obstacles for the gliding dislocations.

The first step in the heat treatment is solution heat treatment (SHT), which is typically followed by quenching to room temperature. Now, the 6xxx alloy, the material is a supersaturated solid solution (SSSS), which is nearly homogeneous. Then, the material is aged, either naturally at room temperature or artificially at an elevated temperature. During ageing, the dissolved alloy elements in the SSSS start to cluster, and eventually grow to become precipitates. The precipitates formed will depend on the composition of the alloy. Before the precipitates form, Guinier-Preston (GP) zones form. As the ageing progresses, the GP zones are replaced by other phases in the sequence [14]. The precipitation sequence in a Al-Mg-Si-Cu alloy is given as [15]:

$$\text{SSSS} \rightarrow \text{Atomic clusters} \rightarrow \text{GP Zones} \rightarrow \beta'', \text{L} \rightarrow \beta'_{\text{Cu}}, \text{L, S, C, Q}' \rightarrow \text{Q} \tag{2.19}$$

Which precipitates that form is highly dependent on the composition and thermomechanical treatment of the alloy. Table 3.1 shows the space groups, compositions and lattice parameters of the possible precipitates appearing in an Al-Mg-Si-Cu alloy.

Table 2.2: The precipitates that can appear in an Al-Mg-Si-Cu alloy, their space groups, compositions and lattice parameters.

Phase	Space Group	Composition	Lattice Parameters [Å]	Reference
β'	$P6_3/m$	$Mg_6Si_{3,33}$	$a = 7.15$ $b = 7.15$ $c = 6.74$, $\gamma = 120°$	[16]
β'_{Cu}	$P\bar{6}2m$	$Al_3Mg_3Si_2Cu$	$a = 6.90$ $b = 6.90$, $\beta = 60°$ $c = 4.05$	[17]
β''	C2/m	$Al_2Mg_5Si_4$	$a = 15.6$ $b = 4.05$, $\beta = 105.3°$ $c = 6.74$	[18]
L	-	Varies	$a = 10.39$ $b = 4.05$, $\beta = 105.3°$ $c = 8.10$	[17]
C	$P2_1$	$AlMg_4Si_3Cu$	$a = 10.32$ $b = 4.05$, $\beta = 100.9°$ $c = 8.10$	[15]
Q′	P6	$Al_xMg_{12-x}Si_7Cu_2$	$a = 10.39$ $b = 10.39$ $c = 4.05$, $\gamma = 120°$	[19, 20]
Q	P6	$Al_xMg_{12-x}Si_7Cu_2$	$a = 10.39$ $b = 10.39$ $c = 4.05$, $\gamma = 120°$	[19, 20]
U2	Pnma	AlMgSi	$a = 6.75$ $b = 4.05$ $c = 7.94$	[21]
S	Varies	Varies	Varies	[15]

Chapter 3

Methods

This section describes the material investigated, the data acquisition method, and the data analysis used. The code used in the data analysis is written in Python, using HyperSpy and Pyxem libraries [22, 23]. The code is written in Jupyter Notebook and as Python scripts. The code is listed in Appendix A.

3.1 Material and Heat Treatment

The SPED datasets were recorded from an aluminium 6xxx alloy sample by Tina Bergh and Jørgen A. Sørhaug. Its composition can be seen in table 3.1. The alloy was manufactured by Neuman in Austria.

Before the *in-situ* experiment, the sample was heat treated in 2018, first at 540°C for 12 minutes. It was then water-quenched, and naturally aged for 10 minutes. Finally, it underwent artificial annealing at 180°C for 20 minutes. In 2019, the TEM samples were made by electropolishing and FIB. The samples were stored in air at room temperature until the experiment in May 2022.

Table 3.1: The compossition of the aluminium alloy sample, exluding Al.

	Si	**Mg**	**Cu**	**Fe**	**Mn**	**Zn**	**Ti**	**Cr**
wt.%	0.6600	0.8600	0.2100	0.3600	0.2900	0.1500	0.03000	0.01000
at.%	0.3051	0.4594	0.4293	0.8360	0.06875	0.02978	0.008131	0.002497

The scanned region contains a thickness gradient; 131.8 ± 1.225 nm near the top, 99.65 ± 2.157 nm in the middle, and 69.82 ± 0.902 nm at the bottom, as shown in Figure 3.1. This thickness was determined by Jørgen A. Sørhaug, using convergent beam electron diffraction (CBED) images recorded of an area in sample after the experiment.

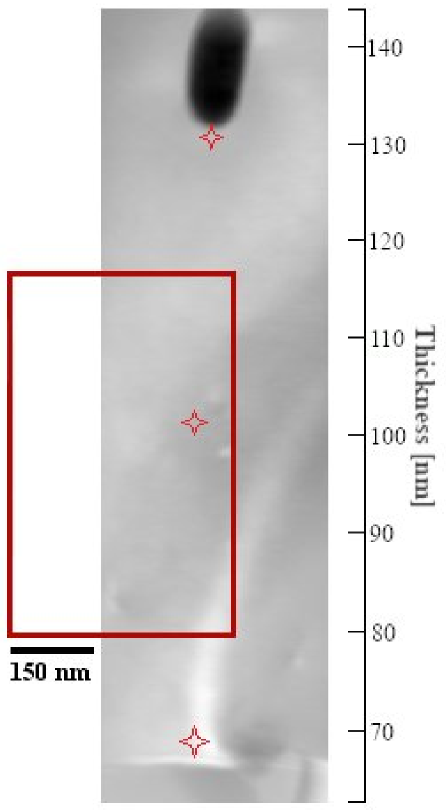


Figure 3.1: The area of the sample used in the scanning CBED. The stars mark the positions at which the thickness was calculated. The red frame is the area considered in this thesis.

3.2 SPED Acquisition and Datasets

The SPED datasets were collected using a JEOL JEM-2100F operated with an acceleration voltage of 200 kV. Each image was recorded with a precession frequency of 100 Hz, and the precession angle was 0.7°(12 mrad). The detector used was a Merlin direct electron detector from Quantum Detectors [24]. The datasets were collected along the ¡001¿ ZA in aluminium. Additional values and their ranges are shown in Table 3.2.

Table 3.2: The paramaters used in the acquisition of the SPED datasets.

Parameter	**Value Range**
Precession Angle (ϕ)	0.7°
Acceleration Voltage	200 kV
Dwell Time	10, 20 ms
Step Length	1.775 - 2.778 nm
Precession Frequency	100 Hz
Scan Rotation	216°

The scans were done over the sample as it was *in-situ* heated from room temperature to 520.1°C, with a heating rate of 0.01K/s. The scans vary in size and resolution. These values are presented in Table 3.3. Each diffraction pattern is 256×256 pixels.

Table 3.3: The sizes, temperature ranges and scan durations of each dataset.

Dataset	**Size [pixels]**	**Size [nm]**	**Start T [°C]**	**End T [°C]**	**Scan duration [min]**
S1	256 × 800	474 × 1420	30	72.6	71
S2	256 × 800	474 × 1420	85.9	128.5	71
S3	256 × 800	474 × 1420	148.4	191	71
S4	256 × 384	474 × 711	194.5	214.9	34
S5	256 × 384	474 × 711	215.4	235.8	34
S6	256 × 384	474 × 711	236.5	256.9	34
S7	256 × 384	474 × 711	257.5	277.9	34
S8	256 × 768	474 × 1420	278.5	299.3	35.5
S9	256 × 768	474 × 1420	302.2	323.5	35.5
S10	288 × 640	666.7 × 1481.6	324.5	344.3	33
S11	352 × 512	977.9 × 1422	346	365.2	32
S12	352 × 512	977.9 × 1422	365.8	384.2	32
S13	352 × 512	1140 × 1660	389	408.2	32
S14	352 × 512	1140 × 1660	-	-	32
S15	352 × 512	1140 × 1660	432.2	451.4	32
S16	416 × 416	1540 × 1540	457.0	475.6	31
S17	416 × 416	1540 × 1540	478.0	496.6	31
S18	416 × 416	1540 × 1540	501.5	520.1	31

In this thesis, only scans S1 to S11 are analysed, as virtual dark-field images reveal that most of the precipitates have dissolved after S11.

3.3 Data Processing Tools

This section presents the data processing tools used in this thesis. It describes the Python libraries used, virtual imaging, and the theory behind the decomposition methods used.

One data processing was done using the Python library `HyperSpy` [22].The SPED datasets were first converted to HyperSpy files. In HyperSpy, the datasets are loaded with 2D navigation and signal spaces, as shown in Figure 3.2. The navigation space contains the real space positions, and the signal space contains the diffraction pattern at the corresponding positions. Additionally, the Python package `Pyxem` was used [23]. Pyxem is an open source library acting as an extension to HyperSpy, providing tools for analysing 4D diffraction data.

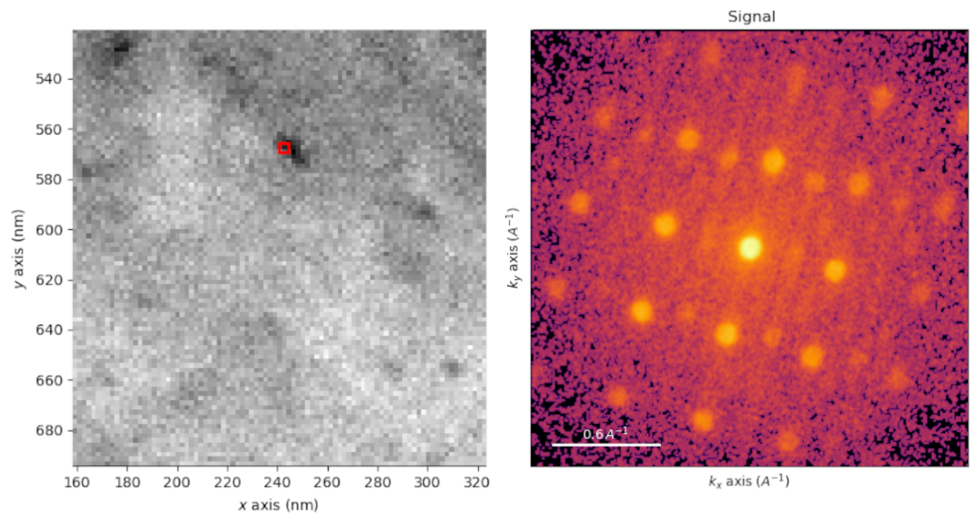


Figure 3.2: The plotting of a 4D HyperSpy dataset. On the right is the navigation axes in the navigator, with the red square indicating which pixel the diffraction pattern is from. The corresponding diffraction pattern, in signal space, is seen on the right.

3.3.1 Virtual Imaging

Using virtual apertures as regions of interest in HyperSpy, virtual dark field (VDF) and bright field (VBF) images can be created. Figure 3.3 shows the creations of VDF and VBF images from an example dataset. Each region of interest used in the integrated intensity in Pyxem are shown over the maximum of the stack of diffraction patterns.

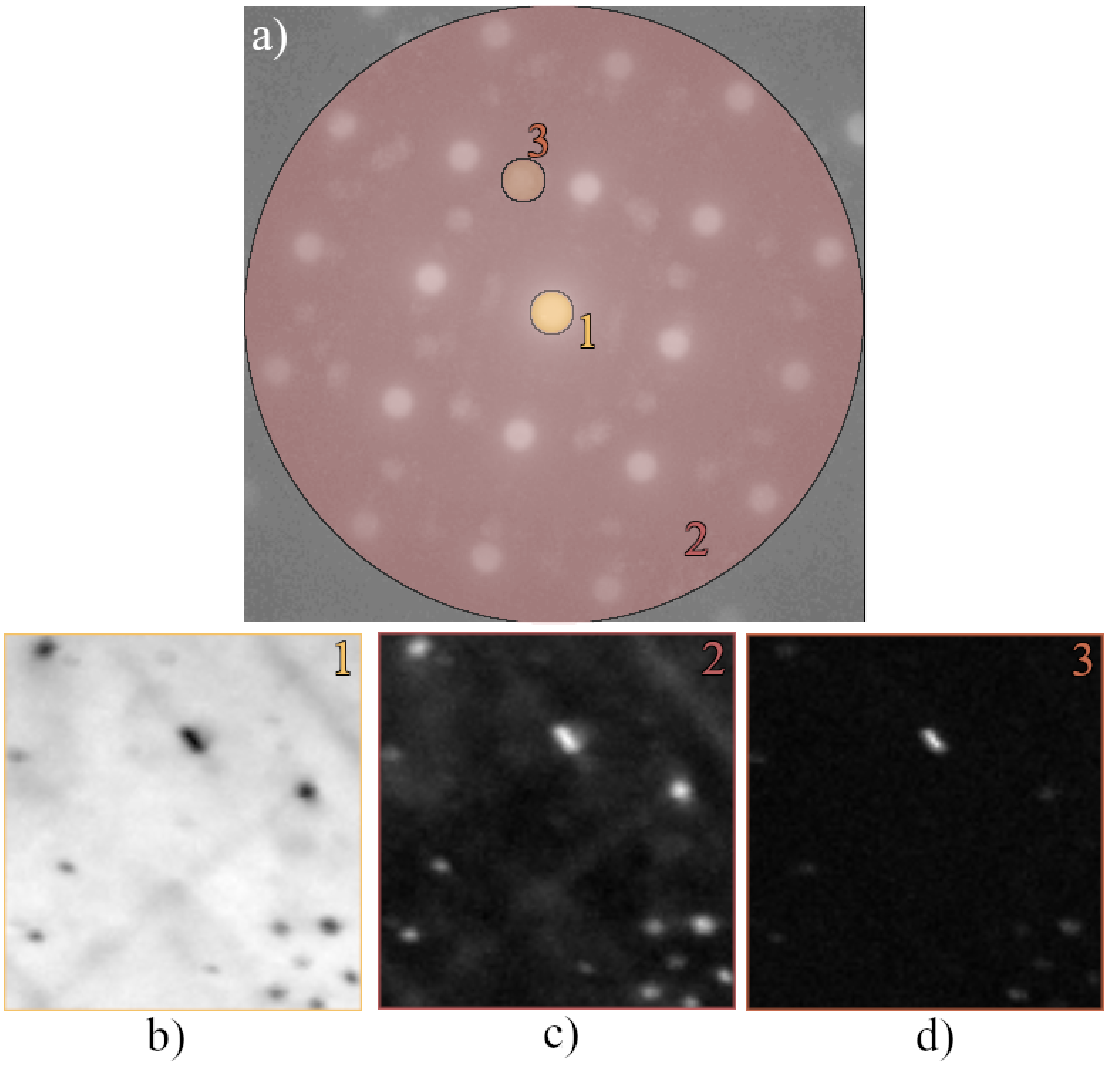


Figure 3.3: Virtual images created using various virtual apertures a) VBF using only the central beam. b) VDF using the largest possible circle, excluding the central beam. c) shows selects only one precipitate reflection, resulting in only the precipitates that give this reflection showing in the VDF.

3.3.2 Singular Value Decomposition

In order to choose the number of components to use in the non-negative matrix factorisation, singular value decompostion (SVD) can be used. This method examines how much of a dataset's total variance can be explained by each component in the decomposition [25, 26].

Let X be a data matrix defined by

$$X = \begin{bmatrix} x_{11} & \dots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{N1} & \dots & x_{NM} \end{bmatrix}. \tag{3.1}$$

where each column vector $\mathbf{x}$ is a non-negative vector with N observations [27]. In our case with SPED, the data matrix X is the SPED dataset.

Given the matrix X, there exists orthognal matrices U and V such that

$$X = U\Sigma V^T, \tag{3.2}$$

where S is a diagonal matrix [28]. The columns of U are the orthonormal eigenvectors of XX^T, while the columns of V are the orthonormal eigenvectors of X^TX. The values in S are called the singular values of X, and are correlated to the amount of variance accounted for by the eigenvectors [28]. SVD results can be visualised by scree plots. Scree plots are graphs showing the variance per component on a logarithmic scale.

3.3.3 Non-negative Matrix Factorisation

As in the previous section, we define X to be a data matrix defined by

$$X = \begin{bmatrix} x_{11} & \dots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{N1} & \dots & x_{NM} \end{bmatrix}. \tag{3.3}$$

where each column vector $\mathbf{x}$ is a non-negative vector with N observations [27].

The non-negative matrix factorisation (NMF) finds a decomposition of X, with dimensions $M \times N$, into two matrices U and V. The matrix U has dimensions $M \times L$ and is the basis matrix, and V is the coefficient matrix with dimensions $L \times N$. These two matrices approximate the matrix X, $X \approx UV$. This can be written as

$$\mathbf{x}_j = \sum_{i=1}^{L} \mathbf{u}_i \cdot \mathbf{V}_{ij}, \tag{3.4}$$

where L is the number of components desired [27].

Decomposition with NMF in HyperSpy returns two signals, *loadings* and *factors*, the number of each corresponding to the number of components. The factors contain the unique components that approximate the entire dataset of diffraction patterns. In this case, the signal contains, among others, approximate diffraction patterns. The loadings are 2D images that show how well the original diffraction patterns at each pixel in the dataset correspond to each factor. The loadings therefore contain information about the locations of the different precipitates. An example of how a loading map and a factor can look like it is shown in Figure 3.4. The greater the match is, the greater the value in the loading map is.

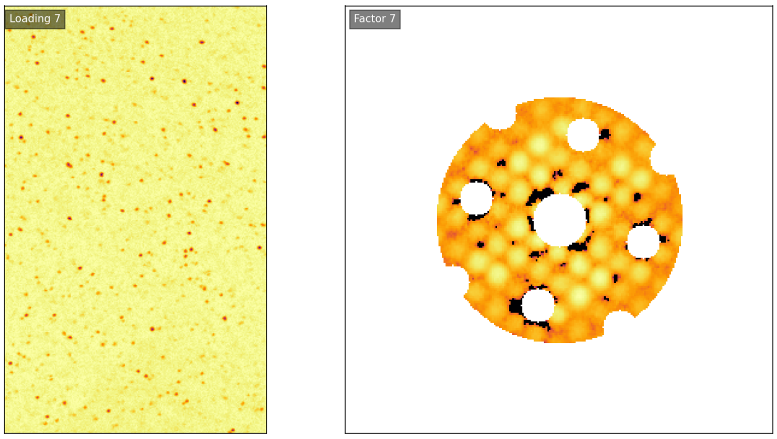


Figure 3.4: An example pair of loading map and factor.

3.4 Data Processing Methods

This section describes the methods followed in the thesis, from raw data to finished phase maps. The method is illustrated in 3.5. First, the preprocessing steps are described, followed by cropping and virtual imaging. Next, the decompositions and steps involved. Next, the data processing methods explored in the creation of phase maps are explained. At last, diffraction pattern simulation are described.

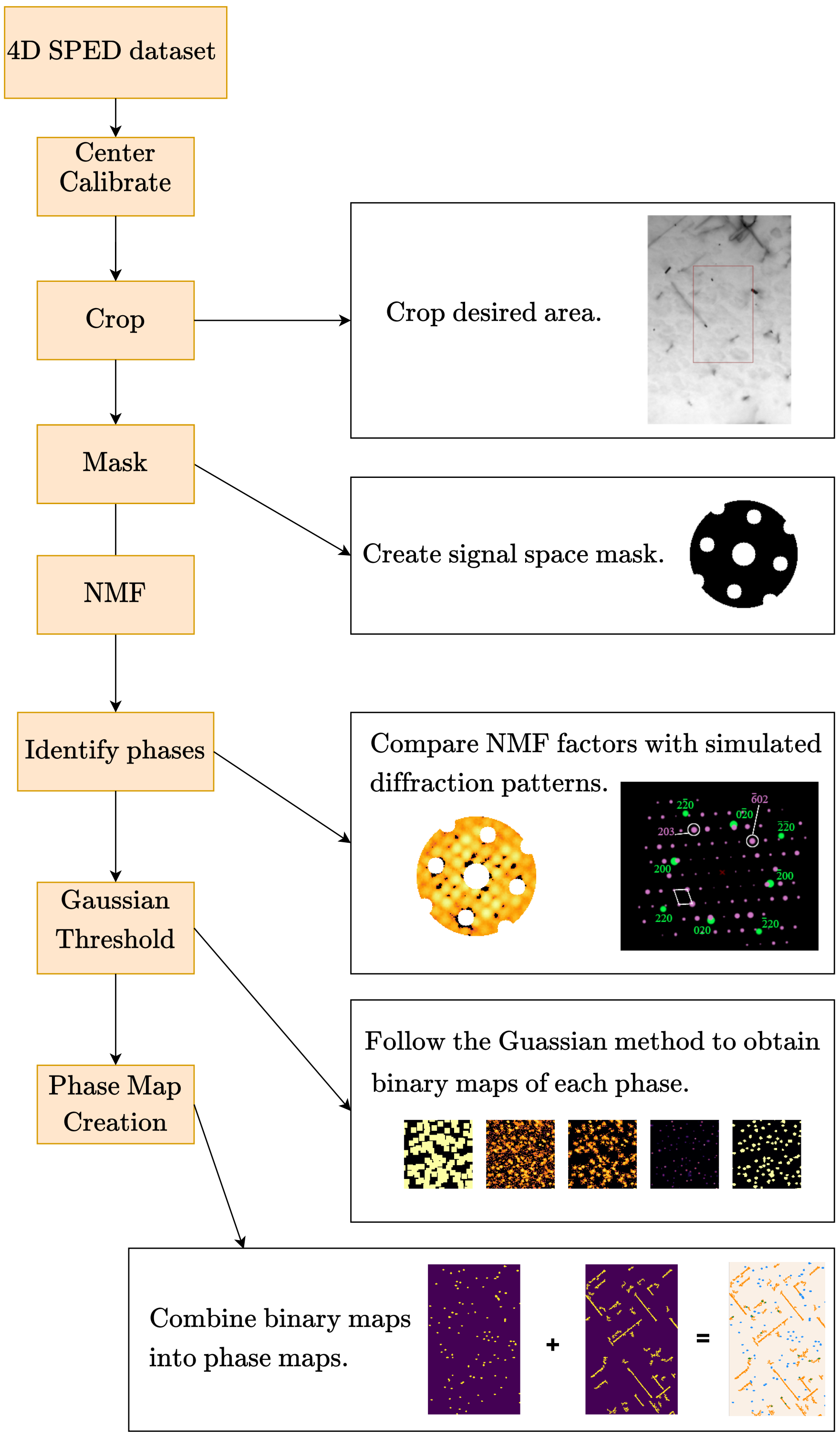

4D SPED dataset
Center Calibrate
Crop
Mask
NMF
Identify phases
Gaussian Threshold
Phase Map Creation
Crop desired area.
Create signal space mask.
Compare NMF factors with simulated diffraction patterns.
Follow the Guassian method to obtain binary maps of each phase.
Combine binary maps into phase maps.
+
=

Figure 3.5: A schematic representation of the method followed.

3.4.1 Preprocessing

First, the datasets were centered. This was done by using the center of mass method from Pyxem. This determines the center of mass for each induvidual diffraction pattern using the area around the direct beam. Next, the required shift to move the central beam to the center of the image is found. Then, each diffraction pattern is moved to align with the center.

Next, each dataset was calibrated in both real and reciprocal space. In real space, the calibration was done using the parameters in Table 3.3 to find the relationship between nm and pixels. For the diffraction patterns in reciprocal space, the 400 and $\bar{4}00$ reflections in aluminium were used to find the relationship between pixels and $Å^{-1}$.

In order to simplify the comparison of the scans, the same area was identified in each scan. This was done by stacking the VBF images from each scan and then marking the largest possible area that covered in all the VBF images. This area is shown in Figure 3.6.

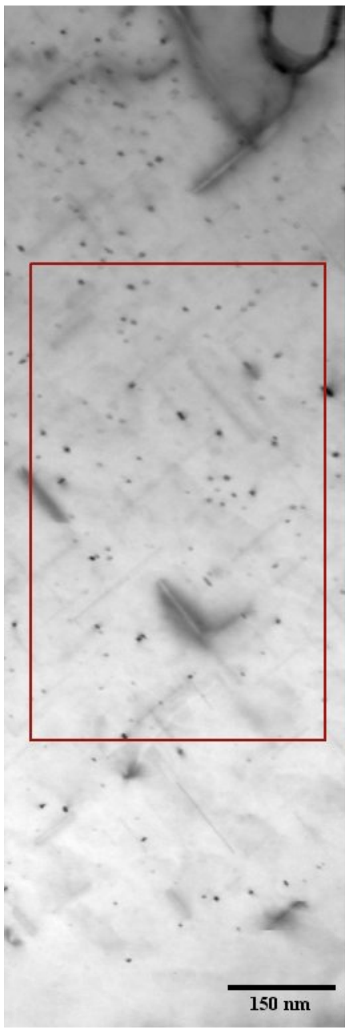


Figure 3.6: The largest possible area choosen, here shown over S8.

The position of the chosen area in Figure 3.6 was then measured in pixels in each scan S1-S11, and converted to nm. Then each dataset was cropped to fit the chosen area. This resulted in an 407×657 nm area.

3.4.2 Singular Value Decomposition

In order to choose the number of components to use in the NMF, a principal components analysis (PCA) was done with singular value decomposition (SVD). In order to investigate the number of componets to use in the NMF, SVD was done on three of the datasets; S3, S5 and S8. From the resulting scree plots, the number of components to use in the initial investigation of NMF components was chosen. The numbers are shown in Table 3.4.

Before the SVD was performed, a signal mask was created to mask out all Al reflections in the diffraction patterns. This was done as the goal is to obtain information about the precipitates, and not Al in itself. The mask also limits the reciprocal space to within the {220} reflections in Al.

Table 3.4: The numbers of compononents chosen for the first 3 NMF decompositions of S3, S5 and S8.

Scan	**1st nr of components**	**2nd nr of components**	**3rd nr of components**
S3	5	9	17
S5	9	12	19
S8	10	14	19

As decomposition with singular value decomposition is demanding, these methods were run on the IDUN cluster at the high performance computing group at NTNU [29].

3.4.3 Non-negative Matrix Factorisation (NMF)

The NMF were first performed on S3, S5 and S8 with the number of components stated in Table 3.4. The same mask as in the SVD stated above was again used in the NMF.

In order to choose the number of components used in the NMF decomposition of the remaining datasets, phase maps were created. These phase maps were created using the global threshold method described below. S3, S5 and S8 had three decompositions each, which resulted in 9 total phase maps. In order to compare the phase maps, the difference in the three phase maps for each phase were calculated pixel by pixel. Based on the difference, which is explained in the discussion, the highest number of components were choosen; 19. Next, NMF with the same masks with 19 components were done on the remaining datasets.

As decomposition with singular value decomposition and non-negative matrix factorzation is demanding, these methods were run on the IDUN cluster at the high performance computing group at NTNU [29]. Similarly to SVD, NMF is also a demanding algorithm. Therefore, this was also run on the IDUN cluster at NTNU.

3.4.4 Data Processing Methods

In the following sections, the 3 different methods attempted in the creation of the phase maps are described. However, they all share some common steps that will be described first.

First, the loading maps containing the same phases are combined into one, making sure the intensities are similar. Next, the background is removed from each of these new components. Then, one can proceed with one of the three methods described below. Further details can be seen in the code in Appendix A.

Global Threshold Method

In this method, a single threshold value is found for each of the combined loading maps. This value is found by applying the *threshold_yen* threshold method from scikit-image [30]. This value is then applied to the loading map, resulting in a binary image. This binary image is then a phase map for this specific phase. The method is repeated for each phase in the dataset.

Individual Threshold Method

This method utilizes the peak finder function from Pyxem. The first step is to use this function to locate the precipitates in the loading map. Next, square masks are created covering each precipitate position. Only areas within these squares are considered next. A threshold value is found for each square, using the same threhold function as above. However, this method results in some debris in the form of small amounts of pixels or single pixels appearing around the edges of the precipitates, disconnected from the precipitate itself. The last step is the removal of this debris, using the *label* library from scikit-image. Every area marked as True in the phase map smaller than a certain amount of pixels is removed. The steps of this method are shown in Figure 3.7. Again, the method is repeated for each phase in the dataset.

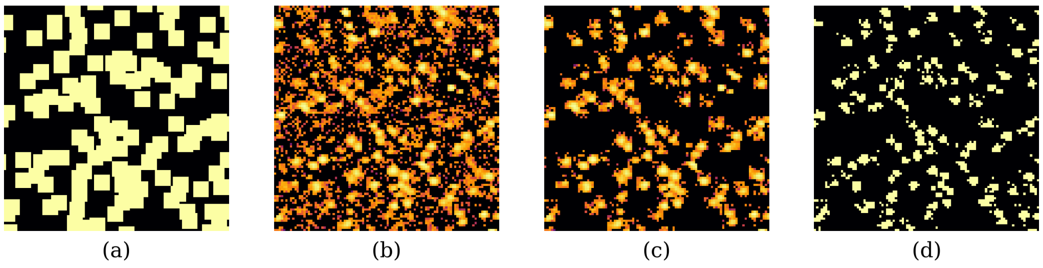


Figure 3.7: The steps in the individual threshold method. (a) is the masks created at each precipitate position. (b) shows the combined loading map. (c) is the areas kept by the mask. (d) show the binary image created by each mask; a phase map for this phase.

Gaussian Method

After the precipitate positions have been found with peak finder, the intensities of the pixels at the exact positions given by the peak finder function are doubled. Then, the masks are created in the same manner as in the individual threshold method. After the mask has been applied, a two dimensional Gaussian function is fitted to each square. Then, a threshold value is found for each square containing a Guassian function, using the threshold yen function. The resulting binary images are then combined into a phase map. The steps of this method are shown in Figure 3.8.

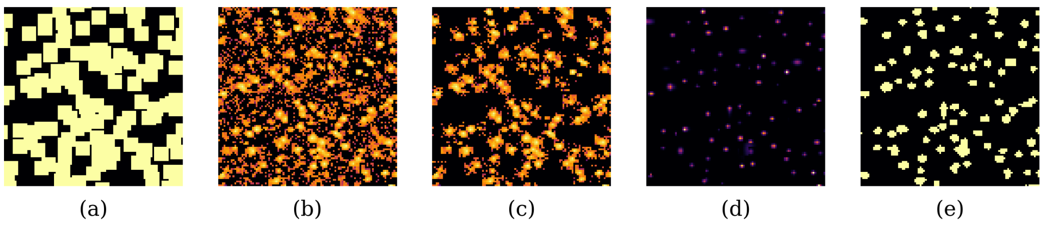


Figure 3.8: The steps in the individual threshold method. (a) is the masks created at each precipitate position. (b) shows the combined loading map, with intensities doubled at the position of each precipitate. (c) is the areas kept by the mask. (d) shows the fitted Gaussian function. (e) show the binary image created by each mask; a phase map for this phase.

3.4.5 Phase Map Creation

Comparing the three different methods described above, the method chosen to proceed with in creation of the phase maps was the Gaussian method. The justification for this will be explained in the discussion.

To create the phase maps, the phase maps for each individual phase are combined. However, as the individual maps can overlap, a new ”not indexed” phase is added containing the overlapping areas from the individual maps. Each individual map is assigned an integer value, and the individual maps are combined, giving each phase one color in the final phase map.

3.5 Diffraction Pattern Simulations

In order to identify the different patterns appearing in the NMF factors, diffraction patterns from the suspected phases were simulated. The simulations were done in ReciPro, an open-source crystallographic software [31]. In order to simulate diffraction patterns, information additional to the parameters presented in Table 2.2 are needed; the atom positions. These, along with the other required parameters, are presented in Table 3.5 for each simulated phase. The structures of each of these phases can be seen in Figure 3.9. These are also simulated in ReciPro. Each DP simulation was simulated considering kinetmatic diffraction, along with exitation error. The intensities are based on the structure factors of each phase.

Table 3.5: Space group, lattice parameters, and atomic positions for each simulated phase.

Phase	**Space Group**	**Lattice Parameters**	**Atomic Positions**			**Atom**
			x	**y**	**z**	
C	$P2_1$	$a = 10.32$, $b = 4.05$, $\beta = 100.9°$, $c = 8.10$	0.50	0.50	0.50	Si
			0.83	0.50	0.08	Si
			0.83	0.50	0.58	Si
			0.62	0.00	0.03	Mg
			0.62	0.00	0.53	Mg
			0.93	0.00	0.36	Mg
			0.93	0.00	0.86	Mg
			0.74	0.50	0.76	Al
			0.70	0.50	0.36	Cu
β″	C2/m	$a = 15.16$, $b = 4.05$, $\beta = 105.3°$, $c = 6.74$	0.00	0.00	0.00	Mg
			0.35	0.00	0.09	Mg
			0.42	0.00	0.65	Mg
			0.05	0.00	0.65	Si
			0.19	0.00	0.22	Si
			0.21	0.00	0.63	Al
Q′	P6	$a = 10.32$, $b = 10.32$, $c = 4.05$, $\gamma = 120°$	0.33	0.66	0.00	Cu
			0.66	0.66	0.00	Si
			0.59	0.87	0.00	Si
			0.23	0.99	0.00	Al
			0.62	0.13	0.00	Mg

Another requirement for the simulations are the orientation relations between Al and the phases. These are given by [20, 32]:

$$\mathrm{Q}': \ [0001]_{\mathrm{Q}'}||[001]_{\mathrm{Al}} \text{ and } (1120)_{\mathrm{Q}'}||(510)_{\mathrm{Al}} \tag{3.5}$$

$$\beta'': \ (010)_{\beta''}||\{001\}_{\mathrm{Al}} \text{ and } [001]_{\beta''}||\langle 310\rangle_{\mathrm{Al}} \text{ and } [100]_{\beta''}||\langle 230\rangle_{\mathrm{Al}} \tag{3.6}$$

$$\mathrm{C}: \ (010)_{\mathrm{C}}||\{001\}_{\mathrm{Al}} \tag{3.7}$$

Using these relationships, the precipitates' rotation relative to Al could be found.

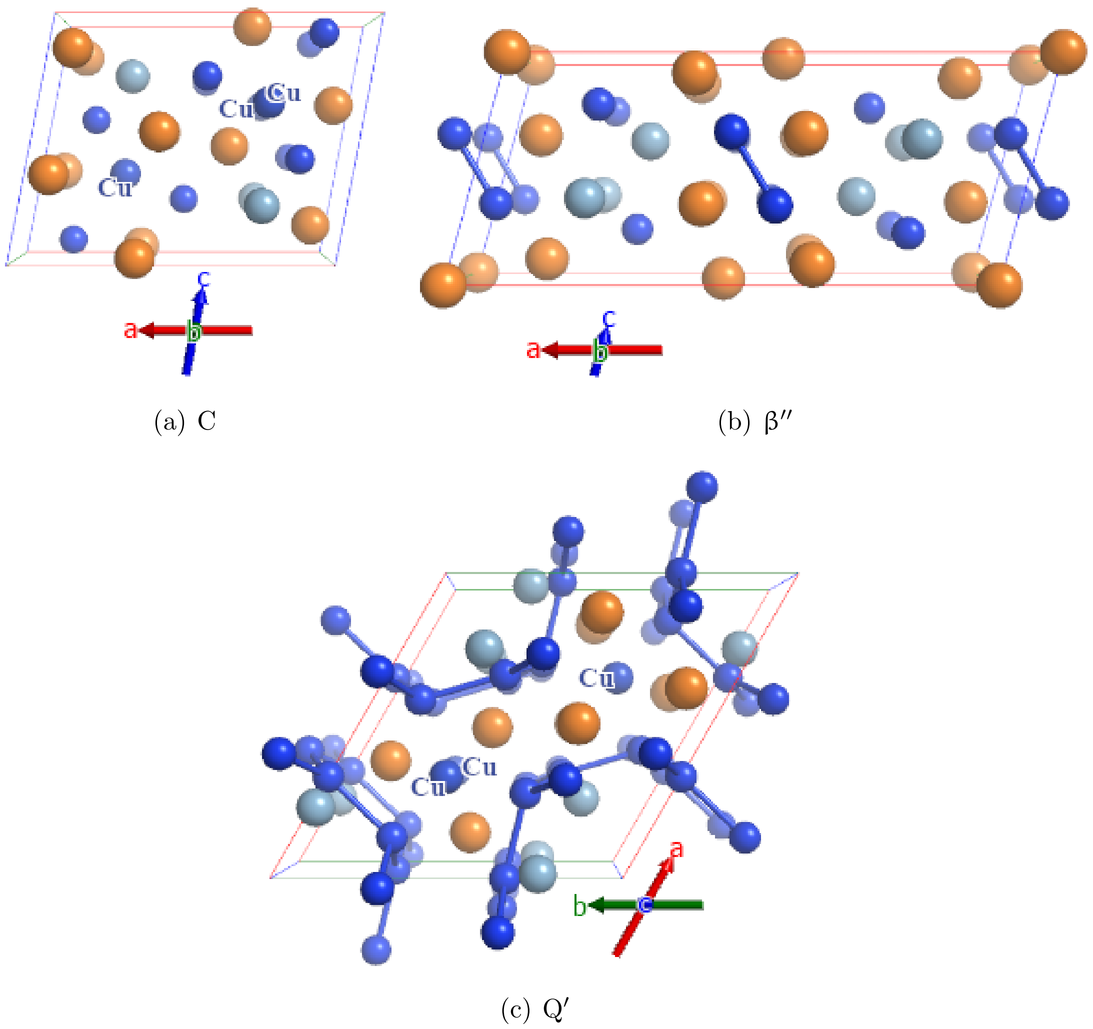


Figure 3.9: The structures of the simulated phases. Each phase is simulated in the ZAs parallel to the [001] ZA in Al. a) C in [010], b) β'' in [010] and c) Q′ in [001]. The orange, silver and blue atoms are Mg, Al, and Si, respectively, except the blue atoms labeled separately as Cu.

3.6 Precipitate Statistics

In order to count how many precipitates of each phase are present in the datasets, the number of peaks returned by *peak_finder* is used. The areas are calculating by summing over each individuall binary phase map. These numbers are then converted to nm^2.

3.7 Phase Transitions

In order to investigate the phase transitions in the phase maps, manual inspection was done. This was done by using the phase maps as the navigator in HyperSpy when plotting the datasets. In scans T9 to T11, every precipitate in the phase maps were checked to either confirm or disprove the results in the phase maps. Further, selected precipitates from various scans that were labeled as ”not indexed” were investigated both before and after the overlap. Phases that change color and label between phase maps were also investigated to see if a phase transition had occurred.

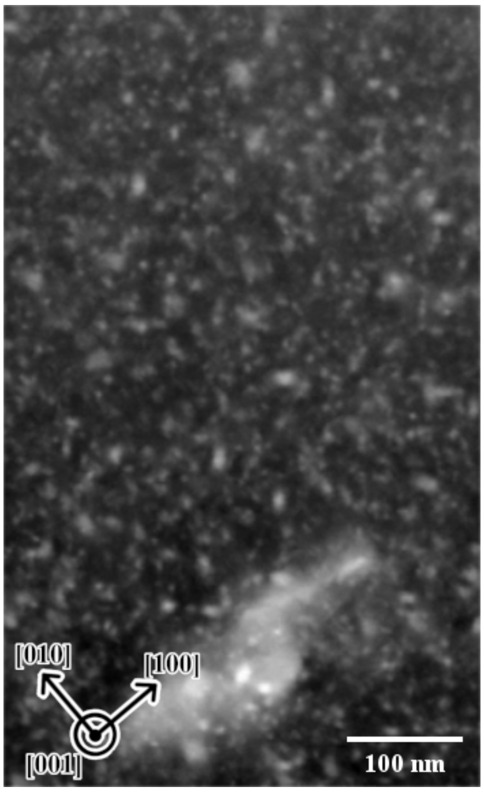
Chapter 4

Results

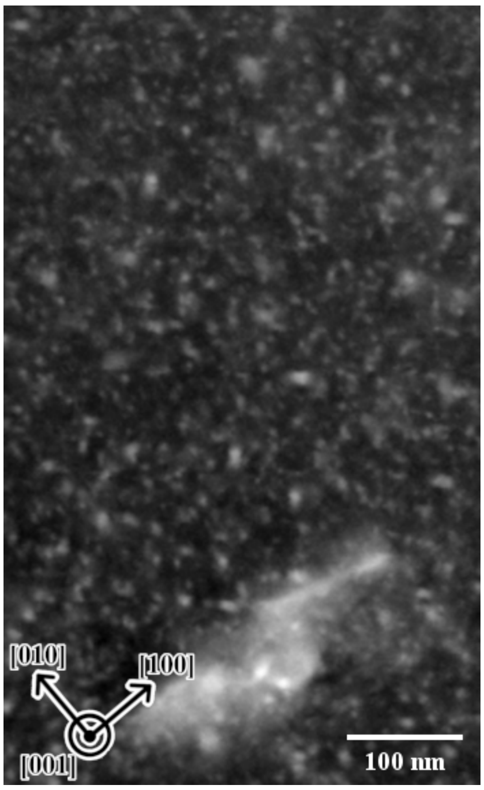
This chapter first presents the VDF images for each scan. Next, the results from the initial investigation regarding number of components to use in the NMF is presented. This includes scree plots from the scans, as well as the differences between the various number of components. The difference in the three phase map creation methods follows, with a phase map created using each method. Next, the completed phase maps using the Gaussian method is presented. The next section presents information about the precipitate evolution in the scans. This includes various precipitate statistics and an overview over which precipitates are present in each scan. Finally, the various phase transitions observed in the scans are presented.

4.1 Virtual Dark-field Images

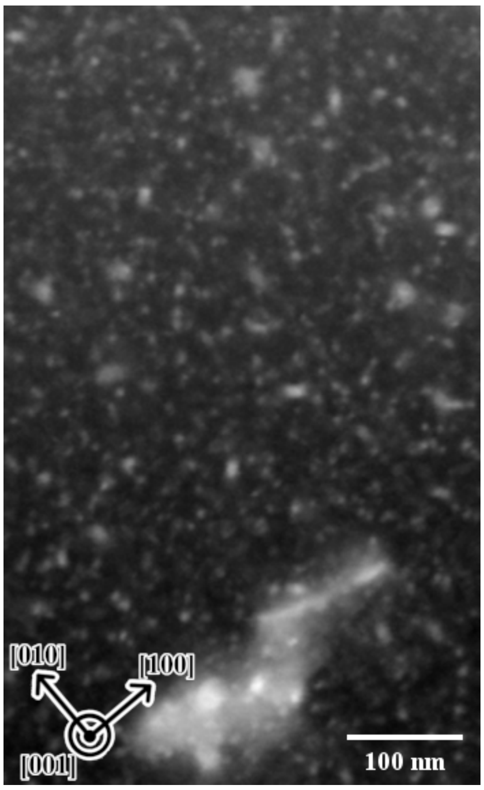
Figures 4.1 to 4.3 show the VDF images created from each scan.



(a) S1: 40-59°C

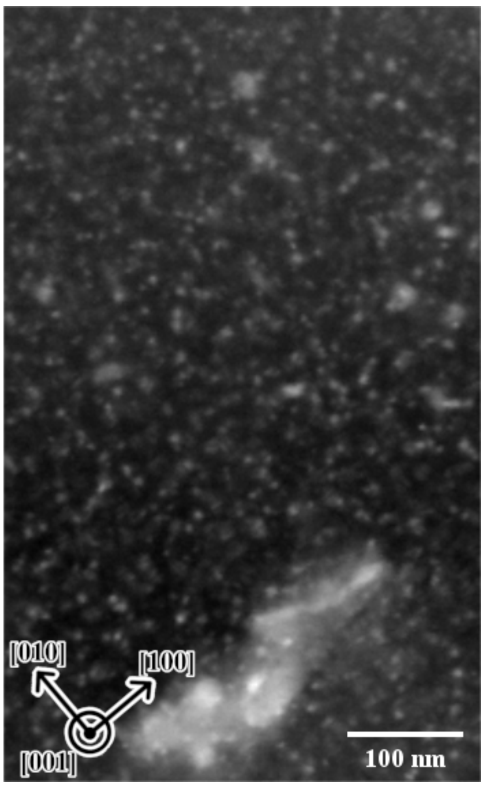


(b) S2: 96-115°

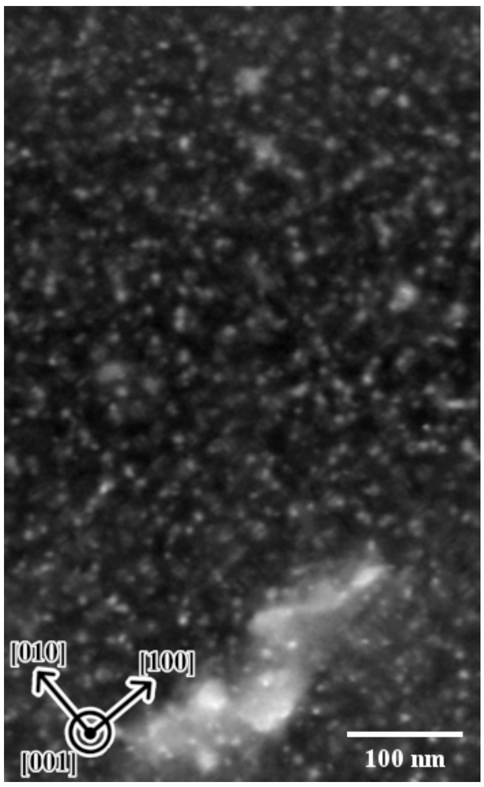


(c) S3: 161-180°

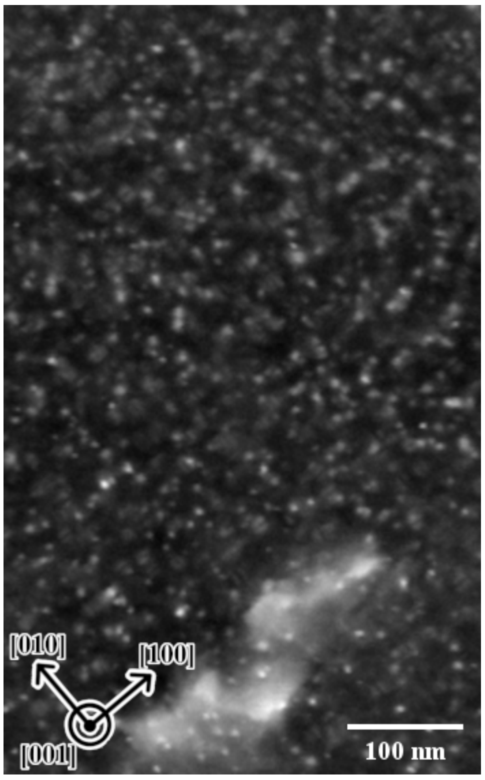
Figure 4.1: VDF images from a) S1, b) S2 and c) S3. In each VDF image, directions in the Al crystal is also indicated. The temperature ranges for the scans are also stated for each dataset.



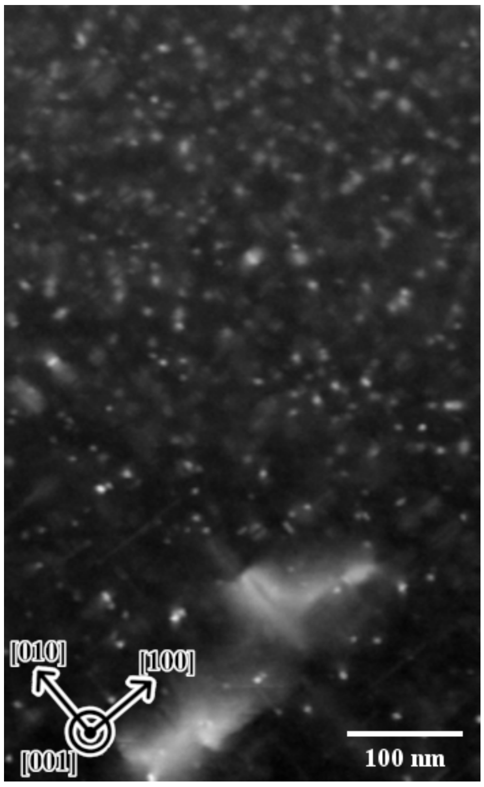
(a) S4: 196-215°C



(b) S5: 216-234°C

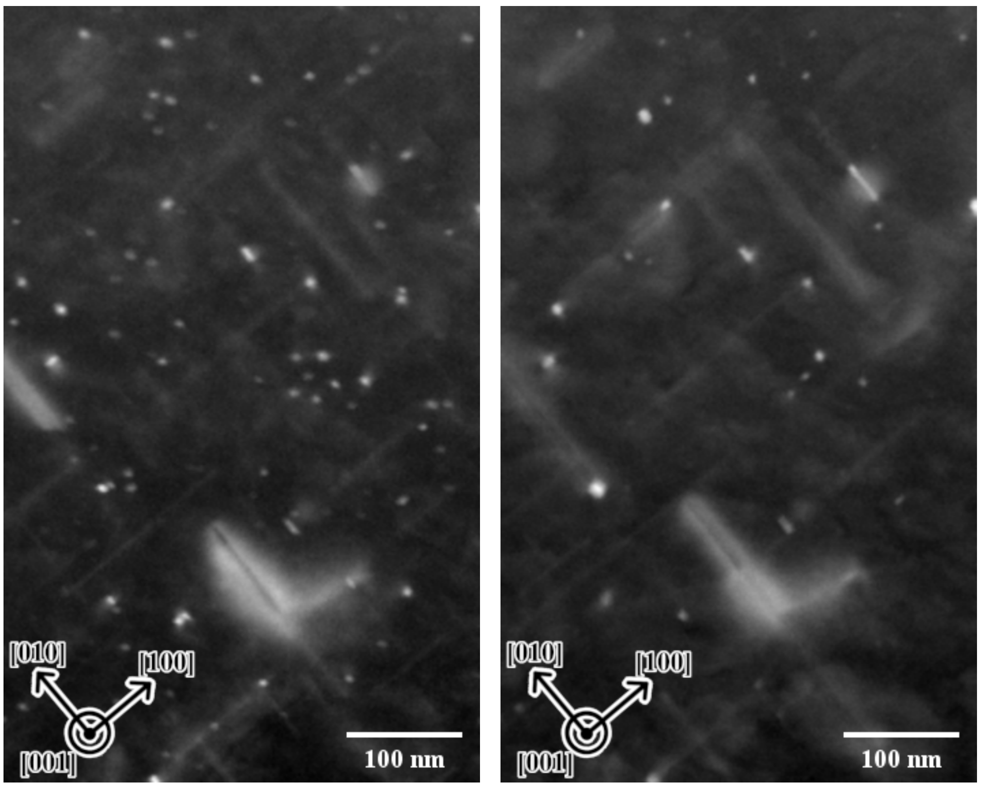


(c) S6: 237-255°C



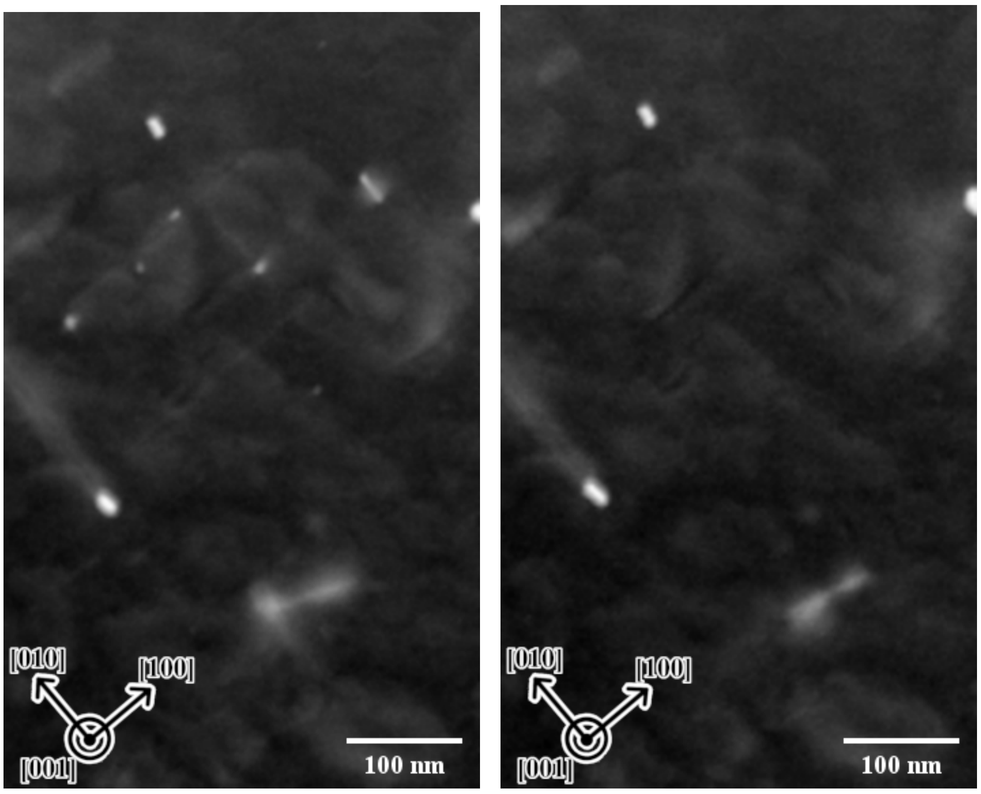
(d) S7: 257-276°C

Figure 4.2: VDF images from a) S4, b) S5, c) S6 and d) S7. In each VDF image, directions in the Al crystal is also indicated. The temperature ranges for the scans are also stated for each dataset.



(a) S8: 284-295°C

(b) S9: 307-317°C



(c) S10: 329-336°C

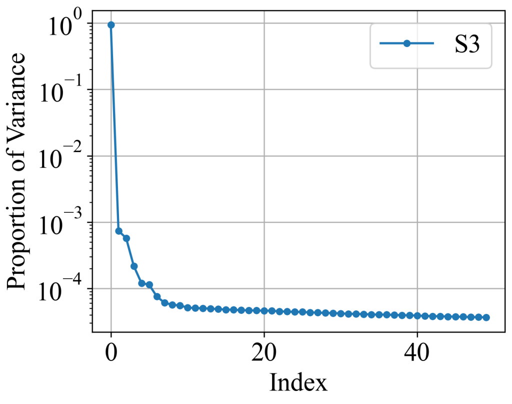
(d) S11: 350-356°C

Figure 4.3: VDF images from a) S8, b) S9, c) S10 and d) S11. In each VDF image, directions in the Al crystal is also indicated. The temperature ranges for the scans are also stated for each dataset.

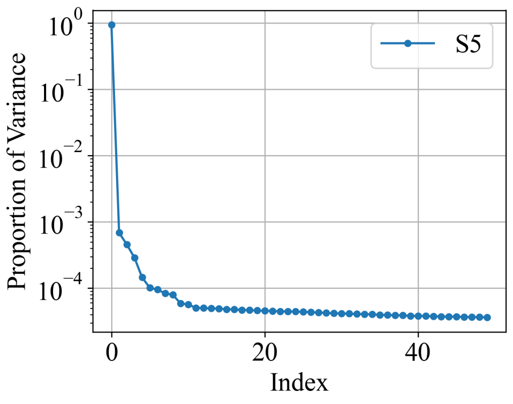
4.2 Choosing the Number of Components

Figure 4.4 shows the scree plots from the SVD of S3, S5, and S8. As seen, the variance per component rapidly decreases per component early on, but flattens considerably after a while. The scree plots from S3 and S5 are similar, whereas the scree plot from S8 has a considerably higher variance per component after 20 components compared to S3 and S5.

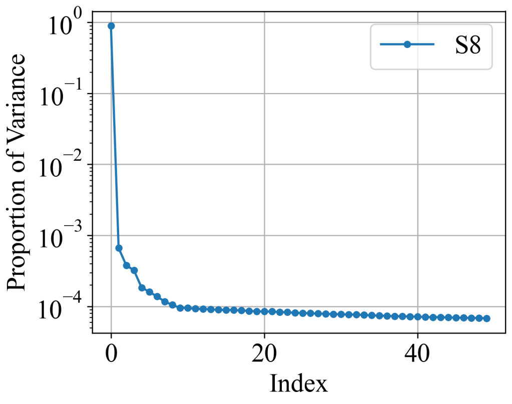
Table 4.1 shows the difference, calculated pixel by pixel, between phase maps based on NMF decompositions with different number of components for S3, S5 and S8. As the decomposition of S3 with 5 components returned no real diffraction patterns, this decomposition was disregarded.



(a)



(b)



(c)

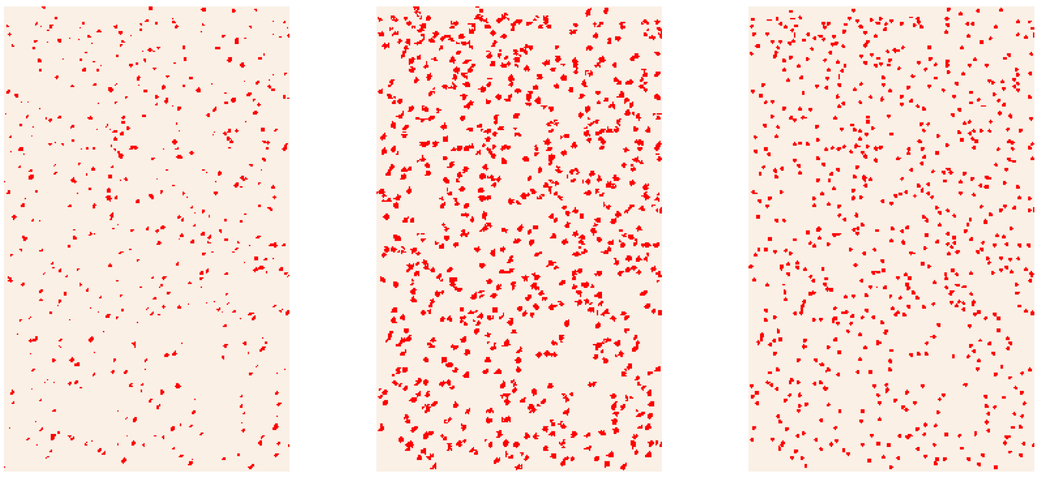
Figure 4.4: The scree plots from the SVD on S3, S5 and S8.

Table 4.1: The differences in the phase maps created with the 3 different numbers of components.

S3		S5		S8	
	Difference		**Difference**		**Difference**
5 vs 9	-	9 vs 12	1.42%	10 vs 14	0.74%
5 vs 17	-	9 vs 19	1.37%	10 vs 19	0.98%
9 vs 17	0.47%	12 vs 19	0.16%	14 vs 19	0.35%

4.3 Choice of Post-Processing Method

Figure 4.5 shows phase maps from S3 using the three data processing methods described in Chapter 3.4.4; the global threshold method, the individual threshold method and the Gaussian method.



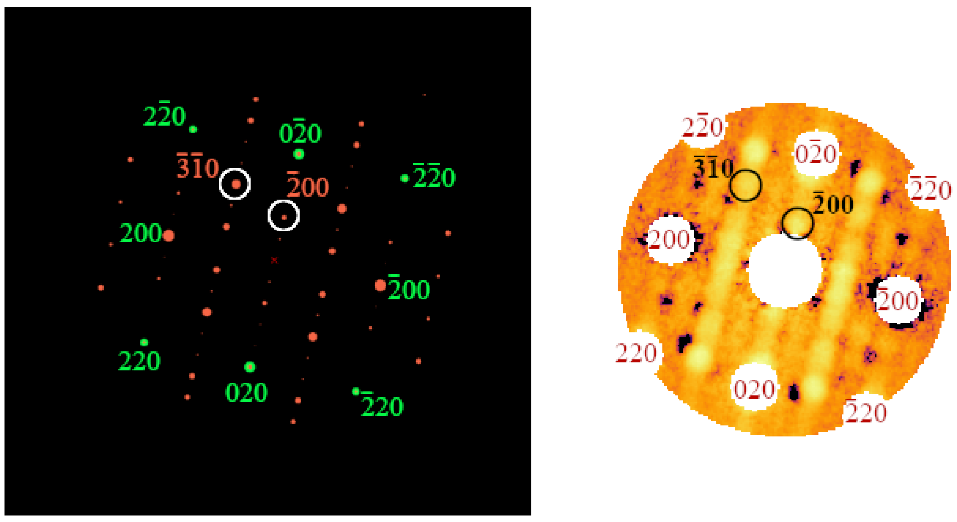
(a) (b) (c)

Figure 4.5: Phase maps from the same phase in S3 found using the three different methods. a) Global Threshold, b) Individual Threshold, c) Gaussian function.

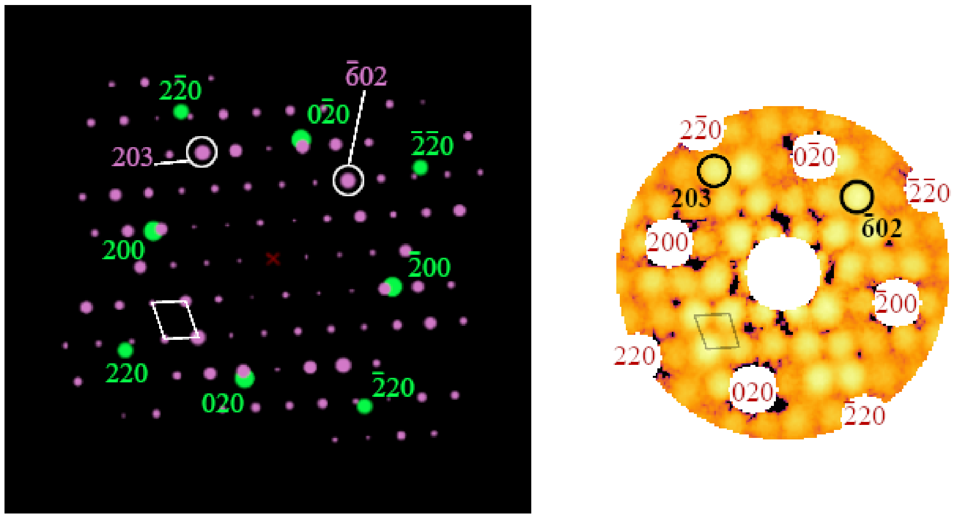
4.4 Precipitate Evolution

Here, the results extracted from the NMF are presented. First, diffraction patterns from C, β'', and Q' appearing as NMF factor are identified and indexed, and compared to simulated diffraction patterns. This is shown in Figure 4.6. Next, Figure 4.7 shows which phases that are present in each dataset S1 to S11. In the first dataset, only β'' is present, and is the only visible precipitate until S6, where L starts to appear. Then, in S7, only L is visible. In S9 both C and Q' are presents, and in S11, only Q' remains.

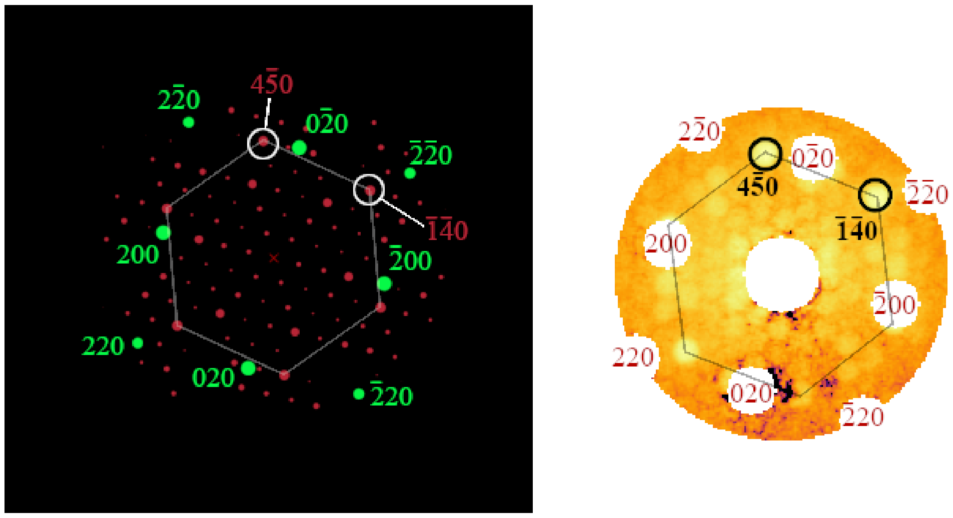
Figures 4.8 through 4.10 show the phase maps created from the datasets. Figure 4.8 a) also shows the assigned colors for each phase. Here, "not indexed" indicates an overlap between two or more phases and is labeled green. C, Q', β'' and L are all considered as growing out of the plane. "Inplane" indicates inplane needles of either of the phases combined.



(a)



(b)



(c)

Figure 4.6: Simulated patterns (right) compared to factors from the NMF decomposition (left). The larger radii in the simulated patterns correspond to higher intensity. Two reflections are identified in each component. a) C, b) β'', c) Q$'$.

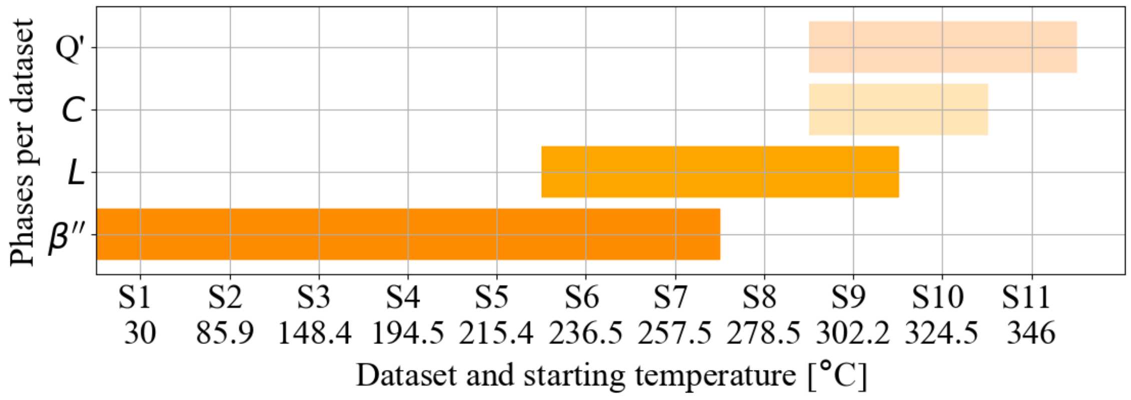
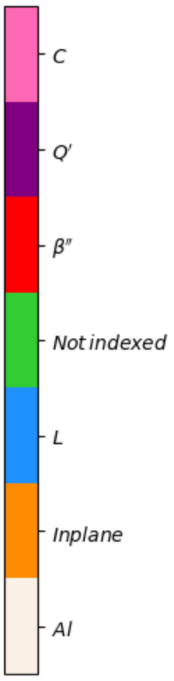
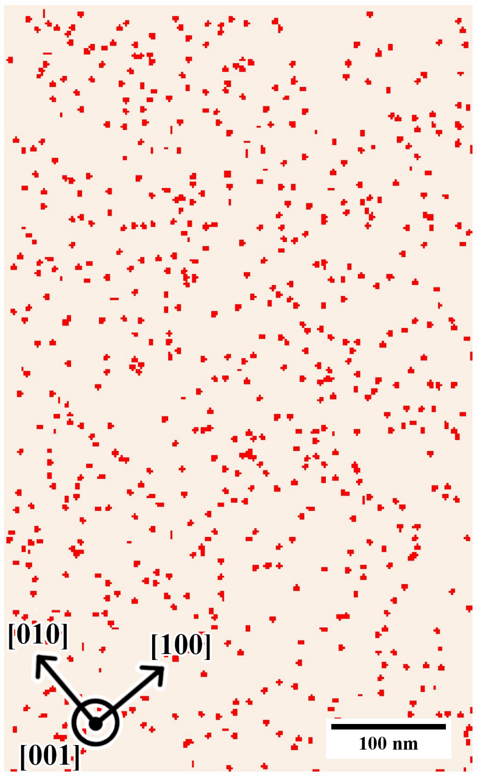


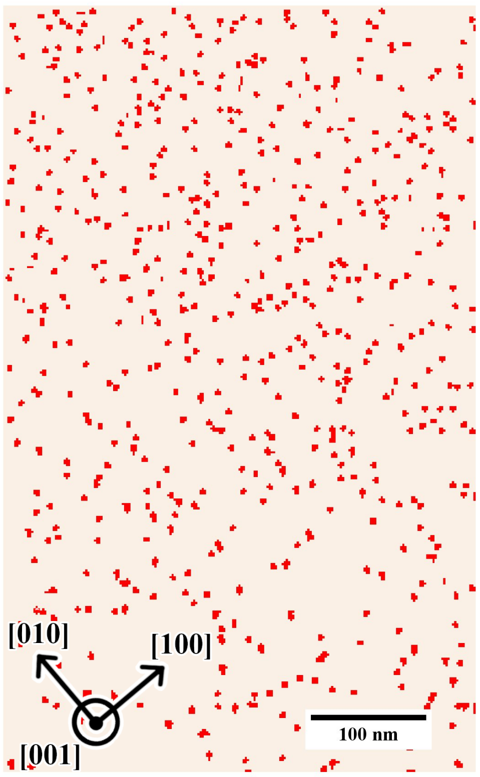
Figure 4.7: The phases appearing in the NMF decomposition of each dataset S1-S11.



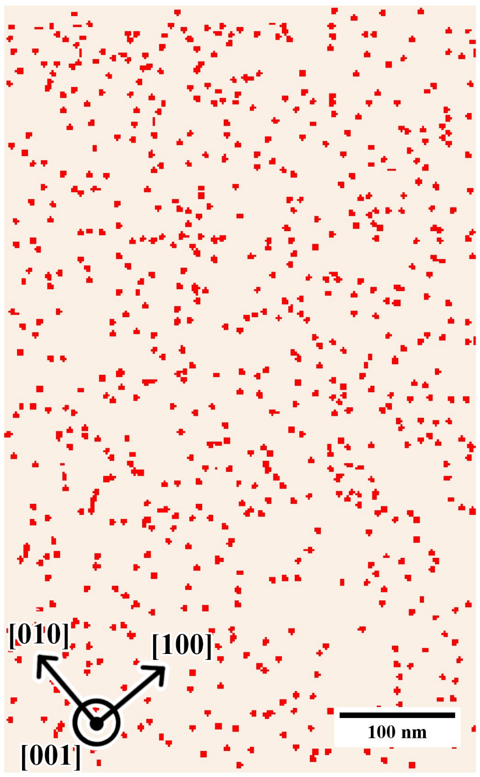
(a)



(b) S1: 40-59°C

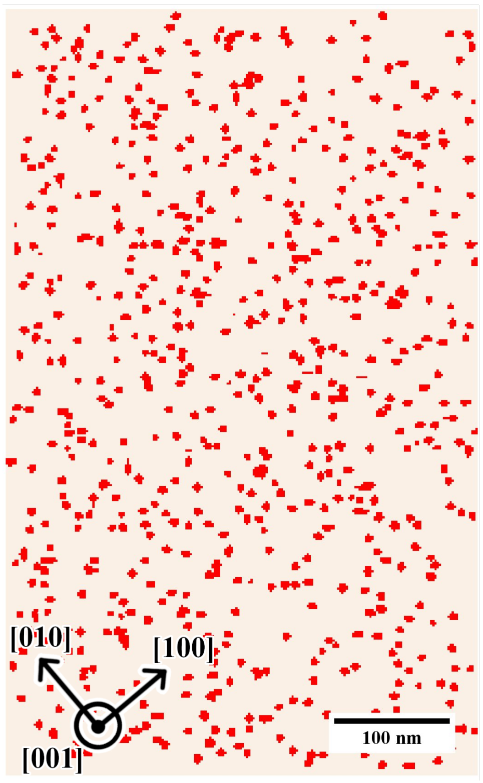


(c) S2: 96-115°C

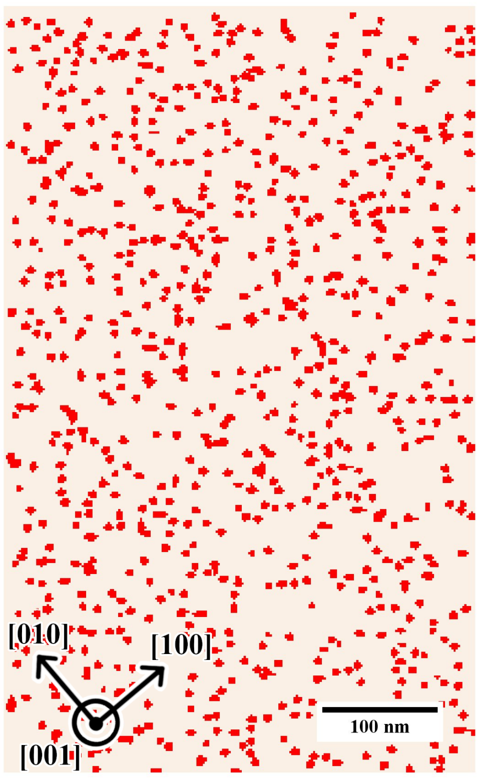


(d) S3: 161-180°C

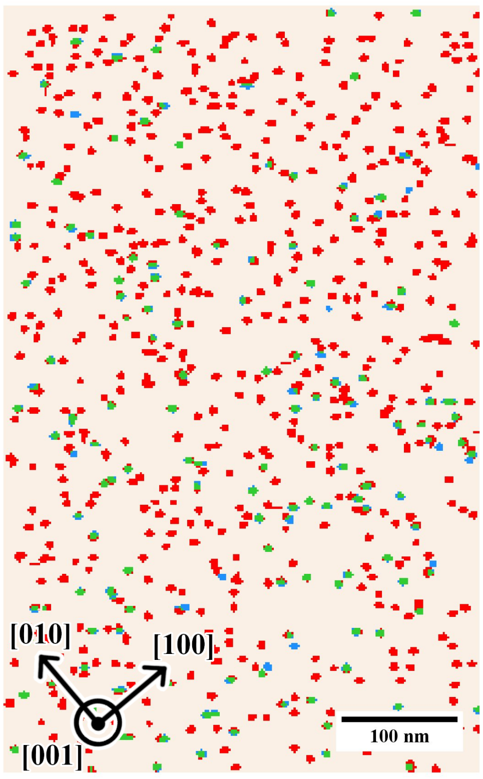
Figure 4.8: Phase maps for S1-S3. b) S1, c) S2, d) S3. a) contains the index for the phase maps. In each phase map, directions in the Al crystal are also indicated. The temperature ranges for the scans are also stated for each dataset.



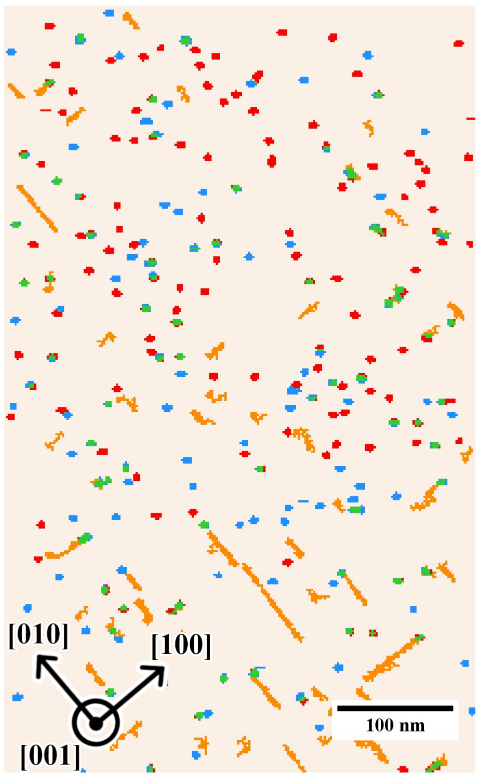
(a) S4: 196-215°C



(b) S5: 216-234°C

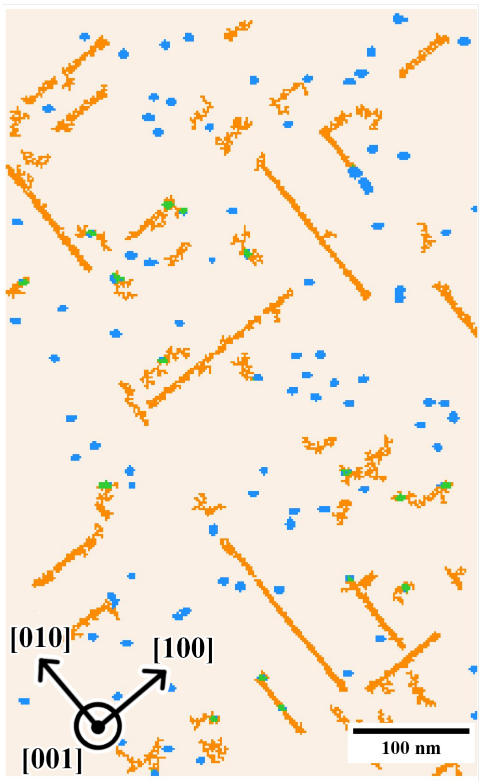


(c) S6: 237-255°C

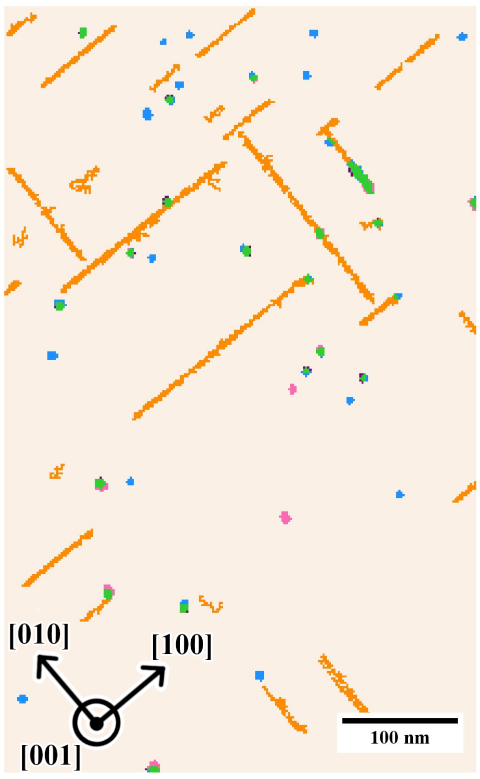


(d) S7: 257-276°C

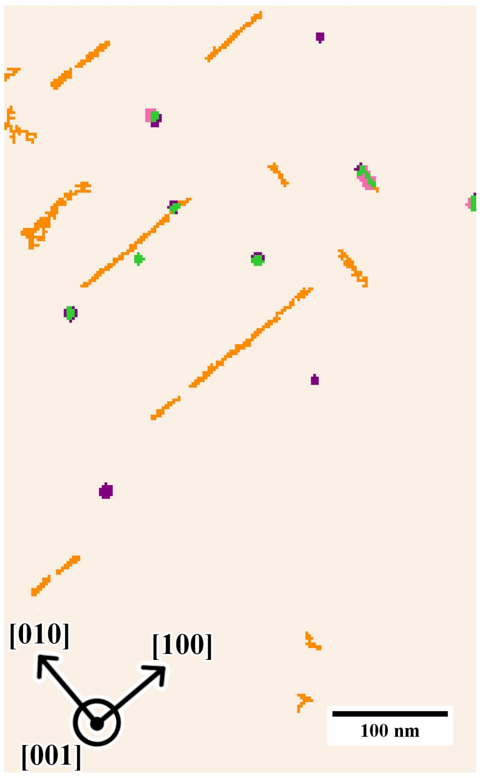
Figure 4.9: Phase maps for S4-S7. a) S4, b) S5, c) S6, d) S7. In each phase map, directions in the Al crystal are also indicated. The temperature ranges for the scans are also stated for each dataset.



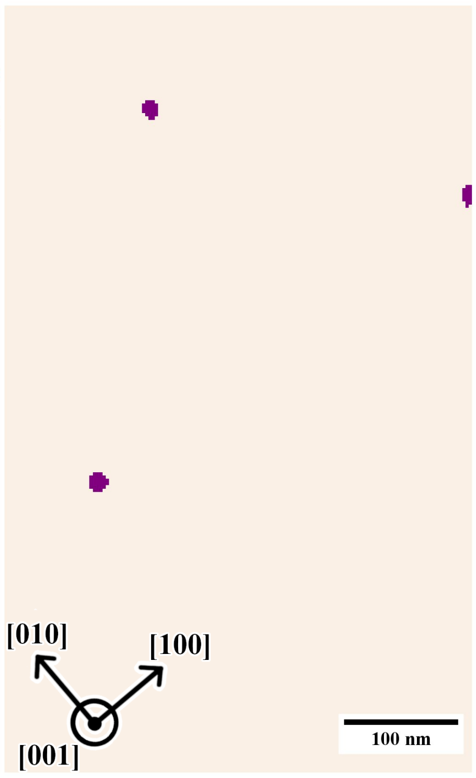
(a) S8: 284-294°C



(b) S9: 307-317°C



(c) S10: 328-336°C



(d) S11: 350-355°C

Figure 4.10: Phase maps for S8-S11. a) S8, b) S9, c) S10, d) S11. In each phase map, directions in the Al crystal are also indicated. The temperature ranges for the scans are also stated for each dataset.

4.5 Precipitate Statistics

This section presents various statistics about the precipitates. These are area per precipitate (cross section), length of inplane needles, area of precipitates as percent of scan area, phase fraction and number of precipitates. Values for the number of precipitates, cross section, length and phase fraction are presented in Table 4.2.

Table 4.2: The number, cross section area, length and phase fraction of the precipitates.

Scan	Phase	Number	Cross Section [nm^2]	Length [nm]	Phase fraction
S1	β''	674	22.8 ± 7.5	-	1
S2	β''	668	25.6 ± 9.1	-	1
S3	β''	703	26.2 ± 9.3	-	1
S4	β''	735	43.9 ± 20.4	-	1
S5	β''	924	46.3 ± 21.2	-	1
S6	β''	724	46.0 ± 18.2	-	0.82
	L	166	40.8 ± 10.8	-	0.18
S7	β''	173	41.9 ± 9.5	-	0.54
	L	159	43.3 ± 11.5	-	0.46
	Inplane	58	-	25.7 ± 15.0	-
S8	L	196	52.5 ± 16.6	1	
	Inplane	57	-	46.9 ± 33.6	-
S9	L	31	52.3 ± 35.2	-	0.48
	Q'	16	47.1 ± 12.9	-	0.21
	C	16	71.7 ± 48.2	-	0.31
	Inplane	31	-	56.7 ± 50.1	-
S10	Q'	10	85.7 ± 26.0	-	0.70
	C	6	107.6 ± 54.6	-	0.30
	Inplane	19	-	41.3 ± 30.9	-
S11	Q'	3	177.8 ± 35.1	-	1

Figure 4.11 shows the mean cross section area of the precipitates in each dataset. Next, Figure 4.12 shows the number of precipitates for each phases in every dataset. The total area of each precipitate as percent of total scan area follows in Figure 4.13. Next, Figure 4.14 shows the phase fractions of the phases in every dataset. Lastly, the average length of the inplane needles in dataset S7 to S10 are presented in figure 4.15.

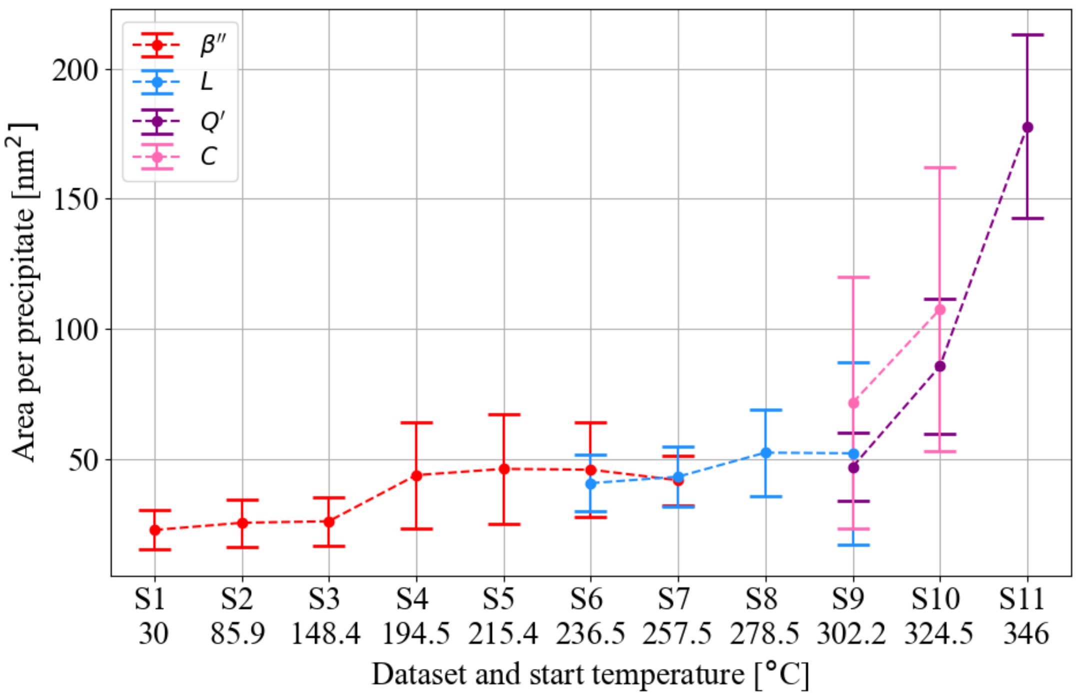


Figure 4.11: Area per precipitate in each scan. Start temperatures are also given for each scan.

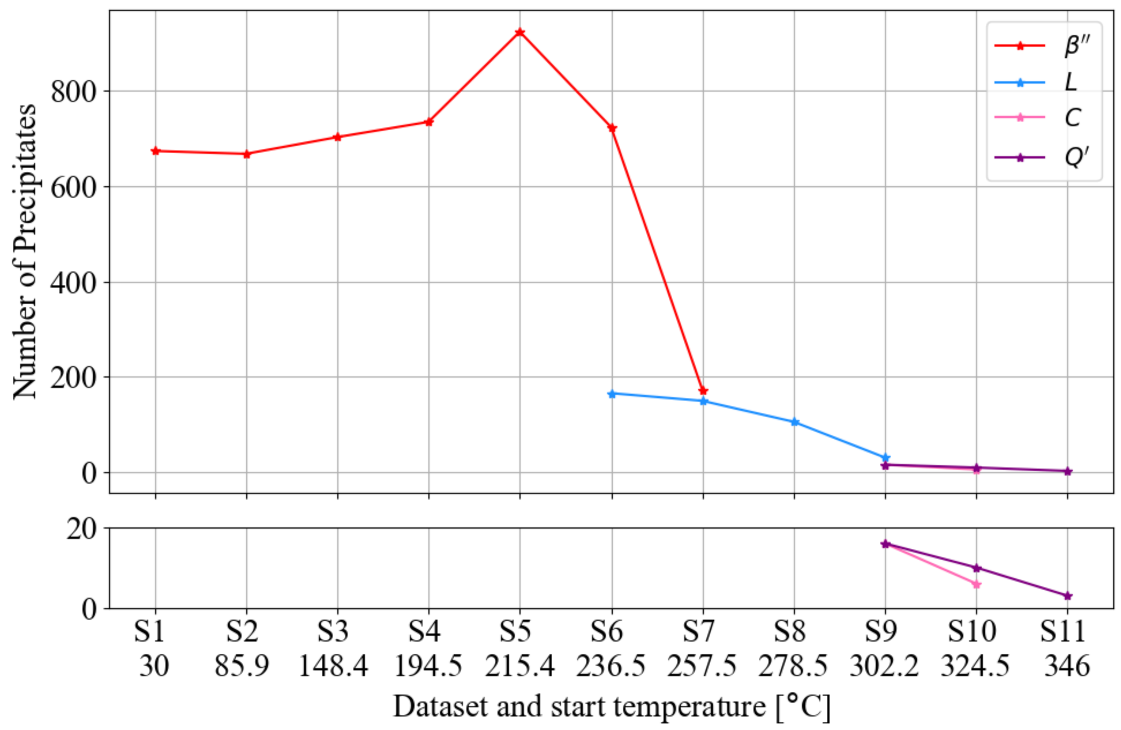


Figure 4.12: The number of each precipitate in each scan S1-S11, along with the starting temperature for the scans. The two axes show the same data, where the lowest axis is limited between 0 and 20 in order to show the difference between C and Q′.

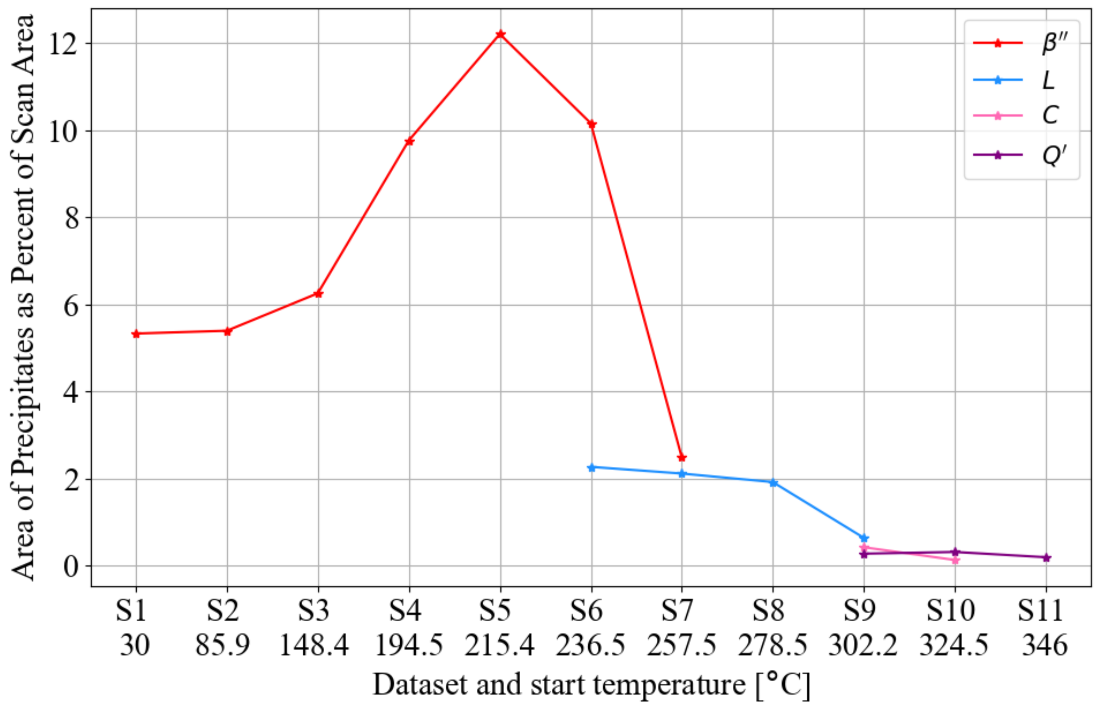


Figure 4.13: The total area of the each precipitate as percent of total area considered. The start temperatures are also given for each scan.

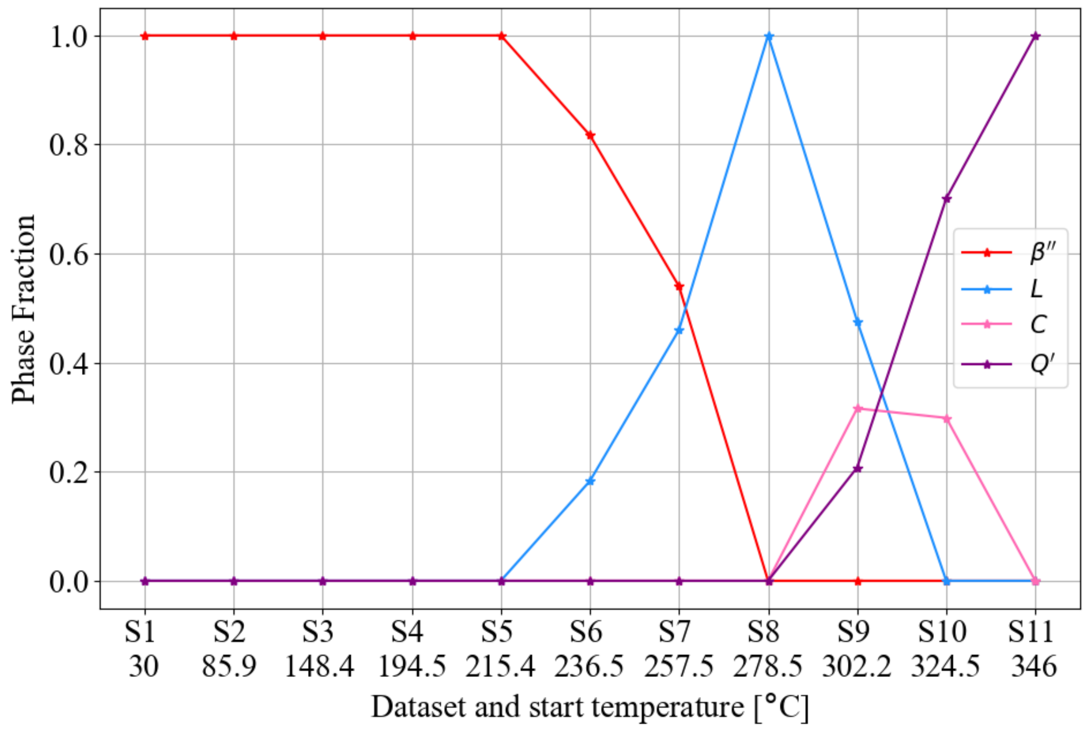


Figure 4.14: The phase fraction for each scan, along with starting temperatures. The phase fraction shows how large of a percentage each phase contributes to the total area of all precipitates.

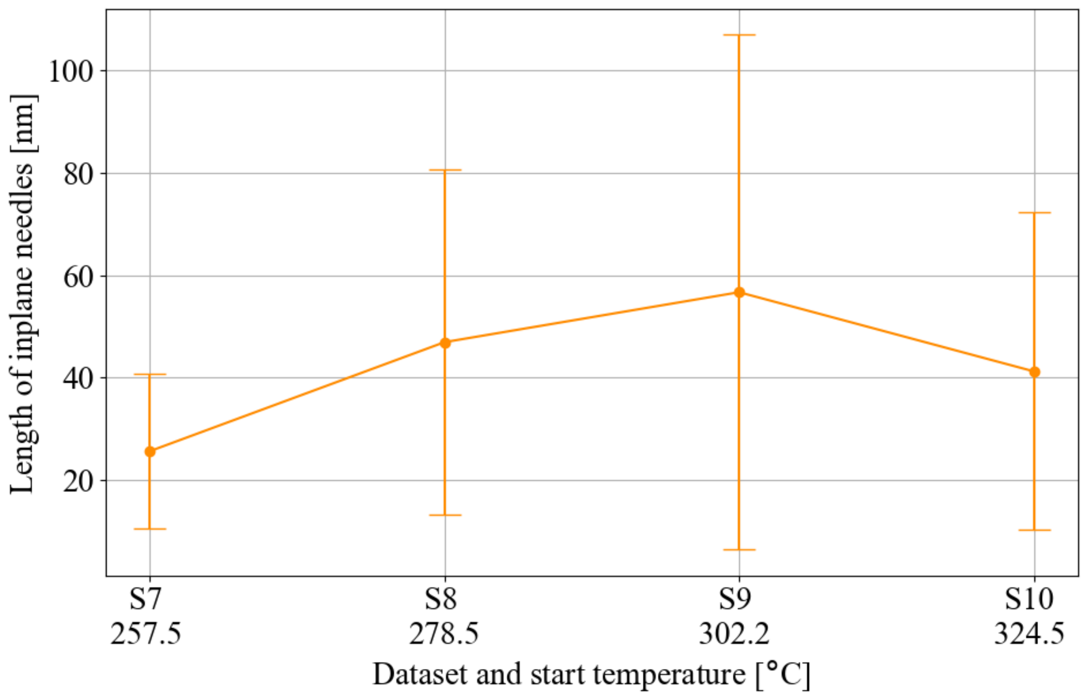


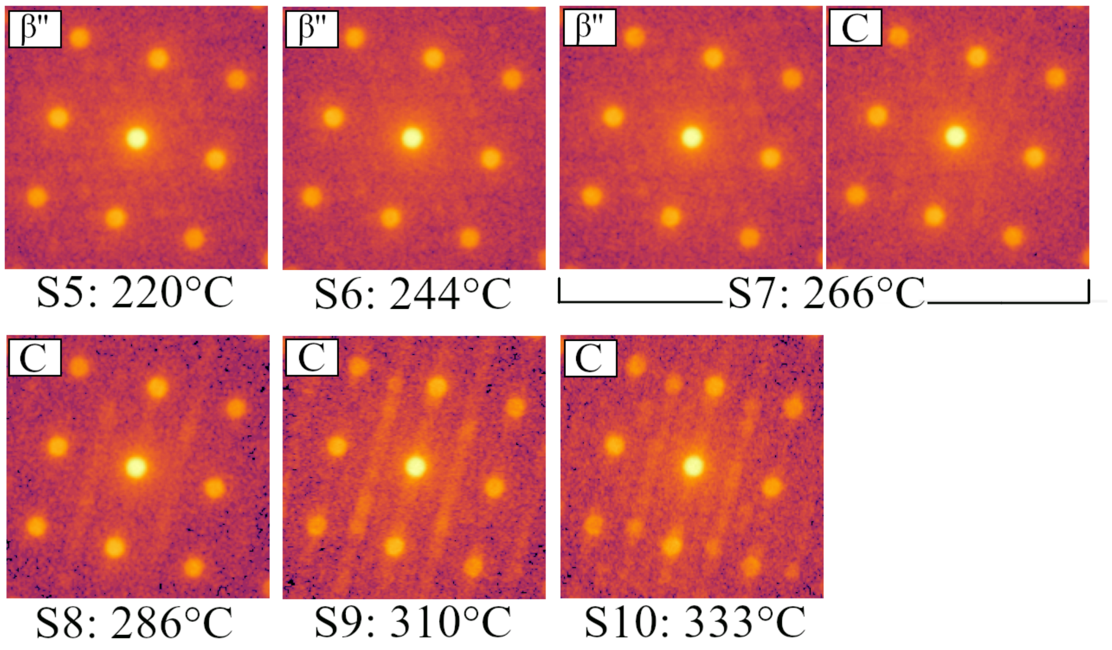
Figure 4.15: The length of the inplane needles in scan S7 to S10, along with start temperatures.

4.6 Phase Transformations

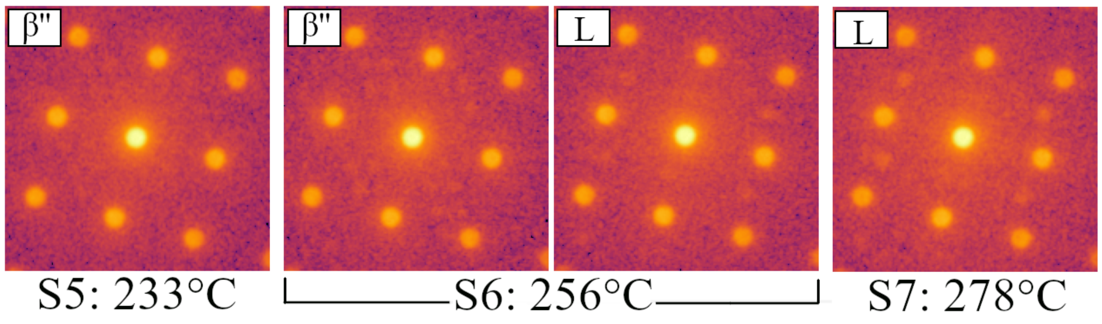
Several phase transformations are observed in the sample:

$$\begin{aligned} \beta'' &\rightarrow \mathrm{C} \\ \beta'' &\rightarrow \mathrm{L} \\ \mathrm{L} &\rightarrow \mathrm{Q}' \end{aligned} \tag{4.1}$$

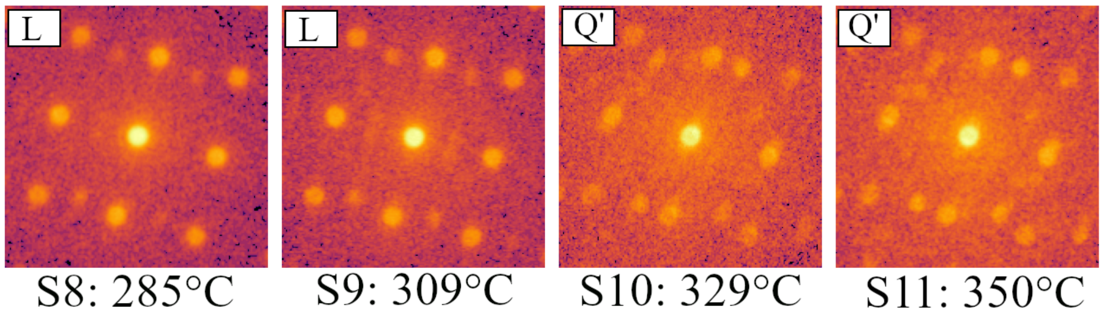
The phase transformations are shown in Figure 4.16(a) - 4.16(c). Where two DPs come from the same dataset, they are from neighbouring pixels. The temperatures shown are the temperatures on that row in the datasets. As seen, the transitions happen at approximately 266°C for β'' to C, 256°C for β'' to L, and between 309 and 329°C for L to Q′



(a) β″ to C



(b) β″ to L



(c) L to Q′

Figure 4.16: Phase transformations observed in the datasets. a) contains a transition from β″ to C, b) from β″ to L, and c) from L to Q′. Two DPs from the same scan indicates that these DPs are from neighbouring pixels in the dataset. The indicated temperatures are calculated for the rows in the dataset.

Chapter 5

Discussion

The discussion revisits all the results in the previous chapter. It is sectioned into two parts; a discussion of the data processing methods and steps taken in the production of the phase maps, and a discussion of the precipitate evolution results.

The former will include advantages and disadvantages of the method as a whole, and of the three data processing methods considered in Chapter 3. A discussion following the choice of NMF components is also included. The latter contains a discussion of the results following the evolution of precipitates, including precipitate statistics, phase maps and phase transformations.

5.1 Data Processing

This section discusses the steps in the data processing method. First, the determination of the number of components is discussed. Next, the choice of data processing method is discussed. Further, various aspects of each of the three methods are discussed, including disadvantages encountered.

5.1.1 Number of Components

Initially, 3 datasets were chosen to examine the number of components needed in the NMF decomposition. S3, S5 and S8 were chosen as they are approximately spaced equally among the datasets, and as S11 contained so few phases in the VDF. Only three were chosen as extensive investigation of number of components within all 11 dataset was outside the scope of this thesis. The goal was that if the results did not vary much depending on the number of components, the same number could be applied to the rest of the datasets.

The choice of the number of components was done on the basis of the three scans with three different numbers of components, see Table 4.1. The phase maps were created using the global threshold method, and the difference between them were found pixel by pixel. Table 4.1 shows the differences between the phase maps for the three different numbers of components. For S3, the NMF decomposition with 5 components contained no real diffraction patterns. Therefore, only the decompositions with 9 and 17 components are considered. The differences between the largest and second largest number of components is smallest for both S5 and S8. However, the probability that the remaining datasets only contain as few diffraction patterns as these three scans, and only one phase, are small. Therefore, the largest number of components was chosen for the remaining datasets; 19. However, this turned out to be too low of a number.

Inspecting the datasets manually, it is clear that the NMF has not yielded satisfying results. As seen in Figure 4.7, the phases appearing the the NMF decomposition do not reflect the real situation. In the decomposition, L starts appearing in S6, and its last appearance is in S9. However, this is not the case. The majority of the precipitates in S10 indexed as C are in fact L, while it also has been seen manually in S5. NMF decompositions on S1 and S10 with 50 components were done to examine this further. Now, L appears in S10, but not in S1. This means that if L is present earlier than S5, the diffraction patters must be very weak.

Additionally, C appears in S7 until S10 when investigating the raw data manually, while it is not present until S9 in the NMF decomposition. In the phase maps, a single C is indexed as L until S10. Some other Cs are visible in S9 as well. This shows that the number of components were not high enough.

As discussed, the NMF decomposition did not use enough components. Phases visible in several datasets when manually exploring did not show in every NMF decomposition. Additionally, a phase not present in any of the NMF decompositions were found during manual exploration. This phases appeared twice in S8. The angle between the Al DP and this phase is approximately 18°, which is similar to β'' (18.4°). However, no attempt to identify this phase has been done in this thesis.

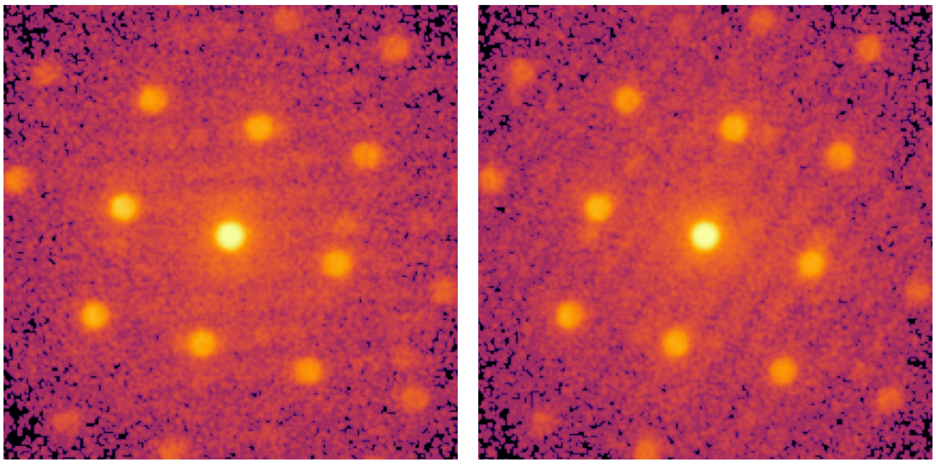

(a) (b)

Figure 5.1: An unidentified phase in S8.

5.1.2 Choice of Data Processing Method

The method chosen to proceed with to create the phase maps were as stated in the method the Gaussian method. There were several reasons for this decision. Figure 4.5 shows phase maps of β'' in S3 created using the three different methods. Starting with a comparison between Figure 4.5 a) and b), with the global and individual threshold methods, several precipitates appear in b) that do not appear in c). As it is desirable to obtain phase maps containing all precipitates, the individual threshold method has an advantage. However, in b), the precipitates appear larger than in a). Manually investigating the dataset reveals that the precipitates in b) are in fact too large. Additionally, the phase map created using the individual threshold method contains areas of a single or a few pixels around the edges, disconnected from the precipitate itself. This is a result of the threshold algorithm used. The edges of the precipitates and the noise in the background itself are of too equal intensities. This method therefore requires an extra step to remove this. Regardless of the apparently wrongly sized precipitates and this debris, the individual consideration of each precipitate in this method is ruled the most favourable. Identifying *all* precipitates is the priority.

Next, improvements to the individual threshold method was attempted. This was done by fitting a 2D Gaussian function to each precipitate before the thresholding. Now, two different methods for thresholding were tested. First, each Gaussian was cut off at its full width at half maximum (FWHM), only marking the areas defined by Equation 5.1 as True in the binary images.

$$\left(\frac{x-x_0}{\mathrm{FWHM}_x}\right)^2+\left(\frac{y-y_0}{\mathrm{FWHM}_y}\right)^2\leq 1 \tag{5.1}$$

However this proved to be unsuitable. A large proportion of the precipitates are very weak, resulting in broad peaks that stretch beyond the limit of the square masks. As a result, phase maps created with this method contained a considerable amount of completely square precipitates. These precipitates filled their squares completely.

As an alternative, the same threshold algorithm applied previously in the individual threshold method was used on the unlimited Gaussian functions. This proved to be an acceptable solution as the precipitates were now closer to their true size, without debris around the edges. Now, each square only contained the 2D Gaussian functions, resulting in cleaner edges around the precipitates. This method still conserved the number of precipitates in the same manner as the individual threshold method. Therefore, this method was applied to each dataset.

Still, drawbacks exist for this method as well. First, the fitting of Gaussians only yields reasonable results when the precipitates are round or elliptical in shape. This works well for out of plane orientations of β'' and L, but for Q′, this is not true. Comparing phase maps for S11 in Figure 4.10 d) with the VDF image in Figure 4.3 d) show that the shape of the precipitates are approximated to be far rounder in the phase maps. The same applies for the in plane needles. The needles were therefore processed using the individual threshold method instead.

Disadvantages of Masking Approach

A challenge in the thresholding of each masked square is overlap. This occurs when the precipitates are closer than the length of the sides of the square masks. If the overlap is small, and does not include the precipitate, the result is not affected. However, if the overlapping region contains a part of the precipitates, the result becomes affected. In the individual threshold method, the choice of threshold value will be changed. The basis used to find the threshold value now contains pixels from an entirely different precipitate, which may differ in intensity. Here, there are three cases to consider. Firstly, if the intensity is much greater in the overlapping region than in the precipitate itself, the likelihood that the precipitate will be entirely or partially removed, as it is has a too weak intensity compared to the overlapping region. Secondly, if the overlapping region has an intensity comparable to that of the precipitate, the overlap might also be kept after the threshold has been applied. The third case is if the precipitate has a much higher intensity than the overlap. In this case, the threshold value might be chosen to be lower than it would have been without the overlapping region, resulting in the precipitate appearing larger than it really is. The more overlap, the less accurate the threshold will be.

As the threshold is applied iteratively to each square, the overlap can result in parts of the precipitates being removed by the next iteration, as the inclusion of each new square in the binary image overwrites the last. However, by applying logical or to the new thresholded square and the same area in the entire binary image, this issue disappears.

However, the overlapping regions pose a different challenge when the Gaussian functions are fitted to the squares. Consider an area of a loading map containing many small precipitates next to each other. This can result in overlap between several squares and many overlapping regions within each square. The fitting of the Guassian will then take all these areas into account, resulting in a Gaussian function that is not located at position of the original precipitate position, is flatter and less distinct. Because of this effect, the resulting area in both the image containing the Guassians and the binary image might consist of fewer precipitates, or more diffuse areas. Still, the precipitates are more distinct with clearer borders than without the fitting of the Gaussians. An example from S5 is shown in Figure 5.2. Here, there is much overlap between masks in a). The resulting Gaussians in b) are fewer than the precipitates in a). One Gaussian is fitted to 3 of the original precipitates. Two other precipitates have also been merged with another precipaite. Thus, the resulting phase map in c) contains too few precipitates.

The choice of the square size can also affect the finished phase map. When the Gaussians are fitted to weak and indistinct precipitates, the Gaussians have a tendency to be broader. This results in less variety in the intensity, and the resulting threshold is chosen to allow a too large portion of the square to appear in the binary image. Thus, straight lines and borders can appear in the phase map, as the Gaussians are too large compared to the squares. Choosing larger squares minimises this problem significantly. However, increasing the square size also increases the chances of overlap, discussed above. Additionally, the smaller squares leaves less information to consider in the determination of the threshold value. The square size has

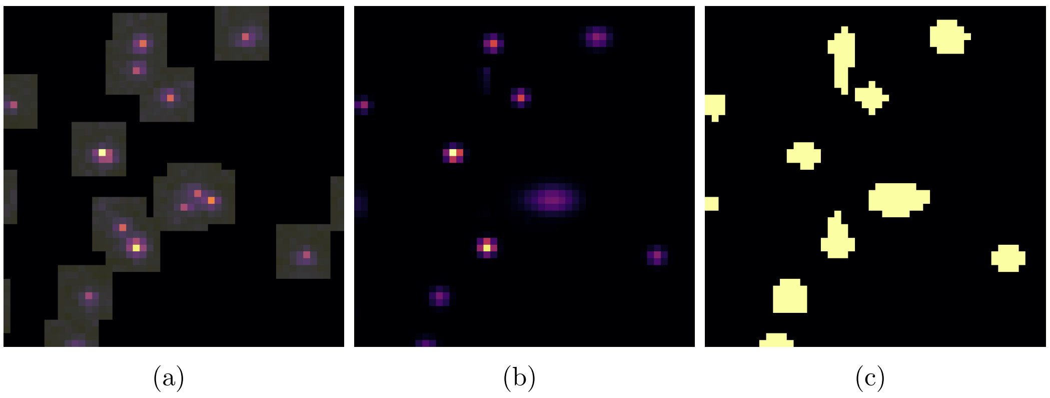


Figure 5.2: An example of overlap in S5. a) shows the constructed masks over the phases identified by peak finder. b) shows the same area after the Gaussian functions have been fitted to each square, revealing fewer precipitates. In c), the thresholdning function has been applied.

therefore been variable through all scans S1-S11, in order to get the least amount of overlap and straight edges, while maximising the information given to the threshold algorithm.

The size of the squares are fixed for each phase in this method. Customising the size of the masks to the actual size of the precipitates could improve the method. In the choice of the square size, the arguments for and against a certain size could all be disregarded. Using blob detection from scikit-image returns not only the position, but the radius as well, of found peaks [33]. Adjusting the size of the squares to a value slightly larger than the diameter could result in a more optimal size for all squares.

As seen in the method in Chapter 3, the intensity at the precipitate positions are doubled in the Gaussian method. This is done to increase the sharpness of the intensity, and thus give more narrow Gaussian functions. The benefit of a narrower Gaussian function is discussed above. However, there is a chance that this results in precipitates that are smaller in the phase maps than they really are. Manual investigation of the data using the phase maps have shown that this is not the case.

A weakness using peak finder, or any similar algorithm, is that features such as noise in the loading maps that are not precipitates can be picked up by the method. This gives precipitates in the loading map that do not exist. The chance that this happens increases when the precipitates are small and weak. Adjusting the parameters in the peak finder method can allow features that are not precipitates to get picked up, when precipitates are as weak as the intensity differences in background.

Precipitate Statistics

As presented in the results, several statistics from the precipitates have been extracted from the phase map creation. However, these results are very uncertain, and there is no guarantee that they are correct. Firstly, there exists no actual phase map for the scans; the only way of verifying the results are by manual inspection. Secondly, there is no correlation between the actual precipitates and the calculated threshold value. However, one aspect is certain; no precipitates are too small in the phase maps. This has been verified manually.

In the phase maps, there are large quantities of overlap and not indexed phases. However, the number of precipitates was retrieved from the peak finder method. Therefore, the numbers of precipitates for each phase can be too high, as two phases might completely overlap in the phase map. However, no other ways of counting the precipitates in the data processing proved to be more favourable. Overlap could be removed from each phase before they were counted, but this could lead too several areas of debris around removed overlap that could be counted as well, potenitally overestimating even more. Additionally, overlap could remove precipitates that actually are the same phase. The number of precipitates could also be incorrect as the peak finder method could identify areas wrongly as precipitates.

Next, the area per precipitate was found, considering each precipitate individually. As some precipitates slightly overlap or are connected in the phase maps, these were considered as one. This resulted in fewer precipitates, with some being two, three or more times larger than average. This affects the resulting areas by increasing both the average and standard deviation seen in Figure 4.11.

However, the results from the precipitate statistics are not without value. Even if the average in Figure 4.11 is inaccurate, the graph still clearly shows the growth of the precipitates. The varying size of the phases are also shown. Similarly, needle length in Figure 4.15 and total area in Figure 4.13 show growth as well.

5.2 Precipitate Evolution

As shown, Figures 4.1 - 4.3 present VDF images from each dataset. From S1 to S5, the VDF images look very similar. In S6, changes in the form of more distinct and clearer precipitates can be seen. No inplane needles can be seen yet. During S7, a dramatic change can be seen from S6 to S8. This change is clearly visible between the upper and lower parts of S7. Here, the temperature is around 267°. From the NMF decomposition, Figure, 4.7, it is clear that this stark contrast can in large part be attributed to β'' precipitates dissolving. This trend can also be seen in Figure 4.12, where the number of β'' starts to reduce after S5. During S7, the first inplane needles begin to become visible in the VDFs as well. From S7 to S9, one can clearly see the inplane needles grow. They grow in the [100] and [001] directions in Al. In S10 the needles are mostly gone. From S8 to S11 the cross section area of the precipitates have clearly increased.

Next, the simulated DPs and their corresponding components are shown in Figure 4.6. Each components fits well with the simulated DPs in term of symmetry and the placement of reflections. However, the intensities of the reflection vary greatly. For C in a), the diffraction spots that appear the weakest in the simulated pattern, appear intense in the component. The reflections also seem to bee smudged in one direction. For β'', the intensities for almost all reflections within the 220 reflections in Al seem intense. This is not the case in the simulation, were the reflections closest to the origin are the weakest. For Q', the six reflection constructing the hexagon indicated are the most intense in both the component and in the simulated DP. However, the rest of the reflections are of varying intensities in a pattern that do not match the intensities in the simulated DP. As the composition of the phase L various, it has not been simulated here. In order to identify L, its component has been compared to component and FFT in [34].

There can be several reasons for the differences in intensity. First, the components are not real DPs. The NMF algorithm does not know that these are DPs, and the method is purely mathematical. Therefore, there is no guarantee that the intensities are physically correct. Next, the simulations are simulated considering kinematic diffraction with excitation error. In a real sample, there is a chance of dynamical and double diffraction. This could also lead to differing intensities. However, both these considerations would still result in the positions of the reflections to be placed correctly, if the datasets were properly centered.

Phase Maps

The phase maps in Figure 4.8 - 4.10. They contain approximately the same information as the VDF images, but they do also show additional information. First, they also show beginning growth of inplane needles, and the apparent dissolution of the β'' in S7 in Figure4.9 c). Also the disappearance of a large quantity of L phases can be seen between S8 and s9 in Figure 4.10 a) and b). However, the phase maps also reveal more information. The VDF images of S5 and S6 in Figure 4.2 contain few distinguishable features. However, the phase maps in Figure 4.9 reveal that there are phases present in S6 that are not visible in S5. This applies to S7 as well, as there is no way of visually identifying that the precipitates appearing as spots are 2 different phases. Side by side, the phase maps reveal more information about the phases present than the VDF images.

However, the information in the phase maps is not always unambiguous. As soon as more than one phase is present in the phase maps, the green "not indexed" phase also appears. This makes some precipitates impossible to identify. Currently, there is no way to decide which precipitates are overlapping. Let there for example exist one phase in a phase map that, because of overlap, only becomes indexed as "not indexed". This phase would then completely disappear from the phase maps. This is not the case in the phase maps presented here. However, a solution to this could be to let the color of overlap and "not indexed" be decided by the actual phases overlapping. In the overlapping regions, the colour could be a mixture of the overlapping regions. However, many phases would yield this a solution with difficult to read phase maps. For now, the "not indexed" phase gives the clearest result. In order to minimize the amount of overlap between phases, phases could also be given priority over each other. This could be done by erasing overlapping phases between two phases from one of them, thus letting all overlap between two phases belong to one. However, this would require knowledge of the nature of each phase's diffraction pattern, and an effort to implement this has not been done here. Additionally, the amount of overlap in the phases maps presented could be excessive, as a consequence of the area determination discussed in the section above.

One difference between the phase maps and the VDF images are the inplane needles. At first in S7 in Figure 4.2 c) and Figure 4.9 c), the inplane needles are hardly visible and easily mistaken for the same as the surrounding spots in the VDF image, apart from at the very bottom. However, the phase map reveals that the needles have started to appear and grow in the very beginning of this scan. Overall, some needles are very weak in the VDF images, but several of them appear clearly in the VDF. Here, all needles are of much more equal thickness. However, not all needles appear in the phase maps. The intensities in the loading maps associated with these needles are on the verge of being too weak. Therefore, some needles are not being picked up by the data processing. The weakness of the needles are also why some of them appear with quite undefined edges. The weakness of the loading maps compared to the background results in the threshold values allowing some background to appear in order for the needle itself to be visible.

5.2.1 Phase Transitions

As shown i Figure 4.16, there are 3 phase transition are observed in the sample: β'' to L, β'' to C and L to Q'. As stated, DPs from the same dataset and temperature are from neighbouring pixels. The temperatures have been calculated based on the vertical positions of the pixels.

For β'' to C in Figure 4.16 a), both phases appear beside each other in S7. As seen in Figure 4.11, the average area of β'' decreases from S6 to S7, while the number of β'' decreases sharply in Figure 4.12. This could indicate that most of β'' have or is about to dissolve, as it is not present in S8. As β'' is dissolving, there could be an increase of alloying elements dissolved around the precipitates, which might increasing the probability of a new precipitate appearing in the same location, or area around. The time β'' uses to dissolve is long enough for new phases to appear.

Similarly for β'' to L in Figure 4.16 b), both phases exist in S6. Here, the average area is barely decreased from S5. However, judging by beginning decrease in the number of β'' S5 to S6, this β'' has also started to disappear. The resulting increase in the concentration of certain alloy elements could better the conditions for L to appear. Another possibility is that there is a continuous phase transformation happening. However, this can not be determined by these images. Lastly, for L to Q' in Figure 4.16 c), the phases do not appear together in any of the datasets. Because of this, it can not be determined how the transformation has happened.

The transformation from β'' to L was observed multiple times in the datasets, as shown here or as a direct transformation with only one phase appearing in the datasets at a time. The transformation fro L to Q' was also observed a couple of times. For β'' to C, there was only observed 1 case of this transformation. Additionally, every C precipitate was investigated, and the only transformation observed is the one presented here. As a result, there is not enough information to determine if this in fact is a phase transformation.

Another factor to discuss here is the thickness. Therefore, the precipitates could not be next to each other, but exist at two different z positions in the sample. From Figure 3.1, it is apparent that the thickness of the regions analysed are between approximately 115 and 80 nm. The needle lengths presented in Table 4.2 are for the most part shorter than this. However, some needles are clearly longer in S8 and S9 in Figure 4.3 a) and b). As a result, it is unlikely that this is the case for the transition from L to Q'. As the needles do not appear yet in the VDF images or the NMF decomposition in S6, it is hard to determine whether this is the case for β'' to L or β'' to C.

Additionally, the phase transformations can be difficult to locate as the DPs from the precipitates are often very weak. Increasing the dwell time could increase them, and make them easier to identify. Increasing the dwell time could have positive effects on the NMF as well, as clearer patterns would make the DPs more distinguishable.

Chapter 6

Conclusion

As presented, the NMF decompositions revealed four phases in the datasets from the Al-Mg-Si-Cu sample. These were β'', L, C and Q'. These were all orientated along the [001] Za in Al. Inplane needles in the [100] and [010] directions were also observed. However, these were not indexed. Additionally, three phase transformations were observed. These were β'' to C, β'' to L and L to Q'.

For the data processing, it became clear that the number of components in the NMF were too few in order to identify every phase present in the datasets. Even so, NMF works well as a method to separate DPs from different phases. The phase mapping with NMF also yields good results. The method appears to identify all the visible precipitates. However, the size of the precipitates in the phase maps do in some cases appear larger than they are. Lastly, a drawback of the NMF is the manual identification of the different phases.

One goal with this work was to investigate the phase transformation in order to identify the manner of each transformation, dissolution and nucleation or continuous phase transitions. As the transitions from β'' to L and β'' to L exhibited DPs from both phases in neighbouring pixels, these transitions could be continuous phase transitions. Alternatively, the dissolution of β'' and nucleation of C or L is happening rapidly. For L to Q', there was less information to draw from the transformation. However, for all three transformations, there exists too little data to identify the manner of these transformations.

Chapter 7

Further Work

Here, several suggestions to improvements to the method used is presented. Small alterations that could be implemented in the Gaussian method is presented first. Next, a decrease in the user input is discussed, and lastly, changes to the data acquisition is suggested.

First, using the radius of the individual precipitates instead of a fixed radius for the masks for each phase. This would, as discussed, reduce the work of the deciding one mask size for the entire phase. Additionally, as the size of the precipitates do depend on the mask size to some extent, a mask size depending on the radii could give a more constant error in the area. Most importantly, implementing this change would reduce the need for user input; no need to visually test and find an appropriate mask size. This could increase the objectiveness of the method.

Secondly, the size of each precipitate could be returned directly from each mask. Thus, the potential overlap and connections between precipitates in the final phase maps would not influence the area per precipitate. This would also remove the step of area of the individual precipitates calculation later on. Similarly, total area could also be returned at the same time. This alteration would also reduce the knowledge of the precipitates required prior to the phase mapping.

In order to decrease overlap between masks, circular masks could be used instead of square masks. Each circle could then be inserted into a square, and the method could be followed as usual. However, the areas around the circle in this square would have to be filled with approximately the same as the background in order for the Gaussian to be fitted to the precipitate, and not the circle.

There are also steps that could be taken to reduce the user input in the code. Currently, the code is in a Jupyter notebook. It also requires the user to run the code separately for each phase in a dataset. Parts of the code could be restructured as functions. The code after the peak finder algorithm could be combined to one function producing the phase maps and precipitate statistics without any further information than currently used in the notebook. However, the peak finder function, or possibly the blob finder function, could still require different variables depending on the precipitates' intensities or shape.

A drawback of the method still is that the labelling of the phases must be done manually. Further work is required to develop methods for automatic labelling combined with NMF. In the project thesis proceeding this master's thesis, an attempt was made to use template matching on the NMF components in order to identify the phase in each component. This is a method that could be explored further.

In the phase transitions, it was not clear how the phase transitions occurred; whether they dissolved and nucleated into a new phase, or if they underwent a continuous phase transition. In order to investigate this, it could be beneficial with a smaller scan area. This could result in shorter temperature intervals, and increase the number of DPs recorded from each position during the heating. More rapid data collection could therefore increase the probability of the manner of phase transitions becoming clearer. Additionally, a smaller step size could perhaps separate precipitates that are *not* actually next to each other. Then, the same data processing method as described in this thesis could be applied.

Bibliography

[1] Royal Society of Chemistry. *Aluminium.* 2023. URL: https://www.rsc.org/periodic-table/element/13/aluminium (visited on 1st Jan. 2023).

[2] Grønt Punkt Norge. *Metallemballasje.* URL: https://www.grontpunkt.no/gjenvinning/metallemballasje/ (visited on 1st Jan. 2023).

[3] International Aluminium Institute. *Report Reveals Global Aluminium Demand to Reach New Highs After Covid.* 2022. URL: https://international-aluminium.org/report-reveals-global-aluminium-demand-to-reach-new-highs-after-covid/ (visited on 16th June 2022).

[4] C. D. Marioara et al. 'The influence of alloy composition on precipitates of the Al-Mg-Si system'. In: *Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science* 36 (2005), pp. 691–702. DOI: 10.1007/s11661-005-1001-7.

[5] E. Thronsen et al. 'Scanning precession electron diffraction data analysis approaches for phase mapping of precipitates in aluminium alloys'. To be published.

[6] J. K. Sunde, S. Wenner and R. Holmestad. 'In situheating TEM observations of evolving nanoscale Al–Mg–Si–Cu precipitates'. In: *Journal of Microscopy* 279 (2019), pp. 143–147. DOI: 10.1111/jmi.12845.

[7] T. Bergh et al. 'Nanocrystal segmentation in scanning precession electron diffraction data'. In: *Journal of Microscopy* 279 (2019), pp. 158–167.

[8] P. C. Hemmer. *Faste stoffers fysikk.* 3rd ed. Fagbokforlaget, 2015.

[9] C. Kittel. *Introduction to Solid State Physics.* 8th ed. John Wiley and Sons, 2005.

[10] C. J. Humphreys. 'The Scattering of Fast Electrons by Crystals'. In: *Reports on Progress in Physics* 42 (1979), pp. 1825–1883.

[11] B. Fultz and J. Howe. *Transmission Electron Microscopy and Diffractometry of Materials.* 4th ed. Springer, 2013.

[12] P. A. Midgley and A. S. Eggeman. 'Precession electron diffraction - a topical review'. In: *IUCrJ* 2 (2015), pp. 126–136.

[13] E. Thronsen et al. 'The effect of heavy deformation on the precipitation in an Al-1.3Cu-1.0Mg-0.4Siwt.% alloy'. In: *Materials Design* 186 (2020). DOI: https://doi.org/10.1016/j.matdes.2019.108203..

[14] T. Bergh. ‘Electron microscopy of intermetallic phases in aluminium-steel joints’. PhD thesis. Norwegian University of Science and Technology, 2021.

[15] C. D. Marioara et al. ‘The effect of Cu on precipitation in Al–Mg–Si alloys’. In: *Philosophical Magazine* 87 (2007), pp. 3385–3413. DOI: https://doi.org/10.1080/14786430701287377.

[16] R. Vissers et al. ‘The crystal structure of the β' phase in Al–Mg–Si alloys’. In: *Acta Materialia* 55 (2007), pp. 3815–3823.

[17] E. Jacobsen. ‘Scanning Precession Electron Diffraction Template Matching for Automated Phase Mapping of Precipitates in 6xxx Aluminium Alloys’. MA thesis. Norwegian University of Science and Technology, 2020.

[18] S. Wenner et al. ‘Atomic-resolution chemical mapping of ordered precipitates in Al alloys using energy-dispersive X-ray spectroscopy’. In: *Micron* 96 (2017), pp. 103–111.

[19] S. Wenner et al. ‘Atomic-resolution chemical mapping of ordered precipitates in Al alloys using energy-dispersive X-ray spectroscopy’. In: *Micron* 96 (2017), pp. 103–111. DOI: https://doi.org/10.1016/j.micron.2017.02.007.

[20] L. Ding et al. ‘The morphology and orientation relationship variations of Q phase in Al–Mg–Si–Cu alloy’. In: *Materials Characterization* 118 (2016), pp. 297–283.

[21] S. Andersen et al. ‘Crystal structures of the trigonal U1-MgAl2Si2 and orthorhombic U2-Mg4Al4Si4 precipitates in the al-mg-si alloy system’. In: *Electron Microscopy and Analysis 2003* (Jan. 2004), pp. 225–228.

[22] F. de la Peña et al. *hyperspy/hyperspy: Release v1.7.0.* Version v1.7.0. Apr. 2022. DOI: 10.5281/zenodo.6492919.

[23] N. Cautaerts et al. ‘Free, flexible and fast: Orientation mapping using the multi-core and GPU-accelerated template matching capabilities in the python-based open source 4D-STEM analysis toolbox Pyxem’. In: *Ultramicroscopy* (2022), p. 113517. DOI: 10.1016/j.ultramic.2022.113517.

[24] I. MacLaren et al. ‘A Comparison of a Direct Electron Detector and a High-Speed Video Camera for a Scanning Precession Electron Diffraction Phase and Orientation Mapping’. In: *Microscopy and Microanalysis* 26 (2020), pp. 1110–1116.

[25] A. Kumar. *PCA Explained Variance Concepts with Python Example.* URL: https://vitalflux.com/pca-explained-variance-concept-python-example/ (visited on 15th Jan. 2023).

[26] I. T. Jolliffe and J. Cadima. ‘Principal component analysis: a review and recent developments’. In: *Philosophical Transactions of the Royal Society A* 374 (2016).

[27] Y. Wang and Y. Zhang. ‘Nonnegative Matrix Factorization: A Comprehensive Review’. In: *IEEE Transactions on Knowledge and Data Engineering* 25 (2013), pp. 1336–1353.

[28] A. Laub V. Klema. ‘The singular value decomposition: Its computation and some applications’. In: *IEEE Transactions on Automatic Control* (1980), pp. 164–176. DOI: 10.1109/TAC.1980.1102314.

[29] M. Själander et al. *EPIC: An Energy-Efficient, High-Performance GPGPU Computing Research Infrastructure*. 2019. arXiv: 1912.05848 [cs.DC].

[30] S. van der Walt et al. 'scikit-image: image processing in Python'. In: *PeerJ* (2014). DOI: 10.7717/peerj.453.

[31] Y. Seto and M. Ohtsuka. 'ReciPro: free and open-source multipurpose crystallographic software integrating a crystal model database and viewer, diffraction and microscopy simulators, and diffraction data analysis tools'. In: *Journal of Applied Crystallography* 55 (2022). DOI: 10.1107/S1600576722000139..

[32] W. Yang et al. 'The diffraction patterns from β'' precipitates in 12 orientations in Al–Mg–Si alloy'. In: *Sctripta Materiala* 62 (2010), pp. 705–708.

[33] Scikit-image development team. *Blob Detection*. URL: https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_blob.html (visited on 17th Jan. 2023).

[34] J. K. Sunde et al. 'The evolution of precipitate crystal structures in an Al-Mg-Si(-Cu) alloy studied by a combined HAADF-STEM and SPED approach'. In: *Materials Characterization* 142 (2018), pp. 458–469.

Appendix A

Code

Here, the code for the processing of the NMF components are presented.

```
[ ]: import hyperspy.api as hs
     import pyxem as pxm
     import numpy as np
     import matplotlib.pyplot as plt
     from pathlib import Path
     from matplotlib.colors import SymLogNorm

     import skimage
     from skimage.filters import threshold_local, threshold_isodata, threshold_otsu,␣
      ↪threshold_yen, try_all_threshold

     from matplotlib.colors import to_rgba
     from matplotlib.colors import LinearSegmentedColormap

     %matplotlib qt
```

0.0.1 1. Load loadings og factors

```
[ ]: datapath_factors = Path('Factor_T3_17.hspy')
     datapath_loadings = Path('Loading_T3_17.hspy')

     factors = hs.load(datapath_factors)
     loadings = hs.load(datapath_loadings)

     datapath=Path('T3_17.hspy')
```

0.0.2 2. Combine loadings for the same phases

```
[ ]: loadings.plot(cmap = 'inferno_r')
```

```
[ ]: loadings.data[3] += loadings.data[7]
```

Create dictionary over phases

```
[ ]: #3
     phase_key = {
         'Al': [],
         'beta``': [3],
         'Not indexed': [1,2,4,5,6,8,9,10,11,12,13,14,15,16]
     }
```

0.0.3 3. Remove background

```
[ ]: #Component which contain all components for the phase
     component_nr = 3
```

```
[ ]: from scipy.ndimage import gaussian_filter
     from skimage.morphology import reconstruction
```

```
[ ]: original = loadings.data[component_nr] # Save original for comparison
```

```
[ ]: threshold = 3000 # Threshold for background subtraction
     sigma = 0.1
     img_temp = gaussian_filter(loadings.data[component_nr], sigma)
     mask = img_temp
     seed = img_temp - threshold
     dilated = reconstruction(seed, mask, method='dilation')
     background_removed = img_temp - dilated

     fig, ax = plt.subplots(1,2)
     ax[0].imshow(original, cmap = 'inferno_r')
     ax[0].set_title("Original")
     ax[1].imshow(hdome, cmap = 'inferno_r')
     ax[1].set_title("Background removed")
     plt.show()
```

```
[ ]: loadings.data[component_nr] = background_removed
```

0.0.4 4. Peak finder

```
[ ]: loadings.set_signal_type("electron_diffraction")
```

```
[ ]: #Find suitable value for min_sigma
     min_sigma = 0.44
     max_sigma = 10
     threshold = 0.1

     peaks = loadings.inav[component_nr].find_peaks(interactive= True,
                                                    method = "laplacian_of_gaussian",
                                                    min_sigma = min_sigma,
                                                    max_sigma = max_sigma,
                                                    threshold = threshold)
```

4.1 Note number of precipitates Substract precipitates which are identified more than once.

```
[ ]: print("Number of precipitates: ", len(peaks.data))
```

0.0.5 5.1 Mask and threshold for out-of-plane precipitates

```
[ ]: image = np.copy(loadings.inav[component_nr].data)
```

```
[ ]: plt.imshow(image)
```

```
[ ]: import lmfit
     from lmfit.lineshapes import gaussian2d, lorentzian
```

```
#Find suitable radius for masks
r = 3

mask = np.full((loadings.axes_manager[2].size,loadings.axes_manager[1].size),␣
 ↪False)
X, Y = np.meshgrid(np.arange(0, loadings.axes_manager[1].size), np.arange(0,␣
 ↪loadings.axes_manager[2].size))
cx, cy = np.array((loadings.axes_manager[2].size, loadings.axes_manager[1].
 ↪size))//2
coords = np.zeros((len(peaks.data), 4))

i=0
for peak in peaks.data:

    #Extract information from peak finder, and assign coordinates to the mask
    cx, cy = peak
    x1 = cx-r
    x2 = cx+r
    y1 = cy-r
    y2 = cy+r
    coords[i] = [x1, x2, y1, y2]
    i+=1

    #Double the intesity at peak position
    image[cx, cy] = 2*image[cx, cy]

    #Create square mask
    mask[int(x1):int(x2), int(y1):int(y2)] = True

#Create masked image, image to store Gaussian functions, and binary image
temp = image*mask
gauss = np.zeros((loadings.axes_manager[2].size,loadings.axes_manager[1].size))
bin_img = np.full((loadings.axes_manager[2].size,loadings.axes_manager[1].size),␣
 ↪False)

for peak, coord in zip(peaks.data, coords):

    cy, cx = peak
    x1, x2, y1, y2 = coord

    if temp[int(x1):int(x2), int(y1):int(y2)].shape[1] > 0 and temp[int(x1):
 ↪int(x2), int(y1):int(y2)].shape[0] > 0:

        #Create coordinate grid
        x = np.arange(temp[int(x1):int(x2), int(y1):int(y2)].shape[1])
        y = np.arange(temp[int(x1):int(x2), int(y1):int(y2)].shape[0])
        X,Y = np.meshgrid(x,y)
        #Find best fitted Gaussian
        model = lmfit.models.Gaussian2dModel()
        params = model.guess(temp[int(x1):int(x2), int(y1):int(y2)].ravel(), X.
↪ravel(), Y.ravel())
        result = model.fit(temp[int(x1):int(x2), int(y1):int(y2)].ravel(), x=X.
↪ravel(), y=Y.ravel(), params=params)

        gauss[int(x1):int(x2), int(y1):int(y2)] = model.func(X, Y, **result.
↪best_values)


        #Find threshold value for Gaussian
        threshold = threshold_yen(gauss[int(x1):int(x2), int(y1):int(y2)])
        binary = gauss[int(x1):int(x2), int(y1):int(y2)] > threshold

        #Insert thresholded precipitate into binary image
        bin_img[int(x1):int(x2), int(y1):int(y2)] = np.
↪logical_or(bin_img[int(x1):int(x2), int(y1):int(y2)], binary)


#Plot results
fig, ax = plt.subplots(1,5, figsize=(15, 9))
ax[0].imshow(mask, cmap = 'inferno')
ax[0].set_title("A")
ax[1].imshow(loadings.inav[component_nr].data, cmap = 'inferno')
ax[1].set_title("B")
ax[2].imshow(temp, cmap = 'inferno')
ax[2].set_title("C")
ax[3].imshow(gauss, cmap = 'inferno')
ax[3].set_title('D')
ax[4].imshow(bin_img, cmap = 'inferno')
ax[4].set_title("E")
#plt.savefig(f'{datapath.stem}_precipitate.png', bbox_inches='tight')
plt.show()
```

[ ]:
```
#If satisfied, save binary image over combined loading map
loadings.inav[component_nr] = imgg
```

Zoom in on selected areas to check results

[ ]:
```
fig, ax = plt.subplots(1,5, figsize=(15, 9))
ax[0].imshow(mask[100:200,100:200], cmap = 'inferno')
ax[0].set_title("A")
ax[1].imshow(image[100:200,100:200], cmap = 'inferno')
ax[1].set_title("B")
ax[2].imshow(temp[100:200,100:200], cmap = 'inferno')
ax[2].set_title("C")
```

```
ax[3].imshow(gauss[100:200,100:200], cmap = 'inferno')
ax[3].set_title('D')
ax[4].imshow(bin_img[100:200,100:200], cmap = 'inferno')
ax[4].set_title("E")
plt.show()
```

0.0.6 5.2 Mask and threhold for inplane needles

[ ]:
```
image = np.copy(loadings.inav[component_nr].data)
```

[ ]:
```
plt.imshow(image)
```

[ ]:
```
#Find suitable radius for masks
r = 3

mask = np.full((loadings.axes_manager[2].size,loadings.axes_manager[1].size),␣
 ↪False)
X, Y = np.meshgrid(np.arange(0, loadings.axes_manager[1].size), np.arange(0,␣
 ↪loadings.axes_manager[2].size))
cx, cy = np.array((loadings.axes_manager[2].size, loadings.axes_manager[1].
 ↪size))//2
coords = np.zeros((len(peaks.data), 4))

i=0
for peak in peaks.data:

    #Extract information from peak finder, and assign coordinates to the mask
    cx, cy = peak
    x1 = cx-r
    x2 = cx+r
    y1 = cy-r
    y2 = cy+r
    coords[i] = [x1, x2, y1, y2]
    i+=1

    ##Create square mask
    mask[int(x1):int(x2), int(y1):int(y2)] = True

temp = image*mask
bin_img = np.full((loadings.axes_manager[2].size,loadings.axes_manager[1].size),␣
 ↪False)

for peak, coord in zip(peaks.data, coords):

    cy, cx = peak
    x1, x2, y1, y2 = coord
    #Find threshold value for the square
    threshold = threshold_yen(temp[int(x1):int(x2), int(y1):int(y2)])
    binary = temp[int(x1):int(x2), int(y1):int(y2)] > threshold

    #Insert thresholded precipitate into binary image
    #bin_img[int(x1):int(x2), int(y1):int(y2)] = np.logical_or(bin_img[int(x1):
 ↪int(x2), int(y1):int(y2)], binary)
    bin_img[int(x1):int(x2), int(y1):int(y2)] =  binary

#Plot results
fig, ax = plt.subplots(1,4, figsize=(15, 9))
ax[0].imshow(mask, cmap = 'inferno')
ax[0].set_title("A")
ax[1].imshow(image, cmap = 'inferno')
ax[1].set_title("B")
ax[2].imshow(temp, cmap = 'inferno')
ax[2].set_title("C")
ax[3].imshow(imgg, cmap = 'inferno')
ax[3].set_title("D")
#plt.savefig(f'{datapath.stem}_inplane.png', bbox_inches='tight')
plt.show()
```

```
[ ]: #If satisfied, save binary image over combined loading map
     loadings.inav[component_nr] = bin_img
```

0.0.7 5.2.1 Remove debris around needles

```
[ ]: img = loadings.inav[component_nr].data
```

```
[ ]: areas = skimage.morphology.label(img, background=None, return_num=False,␣
      ↪connectivity=1)
     props = skimage.measure.regionprops(areas)
```

```
[ ]: props[:] = [prop for prop in props if prop['area'] > 11] #Choose suitable area
```

```
[ ]: props[:] = [prop for prop in props if prop['eccentricity'] > 0.4] #Choose␣
      ↪suitable eccentricity
```

```
[ ]: props[:] = [prop for prop in props if prop['extent'] < 0.5] #Choose suitable␣
      ↪extent
```

```
[ ]: #Add all areas together
     debris_removed = np.zeros((loadings.axes_manager[2].size, loadings.
      ↪axes_manager[1].size))
     for prop in props:
         np.add.at(debris_removed, tuple(zip(*prop['coords'])), 1)
         np.rot90(debris_removed)
```

```
#Plot results
fig, ax = plt.subplots(1,3)
ax[0].imshow(img)
ax[0].set_title('Before debris removal')
ax[1].imshow(res)
ax[1].set_title('After debris removal')
plt.show()
```

```
[ ]: #If satisfied, save result over corresponding component
loadings.inav[component_nr] = res
```

```
[ ]: #Count the needles
print("Number of inplane needles: ", len(props))
```

0.0.8 6. Calculate area of precipitates

```
[ ]: area = np.sum(loadings.inav[component_nr].data)
area_percent = 100*area/(loadings.axes_manager[2].size*loadings.axes_manager[1].
 ↪size)
print("Total area: ", area)
print("Area precipitates/scan area: ", area_percent)
print("Scan area: ", loadings.axes_manager[2].size*loadings.axes_manager[1].size)
```

0.0.9 7. Repeat process for every phase in the dataset

0.0.10 8.1 Create phase map

```
[ ]: #Create individual phase maps for each phase in the dictionary
phase_map = np.zeros((loadings.axes_manager[2].size, loadings.axes_manager[1].
 ↪size) + (len(phase_key),))
phase_patterns = np.zeros(factors.axes_manager.signal_shape)
ax_size = 3 #inches
fig, axes = plt.subplots(nrows=1, ncols=len(phase_key),␣
 ↪figsize=(ax_size*len(phase_key), ax_size), subplot_kw={'xticks': [], 'yticks':␣
 ↪[]})
for phase_id, (phase, ax) in enumerate(zip(phase_key, axes)):
    if len(phase_key[phase]) > 0:
        for component in phase_key[phase]:
            phase_map[:, :, phase_id] += loadings.inav[component].data/np.
 ↪nanmax(loadings.inav[component].data)

        phase_map[:, :, phase_id] /= np.max(phase_map[:, :, phase_id])
        image = phase_map[:, :, phase_id]
        threshold = threshold_yen(image)
        binary = image > threshold
        phase_map[:, :, phase_id] = binary
    ax.imshow(phase_map[:, :, phase_id], 'inferno')
    ax.set_title(f'{phase} map')
```

0.0.11 8.2 Move overlap to "not indexed"

[ ]:
```
not_indexed_id = list(phase_key.keys()).index('Not indexed')
phase_map[:, :, not_indexed_id] = np.sum(phase_map[:, :, :not_indexed_id],␣
 ↪axis=2)>1
fig, axes = plt.subplots(nrows=1, ncols=len(phase_key),␣
 ↪figsize=(ax_size*len(phase_key), ax_size), subplot_kw={'xticks': [], 'yticks':␣
 ↪[]})
for phase_id, (phase, ax) in enumerate(zip(phase_key, axes)):
    if phase_id != not_indexed_id:
        phase_map[:, :, phase_id][phase_map[:, :, not_indexed_id]>0] = 0
    ax.imshow(phase_map[:, :, phase_id], 'inferno')
    ax.set_title(f'{phase} map')
```

0.0.12 8.3 Set everything else to Al

[ ]:
```
Al_id = 0
phase_map[:, :, Al_id] = ~np.any(phase_map[:, :, Al_id:], axis=2)
fig, axes = plt.subplots(nrows=1, ncols=len(phase_key),␣
 ↪figsize=(ax_size*len(phase_key), ax_size), subplot_kw={'xticks': [], 'yticks':␣
 ↪[]})
for phase_id, (phase, ax) in enumerate(zip(phase_key, axes)):
    ax.imshow(phase_map[:, :, phase_id], 'inferno')
    ax.set_title(f'{phase} map')
```

0.0.13 8.4 Assemble phase map

Color names = 'linen', 'darkorange', 'dodgerblue', 'forestgreen', 'red', 'purple', 'hotpink'

[ ]:
```
#Find which colors and phases to use
print(phase_key)
```

[ ]:
```
#Create colormap from the colors
color_names = ['linen', 'purple']
colors = [to_rgba(c) for c in color_names]

cmap = LinearSegmentedColormap.from_list('gt_cmap', colors, N=len(color_names))
```

[ ]:
```
#Create phase map
from mpl_toolkits.axes_grid1 import make_axes_locatable

phase_image = np.zeros(phase_map.shape[:2])
for phase_id, phase in enumerate(phase_key):
    phase_image += phase_id*phase_map[:, :, phase_id]
fig = plt.figure(dpi=1200, frameon=False)
ax = fig.add_axes([0, 0, 1, 1], xticks=[], yticks=[], frameon=False)
im = ax.imshow(phase_image, cmap=cmap)

#fig.savefig(f'{datapath.stem}_phase_map_gauss.png', bbox_inches='tight')
```

0.0.14 9. Save files

[ ]:
```
#Save loadings including the individual phase maps
loadings.save(f'{datapath.stem}_klar.hspy')
```

[ ]:
```
#Save phase map as txt
np.savetxt(f'{datapath.stem}_phase_map.txt', phase_image)
```

0.0.15 A. Phase dictionaries

[ ]:
```
#Phase dictionaries indicating which components have been added

#T1
phase_key = {
    'Al': [],
    #'L' : [],
    #'Inplane' : [],
    'beta``': [3], #7
    'Not indexed': [1,2,4,5,6,8,9,10,11,12,13,14,15,16,17,18],
}

#T2
phase_key = {
    'Al': [],
    #'L' : [],
    #'Inplane' : [],
    'beta``': [3], #8
    'Not indexed': [1,2,4,5,6,7,9,10,11,12,13,14,15,16,17,18],
}

#3
phase_key = {
    'Al': [],
    #'L' : [],
    #'Inplane' : [],
    'beta``': [3], #7
    'Not indexed': [1,2,4,5,6,8,9,10,11,12,13,14,15,16]
}

#T4
```

```
phase_key = {
    'Al': [],
    #'L' : [],
    #'Inplane' : [],
    'beta``': [3], #7
    'Not indexed': [1,2,4,5,6,8,9,10,11,12,13,14,15,16,17,18],
}

#T5
phase_key = {
    'Al': [],
    #'L' : [],
    #'Inplane' : [],
    'beta``': [5], #,7,8,10
    'Not indexed': [1,2,3,4,6,9,11,12,13,14,15,16,17,18]
}

#T6
phase_key = {
    'Al': [],
    'L' : [11],
    'Inplane' : [],
    'beta``': [1], #,5,7,9
    'Not indexed': [2,3,4,6,8,10,12,13,14,15,16,17,18],
}

#T7
phase_key = {
    'Al': [],
    'L' : [6], #,8
    'Inplane' : [11],
    'beta``': [7],
    'Not indexed': [1,2,3,4,5,9,10,12,13,14,15,16,17,18],
}

#T8
phase_key = {
    'Al': [],
    'L' : [1], #,5
    'Inplane' : [6],
    #'beta``': [1],
    'Not indexed': [2,3,4,7,8,9,10,11,12,13,14,15,16,17,18], #8,9- og fremover␣
 ↪svak?
}

#T9
phase_key = {
    'Al': [],
    'L' : [9],
    'Inplane' : [7],
    'C' : [4],
    'Q' : [1],
    'Not indexed': [2,3,5,6,8,10,11,12,13,14,15,16,17,18],
}

#T10
phase_key = {
    'Al': [],
    'C' : [7],
    'Inplane' : [6],
    'Q' : [1],# ,5,8
    'Not indexed': [2,3,4,9,10,11,12,13,14,15,16,17,18],
}

#T11
phase_key = {
    'Al' : [],
    'Q' : [1],#,5,6,
    'Not indexed': [2,3,4,7,8,9,10,11,12,13,14,15,16,17,18],
}
```