Eirik Opheim

Evaluating template matching for orientation analysis based on electron diffraction

Master's thesis in MTFYMA
Supervisor: Prof. Antonius T. J. van Helvoort
June 2020

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

Eirik Opheim

Evaluating template matching for orientation analysis based on electron diffraction

Master's thesis in MTFYMA
Supervisor: Prof. Antonius T. J. van Helvoort
June 2020

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

Summary

To relate structural features to properties, phase and orientation mapping is widely used to get information on for instance grain size distributions, crystallographic textures and possible orientation relationships. In transmission electron microscopy, orientation mapping can be achieved with nanometer spatial resolution by performing scanning electron diffraction, where a two-dimensional diffraction pattern is recorded at each pixel in a two-dimensional scan region. Subsequent data processing and analysis is required to obtain orientation maps, which is most commonly achieved by model based template matching. In template matching, each experimental diffraction pattern is compared to a template bank consisting of simulated patterns for all possible candidate phases and orientations.

This project improves the electron diffraction-based template matching procedure in the open-source Python library pyXem, by adding two new methods for correlating experimental diffraction patterns to simulated candidate patterns; Zero-normalized cross-correlation (ZNCC) and full-frame correlation (FFC). The two methods and the established normalized cross-correlation (NCC) method available in pyXem were evaluated and compared. This was done by running the template matching procedure on simulated and experimental data from GaSb and mordenite, using different test set ups such as tilt series. This work shows that all three methods are able to distinguish between signals from different orientations for the ideal case where every signal has a perfect fit in the template bank. When the simulated orientations were taken from a grid of constant angular spacing on the fundamental zone, the NCC method provided the best results for both GaSb and mordenite. The NCC method is also the fastest method, and requires less memory than FFC. For experimental data, all the methods failed for the parameters used for generating the template library.

For this comparative study, a new metric for evaluating the reliability of a template matching result, labeled weighted reliability index (WRI), was proposed. WRI provided a more accurate evaluation of the orientation result than the established reliability index for all test cases of simulated GaSb and mordenite in this work. To ensure the correctness of the methods applied in this work, key functionalities of the template matching procedure in pyXem were investigated, including generation of a list of candidate orientations in the fundamental zone and kinematic simulation of diffraction patterns. It was found that the functions for generating a list of candidate orientations demonstrated unsatisfactory performance for cubic structures, because the fundamental zone is not fully covered. Consequently, the current version of pyXem must be further improved with regards to template matching by correcting issues as complete covering pointed out in this work.

Future research should focus on improving the template matching procedure, in first instances to tilt series, by improving the method for simulation of diffraction patterns. While the new methods implemented here did not improve template matching for the test cases, it might be better for precessed tilt series and data acquired with better detector technology (ie Direct detection).

Sammendrag

For å relatere strukturelle trekk ved et materiale til dets eganskaper er fase- og orientasjonskart mye brukt, for å få informasjon om fordelingen av korn, krystallografisk tekstur og mulige sammenhenger mellom orientasjoner. I transmisjonselektronmikroskopi kan orientasjonskartlegging gjøres med en romlig oppløsning på nanometerskala ved å utføre en elektrondiffraksjonsskanning, der et todimensjonalt diffraksjonsmønster blir registrert for hver pixel i et todimensjonalt testområde på prøven. Påfølgende dataprossesering og analyse er påkrevd for å kunne lage orientasjonskart. Dette er vanligvis gjort ved hjelp av modellbasert mønstersammenligning. I en mønstersammenligning blir hvert ekspri-mentelle diffraksjonsmønster sammenlignet med en bank av simulerte mønstre for alle mulige kandidatfaser og orientasjoner.

I dette prosjektet forbedres den elektrondiffraksjonbaserte mønstersammenligningen i det åpent tilgjengelige Python biblioteket pyXem, ved å legge til to nye metoder for sammenligning av eksperimentelle differasjonsmønstre til simulerte kandidatmønstre: Null-normalisert krysskorrelasjon (NNKK) og helbilde korrelasjon (HBK). De to metodene og den veletablerte normalisert krysskorrelasjon (NKK), som er tilgjengelig i pyXem ble evaluert og sammenlignet. Dette ble gjort ved å kjøre mønstersammenligningsprosedyren på simulerte og eksperimentelle data fra GaSb og mordenite, ved å bruke forskjellige testsituasjoner, som blant annet en rotasjonsserie. Dette arbeidet viser at alle de tre metodene kan skille mellom mønstre fra forskjellige orientasjoner for den ideele situasjonen der hvert mønster har en perfekt match i banken av simulerte mønstre. Når de simulerte orienteringene blir tatt fra et nett med en grads mellomrom i det symmetrireduserte orientasjonsrommet er det NKK metoden som gir de beste resultatene for både GaSb og mordenite. NKK metoden er også den raskeste og krever mindre minne enn HBK. For eksperimentelle data feilet alle metodene for parameterene som ble brukt for å generere en bank av simulerte mønstre.

I dette sammnenlignende studiet ble en ny parameter for å måle et orientasjonsresultat, kalt vektet troverdighetsindeks, foreslått. Vektet troverdighetsindeks gav en mer presis vurdering av mønstersammenligningsresultater enn den etablerte troverdighetsindeksen for alle testsituasjoner med simulert GaSb og mordenite i dette arbeidet For å sikre at metodene som blir brukt i dette arbeidet er riktig implementert, ble nøkkelfunksjoner i mønstersammenligningsprosedyren i pyXem testet. Dette inkluderer funksjonen for å lage en liste med kandidatorientasjoner i det symmetrireduserte orientasjonsrommet, og simuleringen av kinematiske diffraksjonsmønster. Det ble oppdaget at funksjonen for å generere kandidatorientasjoner leverte utilfredsstillende resultater for kubiske strukturer, fordi listen ikke dekket hele det symmetrireduserte orientasjonsrommet. Det er derfor viktig at den nåværende versjonen av pyXem blir videreutviklet for mønstersammenligning ved å korrigere problemene som ble påpekt i dette arbeidet.

Videre arbeid bør fokusere på å virereutvikle mønstersammenligningsprosedyren, i første omgang for rotasjonsserier, ved å forbedre hvordan mønstre blir simulerte. De nye metodene som ble implementerte i denne oppgaven forbedret ikke mønstersammenligning for testsituasjonene, men de kan muligens være bedre når rotasjonsserien blir avbildet med

presesjon av elektronstrålen, og når en bedre detektorteknologi er i bruk, som muliggjør direkte avbilding.

Preface and acknowledgments

This Master's thesis is submitted as part of the requirements for the degree of Master of Science in Applied Physics and Mathematics from the Norwegian University of Science and Technology (NTNU) in Trondheim. The work presented herein is in part a continuation of a 15 ECTS project work completed in the fall of 2019 and the suggestions for further work therein. The work was carried out during the spring semester of 2020, under the guidance of Professor Antonius Theodorus Johann van Helvoort at the Department of Physics. All method development and data analysis presented is the work of the author.

Foremost I would like to express my sincere gratitude to my supervisor Professor A.T.J. van Helvoort for his continous support and encouragement, without which this work would not have been possible. Thank you for extensive meetings and discussions, and for providing both profound insight in materials science and a constructive and enjoyable learning environment. I am truly grateful for the opportunity to write my Master's thesis under your guidance.

I would also like to thank my co-supervisor Tina Bergh, for continuous feedback and for providing invaluable input on my work during our weekly meetings. Thank you also to Håkon Wiik Ånes for your participation in discussions throughout the year, and for providing much appreciated help with the MTEX software and high performance computing. Thank you to Dr Duncan Johnstone and Phillip Crout for your help and encouragement with anything related to pyXem, and for your input and suggestions for the method development.

Eirik Opheim

Contents

Abbreviations

BFP	=	Back focal plane
CBED	=	Convergent beam electron diffraction
Cryo-EM	=	Cryogenic electron microscopy
CSM	=	Correlation score map
DF	=	Dark-field
EELS	=	Electron energy loss spectroscopy
FCC	=	Face-centered cubic
FFC	=	Full-frame correlation
NCC	=	Normalized cross-correlation
PED	=	Precession electron diffraction
RI	=	Reliability index
SAP	=	Selected area plane
SPED	=	Scanning precession electron diffraction
ZNCC	=	Zero-normalized cross-correlation

Chapter 1

Introduction

1.1 Motivation and background

In the history of engineering, understanding fundamental properties of materials has been a key factor in technological development. Central to the entire discipline of material sciences is the understanding of how microscopic structures relate to macroscopic properties. In the field of nanotechnology, the properties of a material are controlled by the length of the material in the nm regime [1][2]. In order to properly analyze engineering and nanomaterials we require a means to measure this length scale. Transmission electron microscopy (TEM) is a powerful tool in that length scale, and as technique well established.

TEM emerged during the 1930s when Max Knoll and Ernst Ruska built a microscope that achived a resolution of 50 nm and a magnification of 12 000x [3]. A lower bound to the obtainable resolution is given by the wavelength of the electrons in TEM. The reported resolution of 50 nm is much bigger than the wavelength of the electrons, that would be of 25 pm for 200 kV TEM experiments. The basic components of the microscope have not changed substantially since the late 1940s, but the introduction of aberration correction on the apparatus lenses allow modern TEM devices to routinely provide images with a resolution of 390 pm[4]. Although remarkable progress is made in terms of resolution, there is a downside: achieving this resolution comes at the cost of expensive instruments, requirements to the specimen and a cost in terms of electron dose inflicted to the specimen. Any of these could be a showstopper in achieving high spatial information. I want to work towards advantages of electron diffraction, as this technique has a lower cost due to its technical simplicity and requires a lower dose of electrons. Diffraction lies the basic principle in image formation in a TEM and in general it has been the principle behind structural analysis starting with Bragg (ZnS) [5], to discovering the double helix structure of DNA [6] to current rational drug design to battle COVID19 [7].

Relevant structural information such as what crystal phase or strain distribution we have around anomalies at the nm scale can be studied directly by electron diffraction. Detailed structural analysis can be done by convergent beam electron diffraction (CBED)[8]. However, due the strong matter-electron interaction, this and other electron diffraction based techniques will require more demanding dynamic diffraction refinements in the analysis which are complex. Recording series of diffraction patterns and applying dynamic refinements has demonstrated that ab initio structure determination can be done. A prime example is the work by Lukas Palentinus [9] where even hydrogen positions in aspirine molecules could be analyzed using very small volumes and rather simple TEM hardware. This approach is getting huge attention currently [10] as it can solve structures at low dose from 10 nm large particles, including proteins at a fraction of the costs analysis using corrected TEM and CryoEM [11]. One method for collecting three-dimensional electron diffraction data is Rotation Electron Diffraction (RED), which routinely collects complete three-dimensional electron diffraction data from crystals as small as tens of nanometres, several orders of magnitude smaller than what can be studied on a synchrotron light source [12].

An alternative way to reduce the dynamic contributions and hence to simplify the analysis, is to precess the electron beam and record a summed pattern [13]. As the incoming beam come under different angles the pattern has weaker dynamical intensity variations, and that can be analyzed by simpler kinematic diffraction theory [14]. The data analysis can in many practical cases of structural analysis be further reduced by considering candidate structures. The experimental patterns are compared to a simulated bank of diffraction patterns for relevant candidate phases and orientations. This process is labeled *template matching* and can be automated [15]. The best match between experimental pattern and an entry in the bank will give both crystal phase and orientation for the analyzed volume [16]. If the patterns are recorded point-by-point over the sample, phase and orientation maps can be constructed down to a few nm spatial resolution [17]. Although in general it is accepted that precessing the beam is beneficial for structural analysis [18][19], spatial resolution is reduced, exposure time (ie dose) is increased as one precession round is needed, and the intensity is debated [19] [20].

1.2 Objectives

A great deal of work has been undertaken to optimize experimental conditions to best fit a kinematic simulation model [13]. It is still unclear how accurate the final result of a template matching procedure for obtaining an orientation map is, when diffraction patterns simulated with kinematic theory are matched to experimental precessed and dynamic scattered diffraction patterns. Some of the applied algorithms have hardly been evaluated since they were introduced [15]. The software that offer these algorithms is static and often non-transparent. To address this, open-source software is used in the present project.

The major objective of this study was to provide an analysis of how well newly implemented correlation method perform compared to the current implementation of template matching in the crystallographic python library pyXem when experimental data is compared to kinematic simulated templates. This should provide general insights on the precision of the common approach of template matching for ED, allowing systematic improvements to the acquisition, simulation and matching routines. As the software applied for performing template matching is an open-source library still in the beta phase, this dissertation is dedicated to evaluating and improving key template matching functionality, and ensure that it is correct and robust for the wider user base. In particular, this functionality should be applicable to the problem of comparing multiple template matching results from a specific area on a single sample or, like in RED where ED data is collected for different sample orientations with a roughly know tilt relation to each other. By comparing template matching results from different sample orientations it can be determined if the obtained results from the template matching approach results are self-consistent. Such a procedure will be applied in this project, with the aim of better determining orientations and reducing uncertainty in orientations obtained from template matching.

Chapter 2

Basic Theory

In order to appreciate transmission electron microscopy and its applications to solid materials, a thorough understanding of crystallography is a prerequisite. All diffraction techniques make extensive use of the concept of *reciprocal space*, and TEM is a tool for directly probing this space. Our theoretical development of both the *crystal space* and the reciprocal space, has the mathematical footing of *non-Cartesian vector calculus*. The objective of the following discussion is to relate crystal structures and their orientation to a projection in the reciprocal space. We will be following the book Introduction to conventional Transmission Electron Microscopy[3].

2.1 Crystallography

From a purely mathematical point of view, crystallography can be described as *vector calculus in a rectilinear, but not necessarily orthonormal or orthogonal reference frame*. For such a system we will define the *vector dot product*, the *vector cross product*, the computation of the length of a vector and the angle between two vectors as our basic mathematical tools.

2.1.1 Initial definitions

We can define a *crystal structure* as a regular arrangement of atoms decorating a periodic, three-dimensional *lattice*. The lattice is described by three *basis vectors* $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$, so that every point on the lattice is a linear combination of the basis vectors

$$\mathbf{t}_{uvw} = u\mathbf{a}_1 + v\mathbf{a}_2 + w\mathbf{a}_3 \qquad u, v, w \in \mathbb{Z} \tag{2.1}$$

The vector in Equation (2.1) represents a *direction* in the crystal lattice and can be denoted by the symbol [uvw]. Negative indices are denoted by a *bar* above the corresponding index.

The length of a vector is represented by the norm symbol, so that $|\mathbf{a}_1|$ = a, $|\mathbf{a}_2|$ = b and $|\mathbf{a}_3|$ = c. The angles between the basis vectors are called α, β and γ. The set of

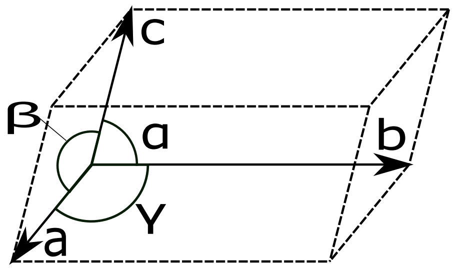


Figure 2.1: A general schematic representation of a unit cell, illustrating the basis vectors and the angles between them.

numbers $\{a, b, c, \alpha, \beta, \gamma\}$ are known as the *lattice parameters* of the unit cell illustrated in Figure 2.1. They can be used to distinguish between seven *crystal systems*. A basic property of a lattice is that all lattice sites can be selected as the *origin*. The seven crystal systems give rise to seven *primitive lattices* with one lattice site per unit cell. We can create additional lattices by introducing so called *centering vectors* inside the unit cell. The possible centering vectors are:

$$A = (0, \tfrac{1}{2}, \tfrac{1}{2})$$
$$B = (\tfrac{1}{2}, 0, \tfrac{1}{2})$$
$$C = (\tfrac{1}{2}, \tfrac{1}{2}, 0)$$
$$I = (\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2})$$

A lattice with the I-vector present is called *body centered.* If the A,B and C vectors are all present at the same time, the lattice is called *face centered.*

2.1.2 Fundamental mathematical tools

We will introduce the *Einstein summation convention* for the following section, and proceed to define the *vector dot product*, the *vector cross product*, the computation of the length of a vector and the angle between two vectors. The Einstein summation convention is defined: *If a subscript occurs twice in the same term of an equation, then a summation is implied over all values of this subscript and the corresponding summation sign need not be written.* A vector can be described by a linear combination of basis vectors in a rectilinear plane. In the Einstein summation convention, this is written

$$\mathbf{p} = p_i \mathbf{a}_i, \tag{2.2}$$

where p_i are the linear weights. The dot product of two vectors $\mathbf{p}$ and $\mathbf{q}$ can be defined by[21]

$$\mathbf{p} \cdot \mathbf{q} = |\mathbf{p}|\,|\mathbf{q}| \cos\theta, \tag{2.3}$$

and the cross product between **p** and **q** can be defined by

$$\mathbf{p} \times \mathbf{q} \equiv \sin\theta\, |\mathbf{p}|\, |\mathbf{q}|\, \hat{\mathbf{z}}, \tag{2.4}$$

where θ is the angle between **p** and **q** and $\hat{\mathbf{z}}$ is a unit vector perpendicular to both **p** and **q**. We will further develop the vector cross product when we have introduced the reciprocal space. By Equation (2.2) and (2.3) the length of a vector in a non-Cartesian reference frame is given by

$$|\mathbf{p}| = \sqrt{\mathbf{p} \cdot \mathbf{p}} = \sqrt{p_i(\mathbf{a}_i \cdot \mathbf{a}_j)p_j}.$$

The quantities $\mathbf{a}_i \cdot \mathbf{a}_j$ are fundamental in crystallography, and are therefore denoted by the symbol

$$g_{ij} \equiv \mathbf{a}_i \cdot \mathbf{a}_j = |\mathbf{a}_i|\, |\mathbf{a}_j| \cos\theta_{ij}, \tag{2.5}$$

commonly referred to as the *direct metric tensor*. The matrix g_{ij} is symmetric and has six independent components, corresponding to the six lattice parameters $\{a, b, c, \alpha, \beta, \gamma\}$. The dot product between two vectors **p** and **q** is given by

$$\mathbf{p} \cdot \mathbf{q} = p_i\mathbf{a}_i \cdot q_j\mathbf{a}_j = p_i g_{ij} q_j. \tag{2.6}$$

From Equation (2.3) the angle θ between two vectors **p** and **q** is given by

$$\theta = \cos^{-1}\left(\frac{\mathbf{p} \cdot \mathbf{q}}{|\mathbf{p}|\, |\mathbf{q}|}\right) = \cos^{-1}\left(\frac{p_i g_{ij} q_j}{\sqrt{p_i g_{ij} p_j}\sqrt{q_i g_{ij} q_j}}\right). \tag{2.7}$$

The tools developed above will be used as a mathematical basis for our further development of crystallography.

2.1.3 Crystal planes and Miller indices

Description of crystal planes is a fundamental part of crystallography. While there are many possible ways to describe a plane, it is customary to use the *hkl* notation and the so-called *Miller indices*. Although the Miller indices were used by several crystallographers before Miller, they are attributed to Miller due to their extensive use in his book on crystallography from 1839[22], and because he introduced the *hkl* notation. The Miller indices of a plane in an arbitrary crystal can be obtained in the following way.

1. If the plane goes through the origin, translate it along the plane normal so that it no longer contains the origin.
2. Determine the intercepts of the plane with the three basis vectors. Call those intercepts s_1, s_2 and s_3. The intercepts must be measured in units of the basis vector length. If a plane is parallel to one or more of the basis vectors, then the corresponding intercept values must be taken as ∞.
3. Invert all three intercepts. If one of the intercepts is ∞, then the corresponding number is zero.
4. Reduce the three numbers to the smallest possible integers (relative primes).

5. Write the three numbers surrounded by round brackets. This triplet [1] of numbers forms the Miller indices of the plane.

Parallel planes have the same Miller indices, but they behave different in a diffraction experiment. It is therefore often necessary to consider planes of type $(nh\ nk\ nl)$ where n is a non-zero integer.

It is useful to relate the Miller indices to a vector that conveys meaningful information about the corresponding plane. In order to do so, we will introduce a new set of basis vectors $\mathbf{a}_i^\star$, $\mathbf{a}_j^\star$ and $\mathbf{a}_k^\star$, known as the *reciprocal basis vectors*. We will later on see that the vector $\mathbf{g} = h\mathbf{a}_k^\star + k\mathbf{a}_k^\star + l\mathbf{a}_k^\star$ is perpendicular to the plane with Miller indices (hkl). The set of plane normals equivalent to (hkl) is denoted by the family hkl, and the number of members in the family depends on the point symmetry of the crystal [3].

2.1.4 Reciprocal Space

The reciprocal basis vectors can be defined from the following definition:

$$\mathbf{a}_i \cdot \mathbf{a}_j^\star \equiv \delta_{ij}, \tag{2.8}$$

where δ_{ij} is the *Kronecker delta*, which is equal to 1 for $i = j$ and 0 for $i \neq j$. From Equation (2.8), considering the cases where $\delta_{ij} = 0$, we have that

$$\mathbf{a}_1^\star = K(\mathbf{a}_2 \times \mathbf{a}_3),$$

where K is a constant and $\times$ is the cross product as defined in Equation (2.4). We can determine the value of K by considering a case where $\delta_{ij} = 1$. We then find

$$\mathbf{a}_1 \cdot \mathbf{a}_1^\star = K\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3) = 1,$$

giving

$$K = \frac{1}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)} \equiv \frac{1}{\Omega},$$

where Ω is the volume of the unit cell formed by the direct-space lattice vectors. Following the same procedure the set of reciprocal lattice vectors can be defined:

$$\begin{aligned}
\mathbf{a}_1^\star &= \frac{\mathbf{a}_2 \times \mathbf{a}_3}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)} \\
\mathbf{a}_2^\star &= \frac{\mathbf{a}_3 \times \mathbf{a}_1}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)} \\
\mathbf{a}_3^\star &= \frac{\mathbf{a}_1 \times \mathbf{a}_2}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)}.
\end{aligned} \tag{2.9}$$

The *reciprocal lattice* is defined as the set of end-points of the vectors

$$\mathbf{g} = h\mathbf{a}_1^\star + k\mathbf{a}_2^\star + l\mathbf{a}_3^\star = \sum_{i=1}^{3} g_i\mathbf{a}_i^\star = g_i\mathbf{a}_i^\star, \tag{2.10}$$

[1]Triagonal and hexagonal lattices may be described with four axes (hkil), where i = -(h+k).

where h, k and l are all integers. We will now investigate the relationship between the reciprocal lattice vector **g** and the plane with Miller indices (hkl).

We find all direct space vectors **r** that are perpendicular to the vector **g** by the use of Equation (2.3). This gives

$$0 = \mathbf{r} \cdot \mathbf{g} = (r_i \mathbf{a}_i) \cdot (g_j \mathbf{a}_j^\star) = r_i (\mathbf{a}_i \cdot \mathbf{a}_j^\star) g_j = r_i g_i, \tag{2.11}$$

where we have used Equation (2.8) to compute the last dot product. We see from Equation (2.11) that $hx + ky + lz = 0$ if **r** is perpendicular to **g**. This relation is the Equation of a plane through the origin of the direct crystal lattice. If a plane crosses the basis vectors $\mathbf{a}_i$ in the positions s_i, then the Equation of that plane is given by [23]

$$\frac{x}{s_1} + \frac{y}{s_2} + \frac{z}{s_3} = 1, \tag{2.12}$$

where (x,y,z) is a point in the plane. For parallel planes, the left hand side of Equation (2.12) can be written equal, whereas the right hand side will differ by a given factor. In particular, a plane containing the origin will have 0 on the right hand side of Equation (2.12). A proof on this is provided in Appendix A.1. Comparing Equations (2.11) and (2.12) we find that h, k and l are reciprocals of the intercepts of a plane with the direct lattice basis vectors. We conclude that *the reciprocal lattice vector **g**, with components (h,k,l), is in fact perpendicular to the plane with Miller indices (hkl).*

As the reciprocal lattice vector is perpendicular to the (hkl) plane, the unit normal to this plane is

$$\mathbf{n} = \frac{\mathbf{g}_{hkl}}{|\mathbf{g}_{hkl}|}.$$

The perpendicular distance from the origin to the (hkl) plane is given by the projection of any vector **t** ending in the plane onto the plane normal **n**. This distance is by definition the *interplanar spacing d_{hkl}*

$$\mathbf{t} \cdot \mathbf{n} = \mathbf{t} \cdot \frac{\mathbf{g}_{hkl}}{|\mathbf{g}_{hkl}|} \equiv d_{hkl}.$$

We know that the (hkl) plane intersects the direct basis vectors in the points $\frac{1}{h}$, $\frac{1}{k}$ and $\frac{1}{l}$. We can therefore set $\mathbf{t} = \frac{\mathbf{a}_1}{h}$ and obtain

$$\mathbf{t} \cdot \mathbf{g}_{hkl} = \frac{\mathbf{a}_1}{h} \cdot (h\mathbf{a}_1^\star + k\mathbf{a}_2^\star + l\mathbf{a}_3^\star) = \frac{\mathbf{a}_1}{h} \cdot h\mathbf{a}_1^\star = 1 = d_{hkl}\, |\mathbf{g}_{hkl}| \,.$$

We observe that we have

$$|\mathbf{g}_{hkl}| = \frac{1}{d_{hkl}}, \tag{2.13}$$

and we conclude that *The length of a reciprocal lattice vector is equal to the inverse of the spacing between the corresponding lattice planes.*

2.2 Electron-matter interaction

We have now had a look at how a crystal can be described in both direct and reciprocal space. We will proceed by examining the physics of a high energy electron beam passing

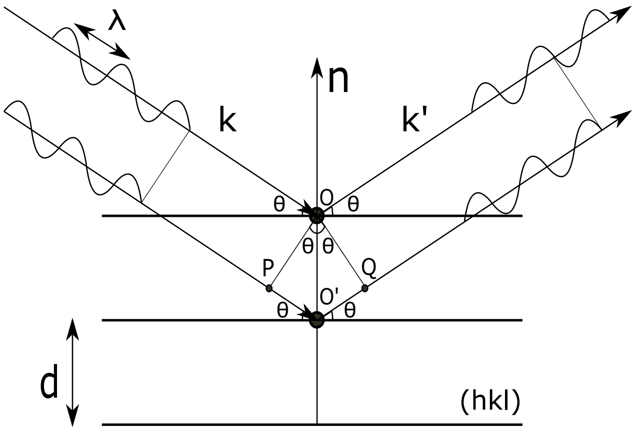


Figure 2.2: Bragg diffraction illustrated for a plane wave incoming at angle θ on the plane (hkl).

through a crystal material, as is the case in TEM. In our exploration of the topic a basic understanding of quantum mechanics is assumed, including an understanding of the bra-ket notation, and the interpretation of quantum states with regard to probability densities. For an instructive introduction to the topic, see the first chapter of professor Linders booklet [24], or any textbook on quantum mechanics. Certain aspects of relativity theory are also of importance, especially when computing the wavelength of high energy electrons. For a derivation of the relativistic de Broglie wavelength, please refer to [25].

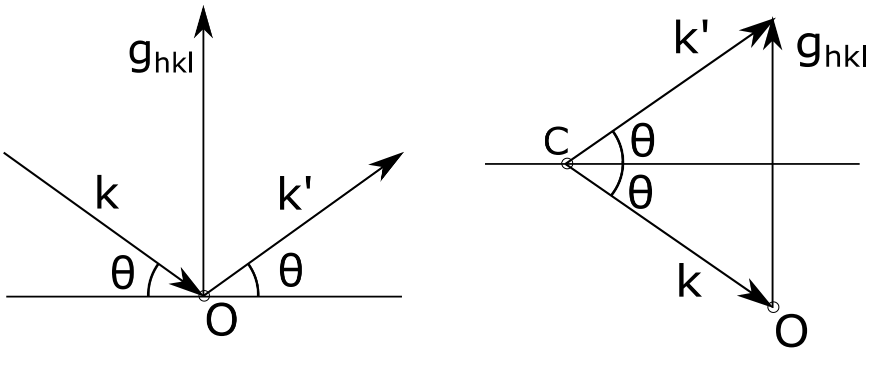
A TEM experiment consists of three stages,

- The illumination stage, in which a beam of electrons is created in a well-defined reference state.
- The interaction stage, in which the sample modifies the reference state of the electron beam.
- The observation stage, in which the modified electron beam is converted into a signal that can be detected by the human eye.

We will put the emphasis of this section on the interaction stage. In the interaction stage a beam of electrons is incident in certain angles on all planes in the material. By Snell's law the electrons will be reflected at the same angle as their incidence angle[26]. Different parts of a reflected beam will have travelled different distances, as seen in Figure 2.2. Bragg's law tells us that we will have constructive interference if the difference in distance travelled is an integer number of wavelengths

$$2\frac{d_{hkl}}{n}\sin\theta = \lambda. \tag{2.14}$$

While the Bragg Equation is elegant and handy in determining the diffraction angles θ, it is not of great use if one wants to calculate the absolute direction of a diffracted wave for



(a) **(b)**

Figure 2.3: (a) Illustration of how a reflection changes the direction of the electron wavevector **k** in reciprocal space. $\mathbf{g}_{hkl}$ is the reciprocal lattice vector corresponding to the plane where the reflection happens. (b) The reflected wavevector is translated to complete the vector sum $\mathbf{k}' = \mathbf{k} + \mathbf{g}_{hkl}$.

a given crystal structure. We would like to treat electrons as plane waves and look for a solution to the Bragg Equation in reciprocal space. By the de Broglie relation [27]

$$p = \frac{h}{\lambda}, \tag{2.15}$$

or equivalently [28]

$$\mathbf{p} = h\mathbf{k}, \tag{2.16}$$

where h is Planc's constant, an electron of momentum p can be described by a plane wave of wavelength λ. We see from Equation (2.16) that an electron can be described by both its momentum vector **p** and its wavevector **k**. The electron wavevector is a vector in reciprocal space which fully characterizes the direction and wavelength of a plane wave with respect to the crystal lattice. We remember from Section 2.1 that a plane can be described by a vector **g**, and seek to develop a reciprocal space equivalent of Equation (2.14). We see from Figure (2.3) that the relationship between the momentum vector of the wave before and after scattering is

$$\mathbf{k}' = \mathbf{k} + \mathbf{g}_{hkl}. \tag{2.17}$$

Projecting every term in the Equation onto $\mathbf{g}_{hkl}$ and using Equations (2.13) and (2.15) yields

$$\begin{aligned}\mathbf{k'} \cdot \mathbf{g}_{hkl} &= \mathbf{k} \cdot \mathbf{g}_{hkl} + \mathbf{g}_{hkl} \cdot \mathbf{g}_{hkl} \\ |\mathbf{k}|\,|\mathbf{g}_{hkl}| \sin\theta &= -\,|\mathbf{k}|\,|\mathbf{g}_{hkl}| \sin\theta + |\mathbf{g}_{hkl}|^2 \\ \frac{\sin\theta}{\lambda} &= -\frac{\sin\theta}{\lambda} + \frac{1}{d_{hkl}}\end{aligned}$$

from which the Bragg Equation follows. We observe that *the Bragg Equation is satisfied whenever the point C in Figure 2.3b lies on the perpendicular bisector plane of* $\mathbf{g}_{hkl}$.

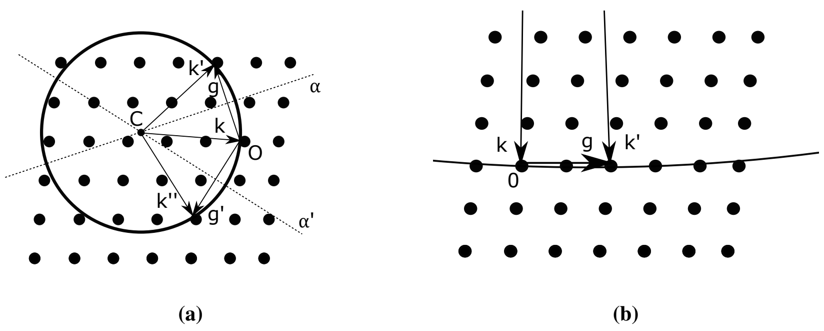


Figure 2.4: **(a)** Schematic representation of the Ewald Sphere, with center in C. Incoming electrons have wavevector **k** and scattered electrons have wavevectors **k**' and **k**". α and α' are the perpendicular bisectors of **g** and $\mathbf{g}'$. **(b)** For high energy electrons the wavevector **k** will be very long, and the Ewald sphere will have a large radius. It will therefore appear almost flat when passing through a crystal plane

The perpendicular bisector plane of $\mathbf{g}_{hkl}$ consists of all points of equal distance from the origin and the point (h,k,l) in reciprocal space. From this realization the simple geometric construction shown in Figure 2.4 can predict possible directions of the diffracted waves, when the incident wave vector and the crystal orientation are given. The construction is known as the *Ewald sphere*. To obtain the Ewald sphere, first draw the reciprocal lattice with origin O. Then draw the incident wave vector **k** so that the endpoint coincides with the origin. The starting point of **k** is taken to be the center of the sphere, of radius $|\mathbf{k}| = \frac{1}{\lambda}$. Whenever a reciprocal lattice point falls on the sphere, the Bragg condition is fulfilled and a diffracted wave with wavevector $\mathbf{k}' = \mathbf{k} + \mathbf{g}$ may occur. For the construction in Figure 2.4, $\mathbf{k}'$ and $\mathbf{k}''$ are both possible scattering directions. Although the Ewald sphere predicts scattering in a given direction we are not certain that scattering actually will occur. We will further examine scattering conditions in the following section.

2.2.1 Further scattering criteria

For the high energy electrons observed in TEM the dominant scattering effect is scattering due to the electrostatic potential of the specimen [29]. The quantum mechanical equation governing the interaction therefore has two important contributions: The kinetic energy term, which is controlled by the accelerating voltage of the microscope, and the *electrostatic lattice potential* $V(\mathbf{r})$, which describes how the incoming electrons interact with the sample. Finding the correct potential for a given crystal structure is a fundamental problem of solid state physics. It turns out that, to a very good approximation, solids can be considered to consist of *spherical point scatterers*, represented by *atomic scattering factors* $f^e(\mathbf{s})$, where **s** is a scattering vector. The true scatter potential is then the obtained scatter potential plus contributions from interatomic bonds. The contributions from interatomic bonds can be ignored for most TEM applications [3]. We will be considering scattering from a single atom, and then proceed to consider scattering from an infinite crystal and finally effects of having a crystal with a finite extent. Only elastic scattering is considered.

The scattering of electrons by a single atom is determined by the total charge distribution of that atom

$$\rho(\mathbf{r}) = |e|\,[\rho_n(\mathbf{r}) - \rho_e(\mathbf{r})], \tag{2.18}$$

where ρ_n is the nuclear charge distribution, and ρ_e is the electron charge distribution. The electrostatic atom potential $V^a(\mathbf{r})$ is related to the charge density through Poisson's Equation[30]

$$\Delta V^a(\mathbf{r}) = -\frac{|e|}{\epsilon_0}[\rho_n(\mathbf{r}) - \rho_e(\mathbf{r})], \tag{2.19}$$

where Δ is the Laplacian delta operator. The probability amplitude that an incident plane wave, with wave-vector **k** is scattered in direction **k**' by the atomic scattering potential $V^a(\mathbf{r})$ can be expressed in *bracket* notation, or equivalently in a familiar integral notation

$$P = \langle e^{2\pi i\mathbf{k}'\cdot\mathbf{r}}|V^a(\mathbf{r})|e^{2\pi i\mathbf{k}\cdot\mathbf{r}}\rangle = \int V^a(\mathbf{r})e^{-2\pi i\Delta\mathbf{k}\cdot\mathbf{r}}d\mathbf{r}, \tag{2.20}$$

where $\Delta\mathbf{k} = \mathbf{k}' - \mathbf{k}$ is the momentum transfer vector. We define the atomic scattering amplitude $f^e(\Delta\mathbf{k})$ to be equal to this probability amplitude, and note that this is the *Fourier transform* of the atomic scattering potential

$$f^e(\Delta\mathbf{k}) = \int V^a(\mathbf{r})e^{-2\pi i\Delta\mathbf{k}\cdot\mathbf{r}}d\mathbf{r}. \tag{2.21}$$

The atomic scattering factor $f^e(\Delta\mathbf{k})$ expresses *the probability that a certain elastic momentum transfer will occur*. The x-ray scattering amplitude $f^x(\Delta\mathbf{k})$ is defined as the fourier transform of the electron charge density $\rho_e(\mathbf{k})$ [31]. We can approximate the nuclear charge density by a delta function of weight Z, the atomic number, located at the origin. Applying now the inverse Fourier transformation to Equation (2.21) and inserting in Equation (2.19) we obtain

$$\Delta\int f^e(\Delta\mathbf{k})e^{2\pi i\Delta\mathbf{k}\cdot\mathbf{r}}d\mathbf{k} = -\frac{|e|}{\epsilon_0}\int[f^x(\Delta\mathbf{k}) - Z]e^{2\pi i\Delta\mathbf{k}\cdot\mathbf{r}}d\mathbf{k}. \tag{2.22}$$

As Equation (2.22) holds true for any volume in reciprocal space, the integrals can be omitted, and by letting the Laplace operate on **r** we obtain

$$f^e(\Delta\mathbf{k}) = \frac{|e|}{4\pi^2\epsilon_0\,|\Delta\mathbf{k}|^2}[f^x(\Delta\mathbf{k}) - Z]. \tag{2.23}$$

While studying crystals, it is customary to introduce the scattering vector $\Delta\mathbf{k} = 2\mathbf{s}$. By Braggs Equation (2.14) we can find the magnitude of **s** to be

$$\mathbf{s} = \frac{1}{2d_{hkl}} = \frac{\lambda}{\sin\theta} = \frac{\mathbf{g}_{hkl}}{2}. \tag{2.24}$$

We can describe the scattering factor in terms of **s** as

$$f^e(\mathbf{s}) = \frac{|e|}{16\pi^2\epsilon_0\,|\mathbf{s}|^2}[f^x(\mathbf{s}) - Z]. \tag{2.25}$$

known as the Mott-Bethe formula. Equations (2.24) and (2.25) can be combined to obtain different variations of the Mott-Bethe formula, useful for instance if scattering is to be considered in terms of scattering angles.

So far we have looked at scattering from a single atom, we now turn our attention to an infinite crystal. Such a system can be described by

$$\mathcal{T}(\mathbf{r}) = \sum_{u,v,w=-\infty}^{u,v,w=\infty} \delta(\mathbf{r} - \mathbf{t}_{uvw}), \tag{2.26}$$

where $\mathbf{t}_{uvw}$ is the direct lattice vector described in Equation (2.1). Every delta function corresponds to a single unit cell on the lattice, and every unit cell contains N atoms located at positions $\mathbf{r}_j$. The potential from a single unit cell can be described by

$$V_{Cell}(\mathbf{r}) = \sum_{j=1}^{N} V_j^a(\mathbf{r} - \mathbf{r}_j). \tag{2.27}$$

The potential for the whole crystal is found by repeating the potential from a single unit cell for every unit cell. This process can be described mathematically by a convolution between the structure of unit cells and the structure within a unit cell

$$V(\mathbf{r}) = V_{Cell}(\mathbf{r}) \otimes \mathcal{T}(\mathbf{r}). \tag{2.28}$$

The potential $V(\mathbf{r})$ is periodic with the periodicity of the lattice. We can transform our potential to frequency space by applying a Fourier transform

$$V(\mathbf{q}) = \frac{1}{\Omega}\mathcal{F}[V(\mathbf{r})]. \tag{2.29}$$

Applying the convolution theorem[32], and combining Equations (2.28) and (2.29) we find

$$V(\mathbf{q}) = \frac{1}{\Omega}\mathcal{F}[V_{Cell}(\mathbf{r})]\mathcal{F}[\mathcal{T}(\mathbf{r})]. \tag{2.30}$$

By applying the Fourier transform on Equation (2.27), and using our definition of the atomic scattering amplitude in Equation (2.21) we find

$$\mathcal{F}[V_{Cell}(\mathbf{r})] = \sum_{j=1}^{N} f_j^e e^{-2\pi i \mathbf{q}\cdot\mathbf{r}_j}, \tag{2.31}$$

where f_j is the scatter contribution from a single atom, as seen in Equation (2.25). Applying the Fourier transform on Equation (2.26) we find

$$\mathcal{F}[\mathcal{T}(\mathbf{r})] = \sum_{\mathbf{g}} \delta(\mathbf{q} - \mathbf{g}), \tag{2.32}$$

where $\mathbf{g}$ is the reciprocal lattice vector as defined in Equation (2.10). We define the Fourier coefficients of the electrostatic lattice potential as

$$V_{\mathbf{g}} = \frac{1}{\Omega}\sum_{j=1}^{N} f_j^e e^{-2\pi i \mathbf{q}\cdot\mathbf{r}_j}. \tag{2.33}$$

Inserting the Fourier transformed potential back into Equation (2.30) we obtain

$$V(\mathbf{q}) = \sum_{\mathbf{g}} V_{\mathbf{g}} \delta(\mathbf{q} - \mathbf{g}). \tag{2.34}$$

We observe that the Fourier transformed potential is discrete, and nonzero only on the reciprocal lattice. Setting $V_{\mathbf{g}}$ constant we can show that, for a periodic potential, the delta function in Equation (2.34) is equivalent to Bragg's law. If Bragg's law is fulfilled, but $V_{\mathbf{g}} = 0$ we will not have scattering.

A crystal is not infinite, and we will consider the effects of the finite extent of the crystal on the reciprocal potential. The shape of the crystal can be described by a function $D(\mathbf{r})$, with the value 1 inside the crystal and zero outside the crystal:

$$D(\mathbf{r}) = \begin{cases} 1 & \text{inside} \\ 0 & \text{outside} \end{cases} \tag{2.35}$$

This results in a finite crystal potential $V_f(\mathbf{r})$, expressed as a product between the infinite crystal potential and the shape function:

$$V_f(\mathbf{r}) = V(\mathbf{r}) D(\mathbf{r}) \tag{2.36}$$

We Fourier transform the potential and apply the convolution theorem to find

$$V_f(\mathbf{q}) = \mathcal{F}[V(\mathbf{r})] \otimes \mathcal{F}[D(\mathbf{r})]. \tag{2.37}$$

As the δ-function is the identity operator for the convolution product, we can combine Equations (2.34) and (2.37) to find

$$V_f(\mathbf{q}) = \sum_{\mathbf{g}} V_{\mathbf{g}} D(\mathbf{q} - \mathbf{g}). \tag{2.38}$$

We see that for a finite crystal *the reciprocal potential is no longer a delta function, but the Fourier transform of the shape function centered at the points of the reciprocal lattice.*

The shape function depends on the geometry of the incoming electron beam and the shape of the crystal. Most TEM experiments are carried out with a radially symmetric electron beam and the enlightened volume is therefore, to a good approximation, cylindrical with top and bottom surfaces not necessarily flat or parallel. The shape function $D(\mathbf{r})$ is determined by the intersection between the incident beam and the specimen. The thickness of the sample will usually be much smaller than the diameter of the enlightened area. For a sample with thickness z_0, and a beam of diameter $D >> z_0$ we can approximate $D(\mathbf{q})$ by calculating the Fourier transformation of a foil of infinite extent in x- and y-directions and thickness z_0:

$$D(\mathbf{q}) = z_0 \delta(q_x) \delta(q_y) \frac{\sin(\pi q_z z_0)}{\pi q_z z_0}. \tag{2.39}$$

The shape described by Equation (2.39) is a one-dimensional rod that extends in the direction perpendicular to the foil. Because these rods reside in reciprocal space, they are

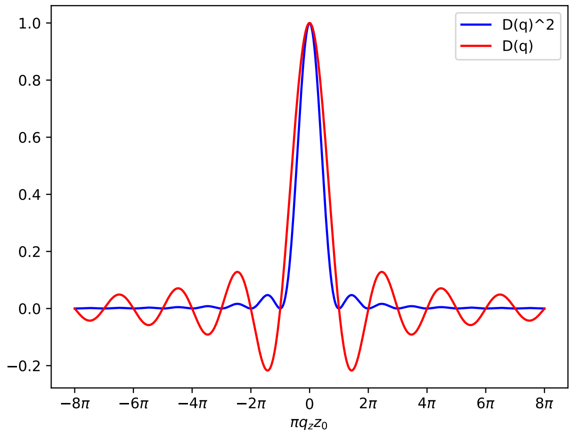


Figure 2.5: The reciprocal shape function $D(\mathbf{q})$ for a foil of thickness z_0, and the squared reciprocal shape function.

called *reciprocal lattice rods* or *relrods*. The rod extend infinitely in reciprocal space, but because of the rapidly decreasing nature of the function it can be approximated by a finite volume. The function and its square are shown in Figure 2.5. In our derivation we have made the approximation of an infinite foil. A general relrod will have a shape determined by the specifics of the sample and beam geometry. With the introduction of relrods, we no longer require that the Ewald sphere passes exactly through a reciprocal lattice point, but that it *intersects the corresponding relrod* to fulfill the Bragg condition. We introduce a deviation parameter **s** so that **s** is the distance from a point on the reciprocal lattice to the Ewald sphere along the direction of the major axis of the relrod. That is, $\mathbf{s} = s\mathbf{e}_z$, as seen in Figure 2.6b.

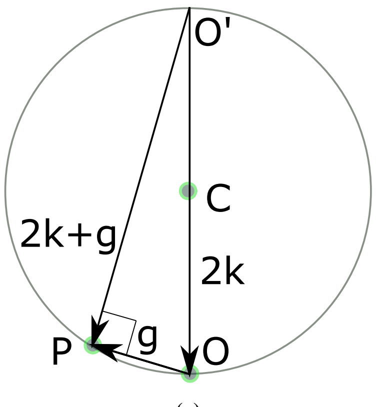
Considering now Figure 2.6a where the **k** vector has been extended to 2**k** in order to become the diagonal of the Ewald sphere. The reciprocal lattice vector **g** and the vector **2k + g** are drawn in the figure, for an arbitrary point on the Ewald sphere. Applying now a classical mathematical result: For a triangle $\triangle O'PO$ where C is the midpoint of the line OO'. If $|CP| = |CO|$, the triangle is a right triangle[33]. The inverse is also true. That is: For a triangle $\triangle O'PO$ where C is the midpoint of the line OO', if the angle $\angle O'PO$ is a right angle, then $|CP| = |CO|$. Applying the aforementioned results together with Equation (2.3) we can uniquely describe the Ewald sphere by:

$$(2\mathbf{k} + \mathbf{g}) \cdot \mathbf{g} = 0. \tag{2.40}$$

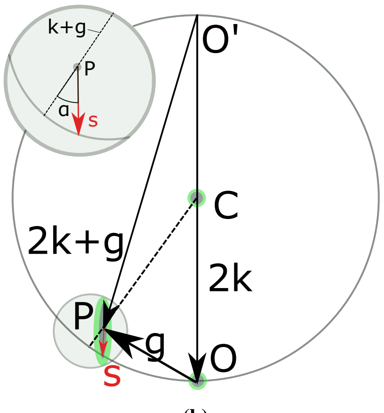
For the situation where the deviation parameter **s** is also present, as described in Figure 2.6b, we can follow the same logic as we applied in the process of deriving Equation (2.40) and obtain

$$(2\mathbf{k} + \mathbf{g} + \mathbf{s}) \cdot (\mathbf{g} + \mathbf{s}) = 0. \tag{2.41}$$

By introducing the angle α between the vectors ($\mathbf{k} + \mathbf{g}$) and **s** we can rewrite Equation



(a)



(b)

Figure 2.6: (a) Ewald sphere geometry where the k vector has been extended to become the diameter of the sphere. g is the vector between two arbitrary points on the Ewald sphere. The angle $\angle O'PO$ must be 90°. **(b)** The reciprocal lattice point is not on the Ewald sphere, but the relrod of P intersects the sphere. The vector **s** translates the center of P to the Ewald sphere. The gray circle to the upper left shows a magnified version of the gray circle in the lower left. The angle α is defined as the angle between **s** and the line defined by the direction (**k** + **g**).

(2.41) to

$$s + \frac{s^2}{2\,|\mathbf{k}+\mathbf{g}|\cos\alpha} = \frac{-\mathbf{g}\cdot(2\mathbf{k}+\mathbf{g})}{2\,|\mathbf{k}+\mathbf{g}|\cos\alpha} \equiv \mathbf{s}_{\mathbf{g}}, \tag{2.42}$$

where we have introduced the *excitation error* $\mathbf{s}_{\mathbf{g}}$. The excitation error is positive when the point P lies inside the Ewald sphere, as is the case in Figure 2.6b, and negative when P lies outside the Ewald sphere. For high energy TEM the approximation $(\mathbf{k}+\mathbf{g}) \approx \mathbf{k}$ can be made to simplify Equation (2.42). The excitation error is an important parameter in TEM, as the observed contrast for both perfect and defect crystals is highly dependant on $\mathbf{s}_{\mathbf{g}}$.

So far we have assumed that scattering is elastic, and that the detected scattering has experienced a single scattering event. In TEM we have both inelastic scattering[34] and multiple scattering. The effects of inelastic scattering are the apperance of Kikuchi lines for relatively thick samples [35]. We also have observed intensity in kinematically forbidden scattering directions when the forbidden direction can be expressed as the sum of two allowed scatter directions [36], and the observed intensities deviate from the kinematically predicted intensities. The theory of multiple scattering is the theory of *dynamic scattering*, a more correct description of intensity distribution for a real sample, and well understood. For the purposes of this report however, diffraction will be treated by the simpler kinematic theory.

By considering a simplified example an estimate of the boundaries of the kinematic model is made. In Equation (2.20) we have that the probability for an incident electron in a given state to enter another state is given by the Fourier transform of the electrostatic potential. We know from quantum mechanics that the probability density of finding a particle in a given state equals the square of the wavefunction of that given state. Assuming now that all incoming electrons are in a given state $|e^{2\pi i\mathbf{k}\cdot\mathbf{r}}\rangle$, the direct beam. We let the sample have the form of a foil of thickness z_0, and we make the two-beam approximation. That is, all scattering goes from the direct beam to a single reflection. Diffracted electrons are thus all in the state $|e^{2\pi i\mathbf{k'}\cdot\mathbf{r}}\rangle$. As we initially had no electrons in the scattered state, the *weight* of this state at the bottom of the sample is given by the Fourier transform of the electrostatic potential, as seen in Equation (2.20). Using the result from Equation (2.39) for the foil, and using that we have a single point of diffraction we obtain

$$c_{\mathbf{k'}} = \mathcal{P} = V_{\mathbf{g}} t \frac{\sin(\pi s_z t)}{\pi s_z t}. \tag{2.43}$$

Where s_z is the excitation error in the z-direction and t is the thickness of the sample. For othogonal wavefunctions, as is the case for the two beam approximation, the intensity at the point of diffraction is the square of the weight

$$I_{\mathbf{k'}} = c^2_{\mathbf{k'}} = (V_{\mathbf{g}} t)^2 \frac{\sin^2(\pi s_z t)}{(\pi s_z t)^2}. \tag{2.44}$$

By extracting a result from dynamic theory, we can rewrite Equation (2.44) in terms of the *extinction distance* ξ_g, defined as the distance at which the direct beam becomes zero and the reflected beam is at a maximum. We then have [37]

$$I_{\mathbf{k'}} = \frac{\pi^2}{\xi_g^2} \frac{\sin^2(\pi z_0 s_{\mathbf{g})}}{(\pi s_{\mathbf{g}})^2}, \tag{2.45}$$

and by assuming that we are exactly on a bragg reflection, that is $s_{\mathbf{g}} = 0$, we find

$$I_{\mathbf{k'}} = \frac{\pi^2 z_0^2}{\xi_g^2}. \tag{2.46}$$

By demanding that this intensity is much smaller than the direct beam intensity we deduce

$$z_0 < \frac{\xi_g}{10}. \tag{2.47}$$

For the kinematic approach to be valid, we require that the sample thickness is less than one tenth of the extinction length. This result is valid for the two-beam approximation. In general, multi-beam effects will limit the validity of this approximation, but it has been used with some success in certain cases [38], and can be used as a rough estimate.

As we will see in Section 2.3.2 the apparent effect of multiple elastic scattering can be decreased by letting the electron beam precess around an axis and sum the diffraction patterns from a full turn around the axis. This technique is known as Scanning Presession Electron Diffraction (SPED), and will be further examined in the remainder of the report.

2.3 The transmission electron microscope and electron diffraction techniques

We have so far introduced a notation for crystallography covering both the direct and the reciprocal space, and we have examined how electrons are diffracted when passing through solids. The process of sending high energy electrons through thin crystal samples and observing the resulting patterns is what enables transmission electron microscopy to provide images with information about the crystal structure on the length scale of interatomic spacing. In this section we will examine the key components of a microscope used for transmission electron microscopy, and examine one of the many imaging techniques available in TEM.

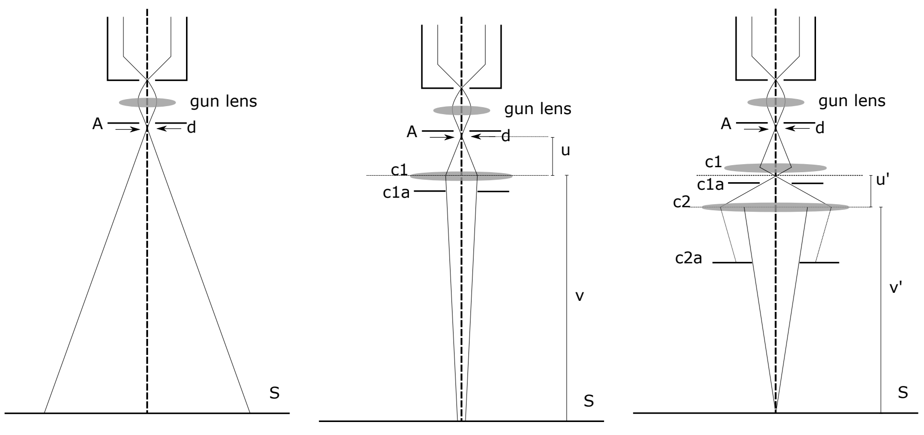
2.3.1 Components

The microscope consists of five main parts, when used in transmission mode: The electron gun, the illumination stage, the objective lens, the magnification and projection system, often with three or more lenses, and the detector. In addition to the five main stages, the microscope contains a number of subsystems that are important for the machine to run.

- *The high voltage system* is usually located in a separate unit and is connected to the electron gun by a thick cable.
- *The vacuum system* allows electrons to travel through the column without getting absorbed or scattered.
- *The cooling system* prevents the lens coils from heating when large electric currents are passing through them.
- *Radiation shields* protect the surroundings of the microscope from the radiation emitted whenever electrons enter solid matter.
- *The photographic camera system* allows the user to record the patterns acquired in the microscope.

The main purpose of the electron gun and the illumination system is to produce a steady current of electrons in a well known reference state, meaning that emitted electrons have a constant kinetic energy. Ideally, all electrons should emerge from a single point on the filament surface, so that the electron beam can be treated as a *point source*. Furthermore, electrons should travel close to the *optical axis* of the microscope to reduce the effect of *lens abberations* and the number of emitted electrons should be high enough to give a steady current of electrons having passed the sample and reached the detector, so that an image can be obtained in reasonable time. These quality statements can be expressed in terms of mathematical expressions, and the design of electron guns relies on finding geometries of the filament and potential surfaces around the tip that give the desired properties.

The electrons emitted from the electron gun will be further enhanced into a beam of electrons in a well-defined reference state in the illumination system. With no lenses, the electrons emitted from the electron gun will spread over a large area before it reaches the specimen, as shown in Figure 2.7a. By introducing a lens in the path of the electrons we can control the spread of the beam, and focus the beam to a spot of diameter $D_{spot} = d\frac{v}{u}$, where d is the gun crossover diameter, and u and v are defined as seen in Figure 2.7b. We would like to have as small spot size as possible, and therefore choose to have a very strong lens $C1$, giving a small v. This will result in the electron beam exiting the lens at



(a) **(b)** **(c)**

Figure 2.7: **(a)** No lens inserted. An apperature filters out parts of the electron beam. **(b)** Same setup with one lens. The size of the beam on the sample, S, is determined by the gun crossover diameter d, the length from beam crossover to lens, u and the length from beam to sample v. The apperature $c1a$ filters out parts of the electron beam. **(c)** With two lenses, the size of the beam when it reaches the sample will still depend on u and v, but they are defined as show in Figure 2.7c. A smaller beam diameter can thus be obtained. The apperature $c2a$ filters out high angle parts of the beam.

large angels, even after passing the $C1a$ aperture, as seen in 2.7c. We therefore introduce a second lens, focusing the beam to a small spot on the sample. The second lens, $C2$, is in focus when its focal length equals the distance from the lens to the sample. The variable range aperature below the second lens, $C2a$, limits the angular range of electron trajectories at the cost of decreased beam current.

While the primary function of the illumination stage is to create an electron beam in a well-defined reference state, the primary function of the objective lens is to bring the various diffracted electron beams to a cross-over while introducing minimal lens aberrations. The objective lens can have different components depending on the imaging mode. Figure 2.8 shows a possible objective lens design in microprobe mode, which is a useful mode for diffraction techniques. A condensor mini lens focuses the beam and creates a crossover right above the objective condensor lens. The electrons leaving the crossover are focused in a parallel, very narrow beam. When the parallel beam passes through the sample it is focused into reflections at different angles, before an objective imaging lens refocuses the beam to a symmetric crossover. An aperture is placed in the back focal plane to limit the range of accepted angles of incoming electrons.

Electrons passing through the objective aperture of the illumination stage enter the magnification and projection system, consisting of at least three lenses. With only the objective lens from the illumination stage present, we would have a fixed magnification, as the distance from the objective lens to the detector screen would determine the focal length of the objective lens, and thus also the magnification. By introducing a projection lens we obtain variable magnification, as illustrated for two different combinations of lens

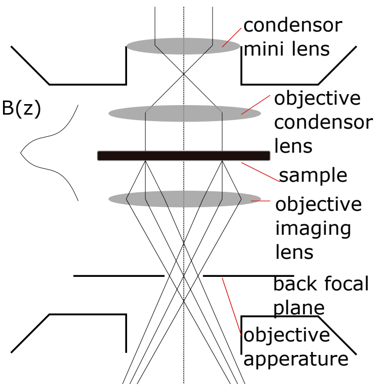
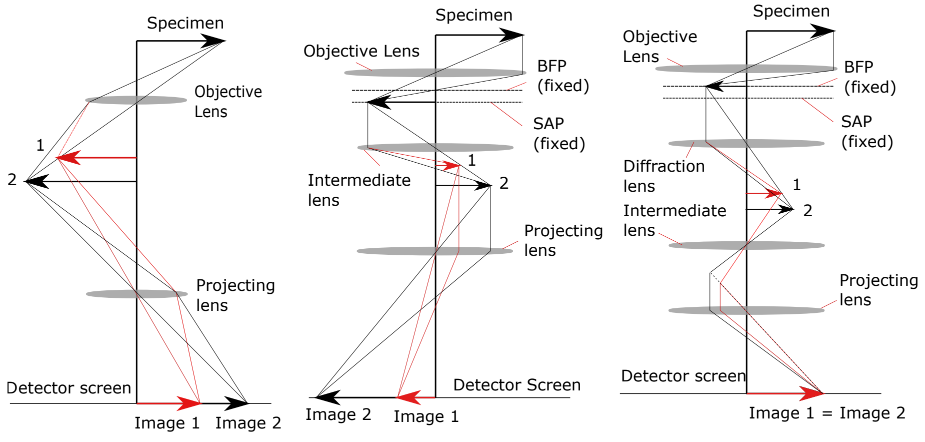


Figure 2.8: The objective lens consists of two or three lenses, the sample and an objective aperture in the back focal plane.

strengths 1 and 2 in Figure 2.9a. This setup achives our goal of magnifying the image, but has the drawback that it relies on changing the current in the objective lens. This is not desirable, as changing the objective lens current can cause the temperature of the lens to change, which in turn may lead to specimen drifts. Furthermore, having a constant objective lens strength gives a fixed location for its focal and image planes, both having an important role in conventional TEM work. It is therefore custumary to take the strength of the objective lens as a given, and introduce an intermediate lens between the objective lens and the projection lens to obtain variable magnification. In order to deal with the combined effects of lens aberrations, it is essential to be able to reach the desired magnification in more than one way. By inserting a third lens, the diffraction lens, a given magnification can be obtained in multiple ways, and aberration effects can be analyzed.

In order to obtain information from the electron trajectories, the electrons are counted as they hit a detector screen. A good detector screen should have sensitivity down to single electrons while maintaining a low noise level. It should maintain a linear relationship between incoming intensity and measured signal for a wide range of intensities. Furthermore, a high spatial resolution is required and the detector should have a high readout speed. We also require that results are reproducible, that is, for equal input the detector will provide the same result.

This setup allows the examination of materials on the atomic length scale, and can obtain images in both direct and reciprocal space. There are a variety of methods available in transmission electron microscopy. In dark field (DF) imaging only a specific diffracted beam is selected and used to create an image. Electron energy loss spectroscopy (EELS) allows for a three dimensional image of the specimen to be generated by comparing the energy of the detected electrons to the energy of incoming electrons. This report will focus on a technique called scanning precession electron diffraction (SPED), where the incoming electron beam precesses around the optical axis. An advantage with the SPED technique are that the detected patterns in many cases can be approximated better by kinematic diffraction theory than conventional patterns. Furthermore, the geometry of the processing Ewald sphere ensures that more reflections are seen relative to a stationary incoming



(a) **(b)** **(c)**

Figure 2.9: **(a)** System with objective lens and projecting lens. Two images with different magnification show how one can achieve magnification by altering the strengths of the objective lens and the projecting lens. **(b)** Same setup with an extra lens, so that the strength of the objective length is kept fixed, and thus also the back focal plane (BFP) and the selected area plane (SAP). For the demonstrated setup the image in the selected area plane is imaged. **(c)** With three lenses, multiple combinations of lens strength can give the same magnification. In this setup the back focal plane is imaged, this produces a magnified diffraction pattern on the viewing screen.

electron beam.

2.3.2 Scanning Precession Electron Diffraction

By scanning the electron beam, diffraction patterns can be acquired pixel-by-pixel to build up a four dimensional data set in which a two dimensional diffraction pattern is stored for each two dimensional real space position. This type of diffraction technique can be used to identify domain structures and changes in orientation across the examined area. In SPED, the electron beam processes around the optical axis of the microscope, an integer number of complete cycles for each pixel. The advantages of using a precession electron diffraction (PED) pattern over a static diffraction pattern is that a far greater number of reflections are seen and that the scatter intensities appear 'more kinematic'. Information that can be obtained from analysis of SPED data include form of overall grain structure, texture and orientation maps.

The phenomen of more reflections appearing when the beam is precessed around the optical axis of the microscope is explained by kinematic diffraction theory. As seen in Section 2.2.1, for a reflection to be seen in the diffraction pattern we require that the relrod of the given reflection intersects the Ewald sphere. For a static diffraction pattern the Ewald sphere has a given thickness depending on the variance in electron momentum, but remains in a fixed position in reciprocal space. For the case of a precessing electron beam the Ewald sphere sweeps an area, as seen in Figure 2.10. This is because the momentum

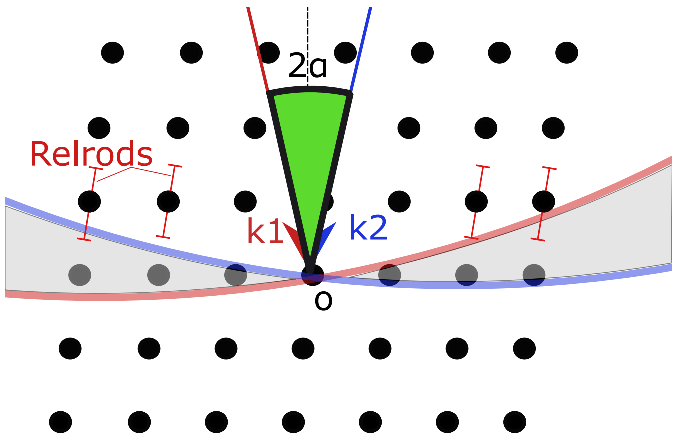


Figure 2.10: Reflections are activated for a point in the reciprocal space if its corresponding relrod intersects the gray area swept by the precessing ewald sphere. The incoming wave vectors **k1** and **k2** are equal in magnitude, and represent the initial wavevector and the wavevector when half a precession cycle is completed. The precession angle is α. In the illustration the sample is slightly tilted, so that the relrods lie at an angle with the direct beam axis.

of the incoming electrons precesses in reciprocal space. For the static electron beam we require that the relrod of a given point on the reciprocal lattice intersects the ewald sphere, whereas for the precession case it suffices that the relrod *intersects the area swept by the Ewald sphere*. It is therefore much more likely to see a reflection from a given point in the case of a precessing beam of incoming electrons.

Early works in precession electron diffraction suggested that for small scattering angles using kinematic scattering theory would lead to poor refinements because of dynamic effects. More detailed analysis has shown that although diffraction intensities are not strictly kinematic in nature, they can in many cases be treated like kinematic intensites and be used to solve structures[39]. A great deal of work has been done to establish the best conditions for PED, especially in terms of precession angle[13]. Still, tests on known structures have shown that applying a full dynamic treatment significantly improves the quality of the result[40].

In this work, the purpose of the obtained diffraction patterns will be to compare them with a library of possible simulated diffraction patterns from a set of orientations and structures. To achieve reliable pattern matching it is beneficial to have many reflections, indicating that we should choose a large precession angle. However, this limits the angular and spatial resolution achievable [13]. As a compromise between a reliable comparison between simulated and observed diffraction patterns and a good angular resolution most SPED examinations are performed at a typical precession angle of around $0.5 - 1.0°$.

2.3.3 Rotation electron diffraction

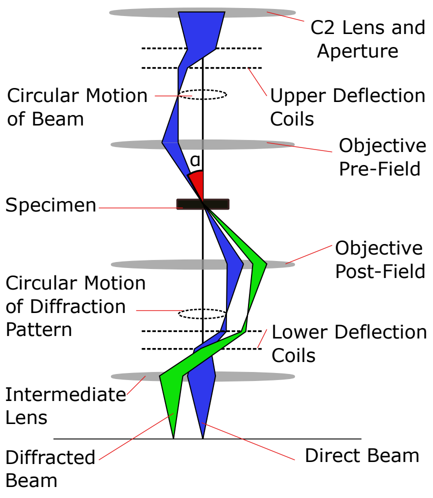
SPED has proven to be an excellent tool for solving simple structures with a short unit cell dimension ($\sim$4 Å), where a single diffraction pattern suffices to resolve atoms [15]. Structures with longer unit cell dimensions require multiple projections in order to be solved [41]. While combinations of techniques such as X-ray powder diffraction and multiple PED projections have been used to solve some of the most complex zeolites known [42], the process is extremely time-consuming. To address this problem, a method for collecting complete 3D electron diffraction data in a standard TEM instrument was proposed [41], and is commonly referred to as rotation electron diffraction (RED).

RED data collection is performed by combining discrete goniometer tilt steps, usually around 2 to 3 degrees, with very fine beam tilt steps, typically $0.05 - 0.20°$, along a common tilt axis. The goniometer steps usually range from around -40 to 40 degrees. For every goniometer step, a series of images is taken with beam tilt ranging from -α to α degrees, where the gonimeter tilt is 2α. The data for a given goniometer step is then found by summing up the images from every beam tilt step. A slight overlap between images from different goniometer steps can be used to account for the limited accuracy of a goniometer, and secure that data collection is continuous and without gaps. The two dimensional images can be merged into a three dimensional data-set, from which the reciprocal space can be reconstructed. The structure unit cell can be determined as described by W. Wan et al [12].

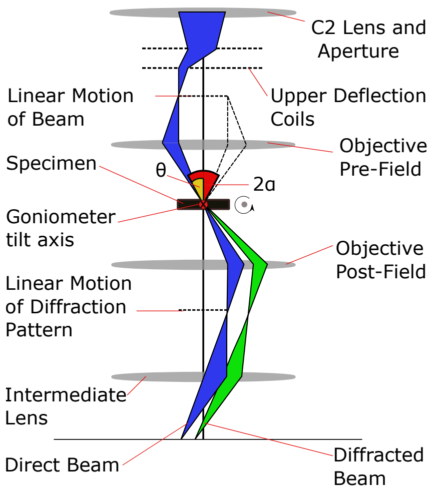
The RED data collection resembles that of the PED method, where the electron beam precesses in circle around the beam direction axis and images from distinct points in the circle are summed up. The difference is that RED shifts the beam along a line that passes through the beam direction axis, whereas PED precesses the beam in a circle as illustrated in Figure 2.11. Furthermore, the goniometer tilts in the same direction as the beam tilt, whereas a typical SPED image will scan across the surface of a material. While it is difficult to sort out which reflections are fully recorded and which are only partially integrated for a given PED image, RED data is better suited for collecting complete 3D diffraction data.

2.4 Representation of orientation

In order to properly describe and compare orientations of crystal lattices, we require a terminology for describing orientations. In material science the orientation of a crystal lattice is described by means of a rotation relative to an external reference frame [43]. A number of rotation representations are in use, each with distinct advantages and disadvantages with respect to calculations and data visualization. In this report, orientations will be described by the Euler angles representation, using the *Bunge convention*. Angular distance will be described by a metric related to the *axis-angle representation*. Algorithms used for data processing rely on *rotation matrices*, *unit quaternions* and *Rodrigues-Frank vectors*. The interpretation of Euler angles, the axis-angle representation and rotation matrices will be outlined below. For an introduction to the remaining representations, refer to [43].



(a)



(b)

Figure 2.11: (a) Ray diagram for SPED, showing how the beam is rocked by deflection coils, generating a circular motion, and a constant precession angle α. The beam is later de-rocked by a set of deflection coils. **(b)** Ray diagram for RED, showing how the beam is rocked by deflection coils, along a line. The beam generates an angle, θ, with the specimen normal ranging from -α to α, where 2 α is the goniometer tilt angle, not to be confused with the precession angle of SPED. The tilt axes of θ and α are aligned. For RED, there is no de-rocking after the objective lens, but the beam tilt is stored along with the resulting diffraction pattern. The figure is inspired by Figure 1 in [13].

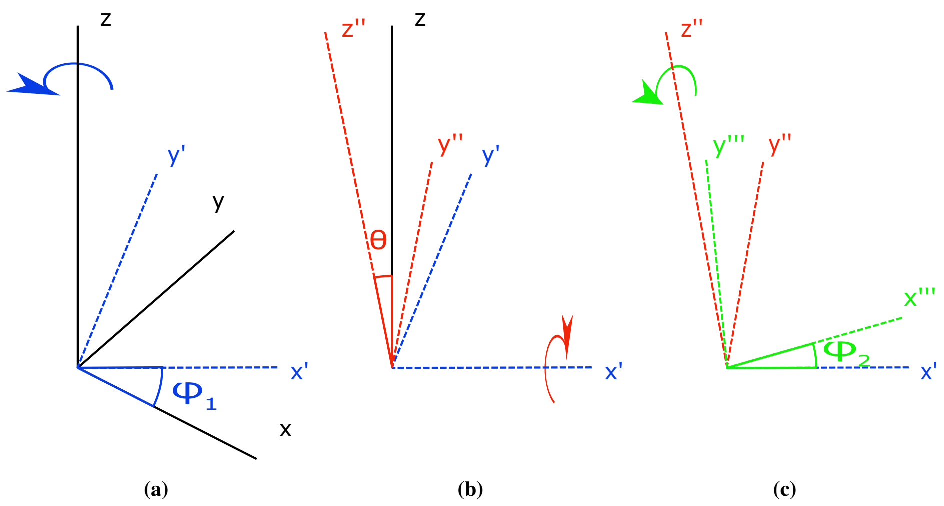


Figure 2.12: Euler angles in three steps. **(a)** Starting from external reference frame, a rotation around z-axis shifting the x axis to x' and the y axis to y'. This rotation is referred to as ϕ_1. **(b)** Rotation around shifted x-axis. This rotation is referred to as θ. **(c)** Rotation around shifted z-axis. This rotation is referred to as ϕ_2.

2.4.1 Euler angles

In the Euler angle representation, orientations are presented by rotating the reference frame around three axes. In the Bunge convension [44], theses axes are the z-axis for the first rotation, the shifted x-axis for the second rotation and the shifted z-axis for the third rotation. Rotations are taken to be positive for counterclockwise rotations when looking from the end point of the rotation axis towards the origin. Euler angles are represented as (ϕ_1,θ,ϕ_2), interpreted as seen in Figure 2.12. The starting frame (xyz) is the external reference frame.

2.4.2 Axis-angle representation

An orientation can also be represented by an axis $\hat{\mathbf{n}}$, and an angle ω. This representation exploits Euler's theorem on the motion of a rigid body, stating that any rigid body displacement with one fixed point can be described as a rotation about some axis[45]. The axis-angle representation consists of four numbers, three defining the axis and one defining the rotation. Conventionally, the angle ω is defined in the interval [0,π]. The metric used to measure angular distance between two orientations is *the smallest obtainable angle in the axis-angle representation of the rotation from one orientation to the other* [44].

2.4.3 Rotation matrices

Rotation matrices have two possible interpretations. In the active interpretation a matrix is operating on an object, whereas in the passive interpretation where a matrix is operating on the reference frame. In this work the passive interpretation is used for three dimensional rotation matrices, as is the common practice in the materials field, and particularly in the texture community[43].

Let the original and the rotated reference frame be represented by the Cartesian unit vectors $\boldsymbol{e}_j$ and $\boldsymbol{e}_i^{'}$, respectively. If we now define a matrix α_{ij} so that

$$\alpha_{ij} = \boldsymbol{e}_i^{'} \cdot \boldsymbol{e}_j \tag{2.48}$$

then the matrix α_{ij} represents a passive rotation from the original reference frame to the rotated reference frame. Note that in Equation (2.48) we have again made use of the Einstein summation convention introduced in section 2.1.2. With α_{ij} representing a passive rotation, we have

$$\boldsymbol{e}_i^{'} = \alpha_{ij}\boldsymbol{e}_j. \tag{2.49}$$

A vector $\boldsymbol{p}$, that has components p_i in the original reference frame will have components

$$p_i^{'} = \alpha_{ij}p_j \tag{2.50}$$

in the rotated reference frame.

Rotation matrices belong to the set of special orthonormal matrices *SO(3)*. For every row and column in the matrix, the sum of the squares of the indices equals 1. Furthermore the determinant of a Rotation matrix is 1, and the transpose of a rotation matrix is equal to its inverse.

2.4.4 Visualization

Conventional TEM observations usually result in two-dimensional images or diffraction patterns. As the object giving rise to those images is three-dimensional, it is necessary to represent the relevant crystallographic information in a two dimensional drawing. One of the most important 2D representations is *the stereographic projection.*

The Stereographic Projection

A stereographic projection is a two dimensional representation of a certain characteristic of a three dimensional object [3]. This characteristic can be anything with a directional character, such as the set of directions parallel to the surfaces of the object. While the stereographic projection was developed in ancient times, it became increasingly popular after its *conformality*, the correctness of angles and shapes, was realized by Edmond Halley [46]. The application of the stereographic projection in crystallography was first suggested by the work of F. E. Naumann and was further developed by W. H. Miller as a systematic way to represent the normals to the faces of natural crystals in a 2D drawing [47].

Starting from the plane normals described above, a point P can be defined as the intersection between a plane normal and an enclosing sphere. In order to obtain the stereographic projection, SP, of P, the point is connected with the south pole, S, by a straight line. The intersection between the line connecting the point and the south pole and the Equatorial plane, EP, gives the stereographic projection, as seen in Figure 2.14. Points on the southern hemisphere would be projected outside the projection circle of the Equatorial plane. In order to avoid very large figures, it is customary to project points on the southern hemisphere to the north pole, N, and represent the projections with open circles. Angles can be read out from the stereographic projection by the use of the *Wulff net* [3]. The Wulff net is the the projection of a collection of great and small circles, which represent lines of latitude (small circles) and lines of longitude (great circles) on the sphere, as seen in Figure 2.13 .

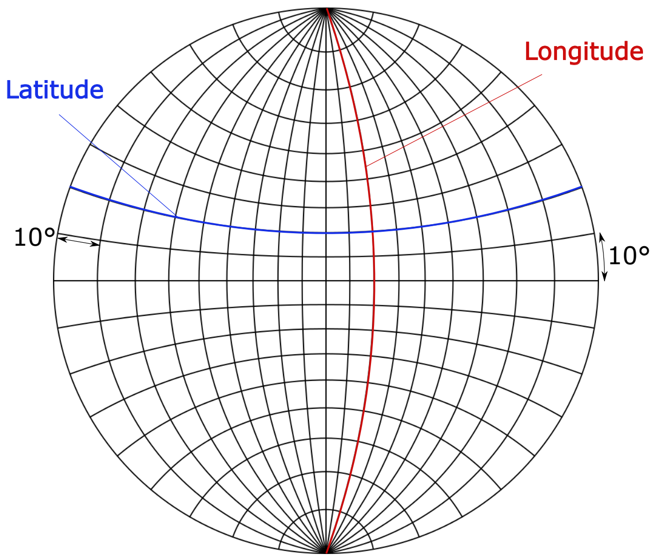


Figure 2.13: A standard Wulff net has a resolution of 1 degree, and a diameter of 20 cm. For the purposes of illustration, the resolution has been set to 10 degrees and the figure dimension has been reduced.

Pole Figures and Inverse Pole figures

The stereographic projection of the directional distribution of all crystallographically equivalent lattice vectors $[uvw]$ or reciprocal lattice vectors (hkl) is referred to as Pole figures [48]. The pole figure does not show the orientation directly, but the effect the orientation has on the respective pole distribution. The directional distribution of the crystal reference vectors in terms of a fixed crystallographic frame (The specimen frame) depicts the inverse pole figure of the three reference vectors.

The commonly used term *Inverse Pole Figure* (IPF) is related to the inverse pole figure, but has a very specific interpretation. Properly expressed, the IPF is the directional distribution of the crystal reference vectors in terms of a fixed crystallographic frame, *within the fundamental region of the respective Laue group*. That is, only the unique part of the inverse pole figure is shown. This concept is illustrated in Figure 2.15.

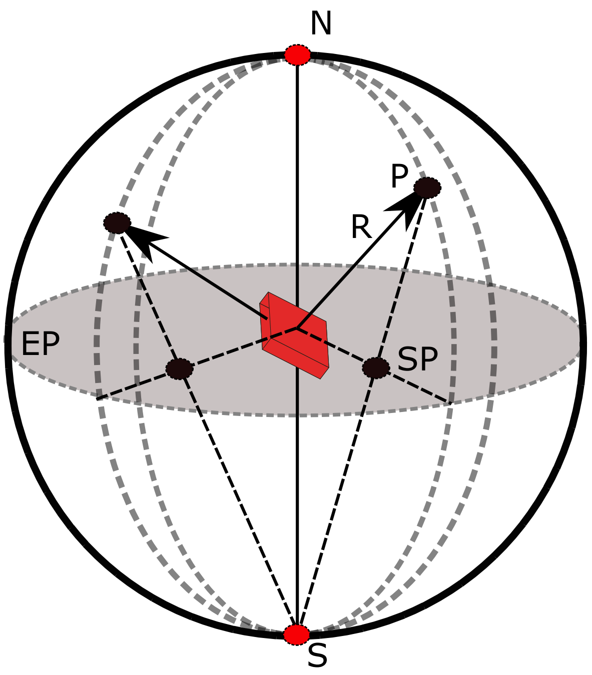


Figure 2.14: The stereographic projection, SP, of a point, P, on the sphere is found by connecting the point to the south pole, S. The intersection between the line connecting the point and the south pole and the equatorial plane, EP, is the stereographic projection, SP.

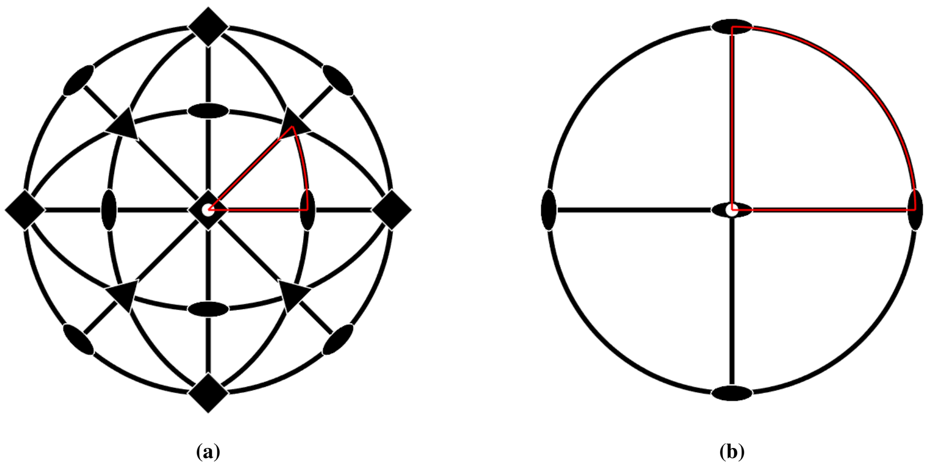


Figure 2.15: The fundamental regions for **(a)** a cubic structure **(b)** an orthorhobic structure.

2.5 Data processing

The ultimate goal for the data processing step is to find the correct orientation and structure for each pixel in the two dimensional region of interest of the probed sample. This is done by comparing simulated diffraction patterns from a set of orientations and structures to the diffraction pattern stored in each pixel. Data from a SPED examination has the form of a four-dimensional array, where each location stores an intensity from 0, no intensity, to the bit dept, the maximum value for intensity measures in the detector. Two of the array indices correspond to the two dimensional position on the sample, the *navigation dimension* and can be used to select a given pixel or a set of pixels. The other two correspond to the coordinates in the diffraction pattern of a given pixel, the *signal dimension*. For each pixel in the navigation dimension the stored diffraction pattern is compared to the set of simulated diffraction patterns, and through a method of comparison the best fit is chosen. The orientation that gave rise to the simulated diffraction pattern with the best fit is then taken to be the orientation of the sample in that pixel. A flow diagram of the process is found in Figure 2.16, and the key aspects for each step is listed in Table 2.1. A more detailed explanation of how each step is implemented is provided in Appendix B.

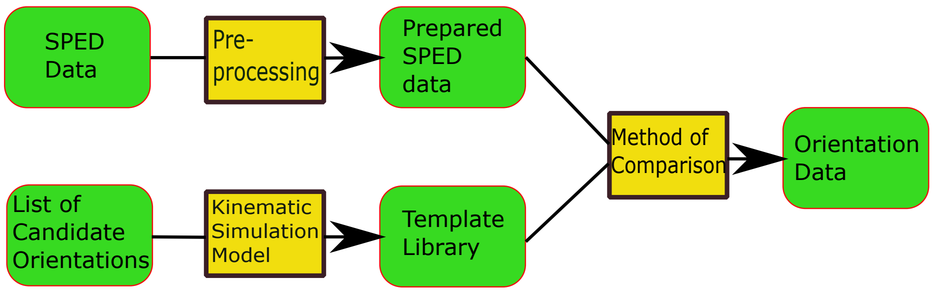


Figure 2.16: The data processing can be visualized as seen above. For a list of candidate orientations and structures we use a kinematic simulation model to generate a template library. SPED data, that has gone through some preprocessing, is then compared to the template library. Through a method of comparison the best fit is chosen to generate orientation data for the examined area on our sample.

2.5.1 Preprocessing

With preprocessing of data there are many sophisticated methods for improving the data quality by operations like for instance removing random noise. The problem with this set of techniques is that we are altering our experimental data. Furthermore, many of the preprocessing algorithms rely on tuning parameters, that are chosen as data is being analyzed. The choise of these parameters can potentially alter the final outcome of the process, and as one typically has an expectation as of what to find in the sample, these parameters could potentially introduce a bias towards the expected result.

We would like to keep our use of preprocessing methods to a minimum. Many of the methods of comparison between the experimental and simulated data do however rely

Table 2.1: The key aspects for each step of data processing. For a more detailed description, see Appendix B.

Steps of data processing
1. Preprocessing • Centering the direct beam for the diffraction pattern.
2. Finding a set of orientations for template library • Generate a list of orientations that span set of possible orientations. • Ensure that orientations that are symmetrically equivalent appear only once. • Ensure that distance between neighbouring orientations is set by the stated resolution.
3. Simulating data from a given orientation • Diffraction patterns are simulated in accordance with kinematic diffraction theory. • The shape function is approximated by a linear function in z-direction.
4. Comparing and correlating • The best fitting template is selected for a given diffraction pattern. • The decision is based on a function for evaluating correlation scores.

on the experimental data being processed to fit the format of the simulated data. For the case of *template matching* methods, it is often required that for every diffraction pattern in the SPED dataset, the direct beam is in the center of the diffraction pattern. This can usually be achieved by shifting the pattern a few pixels in a given direction. While altering the data, this operation conserves all relative intensities and distances, and should not be prone to introducing bias. A great advantage of template matching methods is that it can be used with a minimal amount of preprocessing, and it is therefore a more objective and reproducible set of methods, although the choice of candidate orientations and phases will affect the result.

2.5.2 Template matching

One of the most fundamental means of object detection within an image field is by template matching, in which a single classification function is applied to compare samples from a population with a set of candidate distributions [49][50]. In crystallography, template matching refers to the process where a experimental diffraction image is compared to a set of previously simulated templates by a correlation function [15]. The template giving the highest correlation score with the digital diffraction image is selected as the best match.

Normalized Cross-Correlation

A viable correlation function is the normalized cross- correlation, described by Rauch and Dupuy [15]

$$\gamma = \frac{\sum_{i=1}^{m} f(x_i, y_i) t(x_i, y_i)}{\sqrt{\sum_{i=1}^{m} t^2(x_i, y_i) \sum_{i=1}^{m} f^2(x_i, y_i)}}. \tag{2.51}$$

Here, $f(x_i, y_i)$ is the value of the experimental diffraction pattern at coordinates (x_i, y_i), and $t(x_i, y_i)$ is the value of the simulated diffraction pattern at the same coordinates. It is essential that the templates and the experimental images are properly aligned in the preprosessing step. The normalized cross-correlation holds the advantage that it can be implemented with high computational efficiency, and will provide a correlation score between 0 and 1, where 1 is a perfect fit. For the implementation in *pyxem* [51], the factor $\sqrt{\sum_{i=1}^{m} f^2(x_i, y_i)}$ is omitted for computational speed. This changes the correlation scores so that they are no longer normalized, but does not change which template is selected. While the Normalized Cross-Correlation can be defined to return a score between 0 and 1, it is not invariant under a constant change in image brightness [52]. To provide this invariance, the zero-normalized cross-correlation is used.

Zero-Normalized Cross-Correlation

The correlation function for zero-normalized cross-correlation is given by

$$\gamma = \frac{\sum_{i=1}^{m} (f(x_i, y_i) - \bar{f})(t(x_i, y_i) - \bar{t})}{\sqrt{\sum_{i=1}^{m} (f(x_i, y_i) - \bar{f})^2 \sum_{i=1}^{m} (t(x_i, y_i) - \bar{t})^2}} \tag{2.52}$$

where $\bar{f}$ and $\bar{t}$ are the average intensities of the diffraction pattern and the template, respectively. This correlation metric will return a correlation score between -1 and 1, where 1 is a perfect fit. The zero-normalized cross-correlation is comparable in computational speed to the normalized cross-correlation [53]. For an in-dept analysis of computational performance and precision of an extensive set of correlation functions in real space, the interested reader is referred to [53].

Full-Frame Correlation

Another approach to correlate the experimental and the correlated patterns is to perform the correlation in the frequency domain, by applying the Fourier transform to both the templates and the image [54]. The correlation function is then

$$\gamma_{m,n} = \mathscr{F}^{-1}(\mathscr{F}(f) \odot \mathscr{F}(t)), \tag{2.53}$$

where $\mathscr{F}(f)$ and $\mathscr{F}(t)$ are the Fourier transformations of the image and the template, and $\odot$ is the Hadamard product, the index-wise multiplication between two arrays. The result of the correlation is an array with the same dimensions as the image and the template. Every array entry (m, n) is the correlation score given that the image is shifted by m pixels in the x-direction and n pixels in the y-direction. The selected correlation score is therefore the maximum value of the correlation array, and can be normalized to provide a value between 0 and 1 by dividing by the norms of the template and the image.

One advantage of doing the correlation in Fourier space is that the correlation score function has a broader peak, so that fewer templates are required to obtain a result. This means that upon moving away from the true result, the correlation score decreases more quickly for the Normalized Cross-Correlation and Zero-Normalized Cross-Correlation than for the Full-Frame Correlation. Furthermore, the procedure takes possible misalignment of the image and the template into account. A similar procedure can be used for both drift correction and identifying the position of a template in a larger image. A third advantage is that a refinement of the orientation can be calculated to provide an orientation precision which is substantially better than the precision of library sampling, by applying the duality of the gnomonic projection and assuming that the result is within an angular distance of around 7° from the true value [54]. A disadvantage of the Fourier based approach is that even when the Fast Fourier Transform [55] is used, the real space methods are substantially faster.

2.5.3 Confidence Metrics

As template matching methods select the pattern with the highest correlation score, they will always provide a solution. While a solution is provided, it is not necessarily a good one. As the correlation score determines the quantitative correlation between an image and a template, it is possible that multiple templates have similar correlation scores for a given image.

The Reliability Index

This indistinctness may be quantified by the orientation reliability index, R, defined as [15]

$$R = \left(1 - \frac{\gamma_2}{\gamma_1}\right), \tag{2.54}$$

where γ_1 and γ_2 are the correlation scores for the best and second best fit, respectively. A high reliability index corresponds to a single template receiving a higher correlation score than all other templates, while a lower reliability index signals that multiple templates have similar correlation scores. This metric provides an insight on how sure we are that we have the best match and not the second best match.

The Near Match Similarity Index

While the Reliability Index considers how the highest correlation score relates to the second highest score, it does not take into account the angular distance between the Euler angles related to each correlation score. One approach to address this issue is to compare the list of best matches for a given position with the list of best matches for the images in neighbouring positions in the tilt series. As the rotation between each image is usually relatively small, 3-4 degrees, the list of best matches should have some overlap with neighbouring indices (I do not believe this, if we use for instance 1 degree resolution in orientation library, there should be many orientations that are closer to the best fit than the tilt series neighbour. There could be something I do not understand, though). For a list of the 20 to 30 best matches for a given index, the near-match similarity index for position i, η_i is given by

$$\eta_i = \frac{1}{2}(\#(S_{i-1} \cap S_i) + \#(S_i \cap S_{i+1})), \tag{2.55}$$

where S_i is the set of N best matches for position i, and $\#$ is the *cardinality* of the given set, that is the number of unique members. This metric will provide a high score if neighbouring positions share many top matches, and a low score if the opposite is the case.

The Weighted Average Reliability Index

In order to take into account both the correlation score and the angular distance from the best fit to the respective orientation, a new metric is proposed. The metric is inspired by the fidelity index for EBSD patterns [56]

$$\frac{1}{m}\sum_{k=1}^{m} \cos^{-1}(|\hat{\boldsymbol{v}}_{d,k} \cdot \hat{\boldsymbol{v}}_{c,k}|) \tag{2.56}$$

where $\hat{\boldsymbol{v}}_{d,k}$ is a normalized detected plane normal and $\hat{\boldsymbol{v}}_{c,k}$ is another estimate for the same plane normal. The fidelity index gives the average angle between the detected vectors, and will provide a score close to one if the two estimates have a small angular distance between them. By combining Equations (2.54) and (2.56) a new metric is suggested. The metric is defined by the expression

$$R_w = \frac{1}{N-1}\sum_{i=2}^{N}\left(1 - \frac{\gamma_i}{\gamma_1}\frac{\theta_{i,1}}{\theta_{max}}\right), \tag{2.57}$$

where γ_i is the correlation score of the i-th index, $\theta_{i,1}$ is the angular distance between the orientations corresponding to the best fit and the i-th best fit and θ_{max} is the maximum angular distance between orientations for a given crystal symmetry. For this metric the N highest correlation scores are considered. The metric will provide a value close to 1 if the angles with high correlation scores are aligned with the best fit, and a lower score if orientations with high correlation scores have a larger angular distance to the best fit.

2.5.4 Evaluating Confidence Metrics

A good confidence metric should provide a high score if the result is to be trusted and a lower score if the result is uncertain. For orientation mapping, the angular distance from the true orientation to the obtained orientation can be used as a quality parameter, as a good template matching result is close to the true orientation, while a bad template matching result is further from the true orientation. For a good confidence metric, higher confidence values should coincide with orientations that are closer to the true value. That is, when the confidence metric gives a high value, the result is more likely to be close to the true result. By computing the confidence metrics for a template matching result where the true orientations are known, the correlation between the quality of the result and the confidence metric can be computed. The *correlation coefficient* examines if there is a linear relationship between two distributions, and is defined as [57]:

$$r = \sqrt{\frac{S_{xy}^2}{S_{xx}^2 S_{yy}^2}} \tag{2.58}$$

where

$$S_{xy}^2 = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}), \qquad S_{xx}^2 = \sum_{i=1}^{n}(x_i - \bar{x})^2, \qquad S_{yy}^2 = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

and x and y are the two distributions. By comparing the angular distances from the true result to the values of a given confidence metric, the credibility of the confidence metric can be analyzed.

2.5.5 Visualization of Reliability

While the confidence metrics provide a numerical estimate of the reliability of a result, a more complete picture of the result can be conveyed through visualizations. For a template matching result from a tilt series, the correlation score of the N best matches for every index in the tilt series can be plotted as a color-map in a N by M matrix, where M is the number of indices in the tilt series. This visualization will be referred to as a *correlation score map*, and provides an overview of the correlation scores for all the indices. Another visualization of reliability is plotting orientations from the N best matches of a selected index in an inverse pole figure, with a color gradient corresponding to the correlation score of the given orientation. This visualization conveys information on both angular distance between the best matches and the corresponding correlation scores, but is limited to one tilt series index per plot. This visualization will be referred to as a *IPF index plot*, and conveys information on both correlation scores and angular distance between the best matches.

2.6 Testing the procedure

A common way to obtain crystallographic data is through a tilt series. The sample is tilted a fixed amount around a given axis between subsequent examinations. The tilt angle between subsequent sample entries is therefore constant within experimental deviations, and can be used to examine the precision of the orientation mapping procedure.

In order to determine the precision of an orientation mapping procedure performed on data from a tilt series, the rotation between subsequent orientations is found. A simple test to estimate the quality of the result is to consider the angular distance between subsequent orientations, as discussed in section 2.4.2. If the tilt angle remains close to constant, at a value not far from the expected tilt angle, it would indicate a successful template matching. If the tilt angle varies substantially between different pairs of subsequent orientations, the orientation mapping results are likely to be wrong. This test corresponds to finding the blue area in Figure 2.17, while the area we would like to find is the green area. The green area can be found by adding a further refinement step. Define a density function by letting every rotation that has been labeled as correct give a value +1 to the region of orientation space withing an angle Δ_θ from the rotation. Find the region of the density function in orientation space with the highest value. The value of this region is the number of rotations within the green area in Figure 2.17. The center of this region is now the estimate of μ. If two disconnected regions both contain the max value, the center of the largest region is the estimate for μ. The value of the region in which μ is confined is the number of rotations within the green area in Figure 2.17.

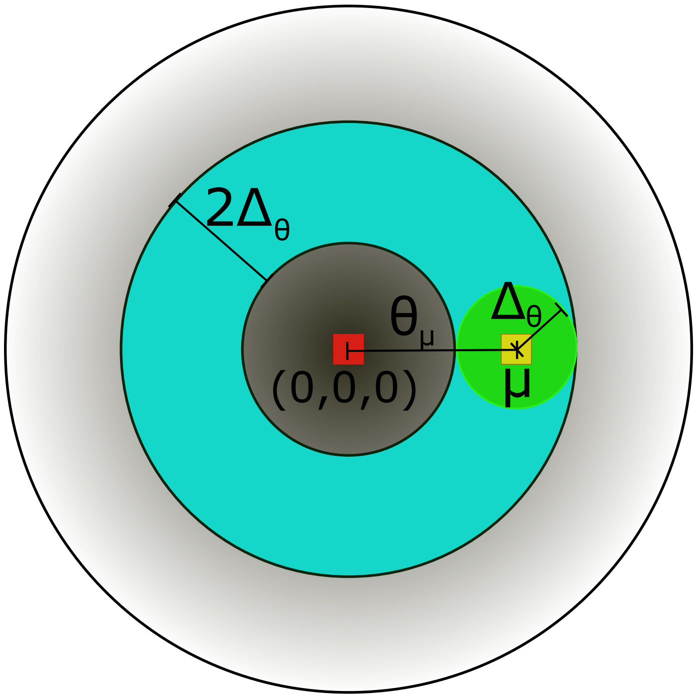


Figure 2.17: Illustration in 2D of the region in orientation space (green circle) around the true rotation estimate μ that should be accepted as correct. The parameter Δ_θ defines the strictness of the test. The true rotation estimate is at a distance θ_μ from the (0,0,0) orientation. Any rotations that is confined within the blue area will be accepted as correct, in the basic model.

As will be discussed in section 2.6.1, random errors in SO(3) should be modelled by the von Mises - Fisher distribution rather than the normal distribution. Computing parameters such as the mean and variance for the obtained angular distances to quantitatively determine the quality of a template matching result is therefore not justified. This test is therefore only meant to provide an indication on whether the template matching has been successful or not, and is not suited to quantitatively analyze and compare different template matching procedures.

2.6.1 The von Mises - Fisher distribution

The Gaussian normal distribution is used to model random errors in the Euclidean space. For directional data distributed on the unit hyper-sphere, the von Mises - Fisher distribution [58] is the analogue of the normal distribution in Euclidean space [59]. The von Mises - Fisher (vMF) distribution is defined by two paramets: The concentration, $\boldsymbol{\kappa}$, and the mean orientation $\boldsymbol{\mu}$.

A d-dimensional unit vector, $\boldsymbol{x}$, is von Mises - Fisher distributed if it has a probability density function

$$f(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\kappa}) = c_d(\boldsymbol{\kappa})e^{\boldsymbol{\kappa}\boldsymbol{\mu}^T\boldsymbol{x}} \tag{2.59}$$

where $c_d(\boldsymbol{\kappa})$ is a normalization constant. $c_d(\boldsymbol{\kappa})$ is defined by

$$c_d(\boldsymbol{\kappa}) = \frac{\boldsymbol{\kappa}^{\frac{d}{2}-1}}{(2\pi)^{\frac{d}{2}}I_{\frac{d}{2}-1}(\boldsymbol{\kappa})} \tag{2.60}$$

where I_r is the modified Bessel function of the first kind, and order r.

The maximum likelihood estimates of $\boldsymbol{\mu}$ and $\boldsymbol{\kappa}$ may be obtained from a set of vMF-distributed orientations with equations [60]:

$$\hat{\boldsymbol{\mu}} = \frac{\sum_{i=1}^{N}\boldsymbol{x}_i}{||\sum_{i=1}^{N}\boldsymbol{x}_i||}, \tag{2.61}$$

and

$$A_d(\hat{\boldsymbol{\kappa}}) = \frac{I_{\frac{d}{2}}(\hat{\boldsymbol{\kappa}})}{I_{\frac{d}{2}-1}(\hat{\boldsymbol{\kappa}})} = \frac{||\sum_{i=1}^{N}\boldsymbol{x}_i||}{N} = \bar{r}. \tag{2.62}$$

Since computing $\hat{\boldsymbol{\kappa}}$ involves an implicit equation (2.62) that is a ratio of Bessel functions, it cannot be solved analytically, and numerical or asymptotic methods must be applied to obtain an approximation.

Two asymptotic approximations can be found when considering $\bar{r}$ [61]. For $\bar{r} \approx 1$ we have

$$\hat{\boldsymbol{\kappa}} \approx \frac{d-1}{2(1-\bar{r})}, \tag{2.63}$$

while for $\bar{r} << 1$ we have

$$\hat{\boldsymbol{\kappa}} \approx d\bar{r}\{1 + \frac{d}{d+2}\bar{r}^2 + \frac{d^2(d+8)}{(d+2)^2(d+4)}\bar{r}^4 + O(\bar{r}^6)\}. \tag{2.64}$$

An iterative approximation can be found using Newtons method [23] and inserting $A'_d(\hat{\boldsymbol{\kappa}})$ as given in [61]

$$\hat{\boldsymbol{\kappa}}_{i+1} = \hat{\boldsymbol{\kappa}}_i - \frac{A_d(\hat{\boldsymbol{\kappa}}_i) - \bar{r}}{1 - \frac{d-1}{\hat{\boldsymbol{\kappa}}_i}A_d(\hat{\boldsymbol{\kappa}}_i) - A_d(\hat{\boldsymbol{\kappa}}_i)^2}. \tag{2.65}$$

An initial estimate of $\hat{\boldsymbol{\kappa}}$ can be found using Equation (2.63).

An estimate of $\hat{\boldsymbol{\kappa}}$ with an empirical touch is provided in [60] to handle the cases falling between the asymptotic limits:

$$\hat{\boldsymbol{\kappa}} \approx \frac{\bar{r}d - \bar{r}^3}{1-\bar{r}^2}. \tag{2.66}$$

Equation (2.66) has the benefit that the number of iterations does not have to be specified, while the domain of use covers the full range of $\bar{r}$.

The axis-angle representation of a set of rotations can be divided into a set of rotation axes and a set of rotation angles to be treated separately. The rotation axes are then analyzed within the von-Mises framework, while the rotation angle distribution is treated with Gaussian statistics.

Chapter 3

Methods

The main objective of this work was to further develop a template matching procedure in the python library *pyXem* and apply it to two experimental RED datasets. The results of the template matching procedure were evaluated to examine how the reliability of the result depends on the selected method for performing the correlation step, and the choice of parameters used to generate a template library.

In order to examine how the template matching procedure performs, a tilt series was simulated for each of the two structures in the experimental RED datasets. The template matching results from the simulated tilt series were analyzed to provide an initial estimate of the precision of the orientation mapping procedure for each of the two structures. Subsequently, the same template matching procedure was performed on the two experimental RED datasets.

3.1 Material and Data acquisition

Experimental series 1 is from GaSb. A GaSb wafer was scratched with a fine diamond scriber and the fine debris material was transferred via a droplet of isopropanol to a 300 mesh Cu grid with a holey amorphous C-film (ca. 20 nm thick). This preparation procedure is an adaption of the routine described in [62]. Data were collected by Tom Willhammar and Hongyi Xu using a DM-script (unpublished data and script) on a JEOL JEM-2100 instrument, operated 200 kV, using a 2k Gatan Orius camera placed above the viewing screen. The series consist of 30 frames (512x512 pixels), in the range -29.4 degrees to 27.99 degrees tilt, with a 1.98 degree tilt step. The second experimental series, series 2, is a RED set of orthorhombic zeolite mordenite, published on Zenodo [63]. Details of the experimental settings are given in M. O. Cichocka et al [64]. The data sets consist of 395 frames (516x516 pixels), from -43.9 degrees to 58.65 degrees, with a 0.2336 degree tilt step. As every tenth frame was extracted, the tilt step in the analyzed dataset was 2.336 degrees. GaSb has space group 216 and Laue Group $m\bar{3}m$ [65]. The crystallographic information file used for GaSb is derived from [66]. Mordenite has space group 63 and Laue Group mmm [65]. The crystallographic information file used for mordenite is derived from [67].

3.2 Template matching for simulated data

In order to simulate RED data, a tilt series was defined by choosing an initial orientation, a tilt axis and a constant tilt step. For every orientation in the tilt series a diffraction pattern was simulated. For the simulation the crystal structure, the energy of electrons in the electron beam, the maximum excitation error and the reciprocal radius were the input parameters. Furthermore a spread parameter was defined, that determines how broad the Gaussian distribution of a single reflection should be. The selected values for the 2 materials studied are summarized in Table 3.1.

Table 3.1: Parameters used to simulate data.

	GaSb	Mordenite
Tilt Angle ($^\circ$)	4	4
Tilt Axis	[1,1,1]	[1,1,1]
Energy (keV)	200	200
Maximum Excitation Error ($Å^{-1}$)	0.025	0.025
Reciprocal Radius ($Å^{-1}$)	2	0.89
Spread ($Å^{-1}$)	0.03	0.008

To confirm that the methods were correctly implemented, and that sensible parameters had been chosen for the simulated data a template match was run with a template consisting of only orientations in the tilt series. With such a template bank, every entry in the simulated data would correspond to a template in the template bank. The procedure should therefore be able to correctly index every entry in the simulated data. Due to a round-off error close to the edges of diffraction patterns for the simulated data in pyXem, the reciprocal radius of the template patterns was chosen slightly smaller than that of the simulated patterns.

After the initial analysis, a new template bank was generated, by having orientations evenly distributed on the *fundamental zone*, with a given maximum angular distance between neighbouring orientations. The fundamental zone, also known as the fundamental sector, is the sector describing the entirety of all representative orientations, and is defined by a size and shape which is often uniquely determined by the respective point group symmetry [48]. The functionality related to generating the orientation grid on the fundamental zone was tested in a previous project[68], and it has been confirmed that recent changes to the functionality have overcome the issues reported in the previous project[1]. From the evenly spaced grid of orientations, a set of templates were generated, using the same crystal structure, electron energy, maximum excitation error and reciprocal radius as for the simulated data. For the templates the convergence angle is not considered and no Gaussian spread is applied so a reflection consists of a single pixel. The simulated data was compared to the template bank using each of the three correlation metrics; Normalized

[1]The discussion if found in PR60 - Pyxem/Diffsims and PR47 - Pyxem/Diffsims on GitHub

Cross-Correlation (NCC), Zero-Normalized Cross-Correlation (ZNCC) and Full-Frame Correlation (FFC). For NCC and ZNCC the template bank had a resolution of 1 degree, while the template bank had a resolution of 3 degrees for FFC. The limited resolution of the FFC method was due to the extensive memory usage from a Fourier transformed template library. For mordenite 3 degrees was the highest resolution obtainable on the high performence computer, with 150 GB memory.

In addition to the tilt series, a set of 1000 random orientations for GaSb and 300 random orientations for mordenite were generated for each of the three correlation methods. Using the parameters in Table 3.1 a signal was simulated and matched against the template libraries with the same resolution as for the simulated tilt series. The angular distance from the true value was determined for each random orientation. By defining every orientation within an angular distance of 3 degrees from the true result as correct and every orientation further than 3 degrees from the true result as misindexed the each correlation method got a score for percentage of correctly indexed orientations, which was used to evaluate the correlation methods. The standard deviation of the correctly indexed orientations was computed.

As the true orientation is not known for experimental patterns, another test was made, based on the distance between orientations. Two integers, int1 and int2 were drawn from the pseudo-random number generator random.randomint in the standard python library random, so that both int1 and int2 were between 0 and the number of random orientations. The distance between the true orientation at index int1 and the true orientation at index int2 was found. Then the distance between the corresponding template matching orientations was computed. The difference between the true distance, and the distance between template matching orientations was computed. If this distance was smaller than 1.8 degrees, the result was defined to be correct. The distance of 1.8 degrees was selected to be as large as possible, without including the 0 rotation, when applied to the tilt series of GaSb and mordenite, with 1.98 and 2.3 degree tilt steps.

In order to evaluate the confidence metrics, the angular distance distribution was compared to the weighted average reliability index scores (WARI) and the reliability index scores (RI) using Equation (2.58). Equation (2.58) was also used to analyze if the total intensity of a signal would affect the chances for a correct indexation by comparing the distribution of total intensities with the angular distance distribution.

The resulting orientations and the rotations between subsequent orientations were plotted in the inverse pole figure using MTEX [69], as discussed in Section 2.4.4. Furthermore, the angular distance from the best fit to the true value was computed, and the angular distance between subsequent entries in the tilt series was computed, as discussed in Section 2.6. The confidence metrics for the results, that is the reliability index and the weighted average reliability index, were computed as discussed in Section 2.5.3, and a correlation score map was generated, as described in Section 2.5.5. By analyzing the distance to the true value and the scores for different confidence metrics, certain entries in the tilt series were selected for further analysis. The selected entries were analyzed using an IPF index plot, discussed in section 2.5.5, and by visually comparing individual templates to the simulated signal.

3.3 Template matching for experimental RED data

The first step in analyzing the experimental RED datasets was to center the direct beam for every frame. The parameters used to center the direct beam are summarized in Table 3.2. The centering was verified manually, by plotting the signal and confirming that the direct beam was centered. For the mordenite dataset, the center direct beam function was run two times with the same input parameters. The number of pixels in every frame was reduced for computational efficiency, by cropping regions with no visible reflections of the mordenite dataset. For the mordenite dataset, a separate version of the data was created for the Full-Frame correlation by applying the rebin function for signals in pyxem. This function reduces the numbers of pixels in each frame merging multiple pixels into a single pixel while summing their intensities. This was done to reduce the memory demands of running the Full-Frame algorithm, and reduced the dimensions of the cropped dataset from (357x357 pixels) to (128x128 pixels).

Table 3.2: Parameters for center direct beam.

	GaSb	Mordenite
Method	Cross-correlate	Cross-correlate
Maximum radius	10	10
Minimum radius	1	1

In order to create a template bank, a list of candidate orientations on a fundamental zone grid were generated, by specifying a crystal symmetry and an angular resolution, where the angular resolution is the maximum distance between neighbouring orientations. A template bank was generated from the list of candidate orientations after specifying the lattice parameters and atomic coordinates from the cif file, the energy of electrons in the electron beam, the maximum excitation error and the reciprocal radius. As the maximum excitation error is a parameter that has a huge impact on the number of reflections in a template bank, while being implemented in a rather nonphysical way in the current version of pyXem, some fine tuning had to be done in order to arrive at a sensible value, where the number of reflections resemble that of the dataset to be analyzed. The interval of values that were considered for the maximum excitation error was [0.025 - 0.055]. The relevant experimental parameters are included in Table 3.3 for the two experimental data sets. For NCC and ZNCC the library resolution was 1 degree, while the FFC library had a 3 degree resolution.

Table 3.3: Experimental parameters.

	GaSb	mordenite
Maximum excitation error	0.025	0.025
Calibration	1.01	1.01
Reciprocal Radius (Å^{-1}	2	0.89
Beam Energy (kev)	200	200

The comparison between the experimental signal and the templates was done for all three correlation metrics, providing three results for each dataset. The resulting orientations and the rotations between subsequent orientations were plotted in the inverse pole figure and the angular distance between subsequent orientations was found. The confidence metrics for the results were computed, and the correlation score map was generated. Based on the results from the confidence metrics, inverse pole figure plots and the angular distance between subsequent orientations, certain indices were selected for further detailed analysis, using IPF index plots and and plotting high-scoring templates on the signal.

3.4 Hard- and Software

All data analysis was performed in the developer version of pyXem 0.12.0 [51] downloaded 30.05.2020 on a branch where the Full-Frame Correlation method is under development. A description of the used code is given in a tutorial style in Appendix B.

For plotting of Inverse Pole Figures, MTEX version 5.3 [69] was used for MATLAB R2020a.

Workflows were run on a HP laptop 15db0xxx, with a AMD Ryzen 5 processor and 8 GB RAM. MATLAB scripts were run on Microsoft Windows 10, and Python scripts were run on Ubuntu 18.04.4 LTS in the integrated development environment (IDE) *Oracle VM VirtualBox*.

For Template matching workflows with a large (more than 100,000 library entries) template library the NTNU IDUN computing cluster [70] was used. The cluster has more than 70 nodes and 90 GPGPUs. Each node contains two Intel Xeon cores, at least 128 GB of main memory, and is connected to an Infiniband network. Half of the nodes are equipped with two or more Nvidia Tesla P100 or V100 GPGPUs. Idun's storage is provided by two storage arrays and a Lustre parallel distributed file system.

Chapter 4

Results

4.1 Simulated GaSb

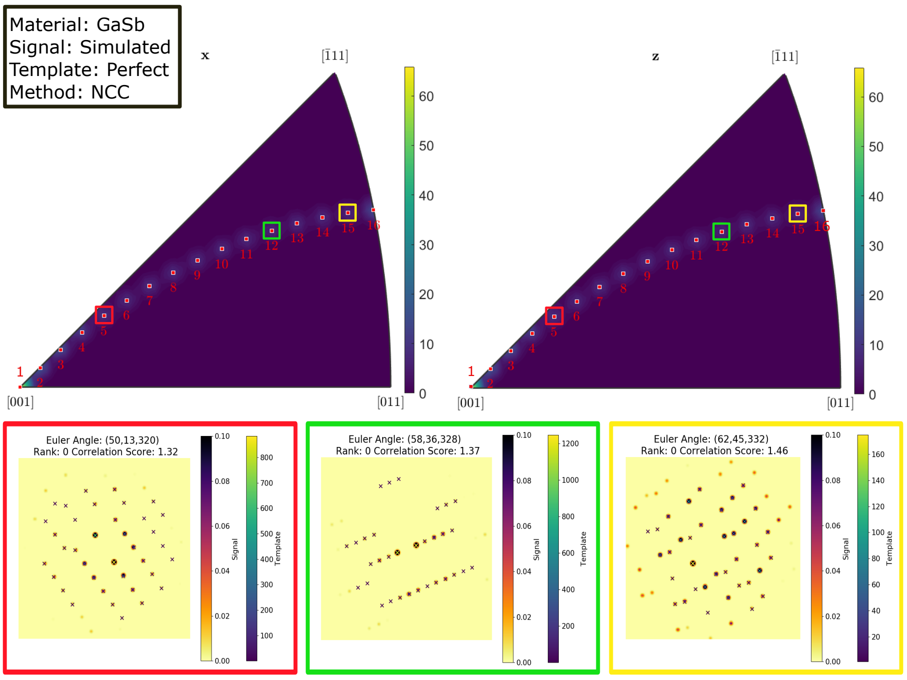
4.1.1 Perfect Template Bank

The orientations in the tilt series were used to generate a template library, and the simulated data was matched to the template library, hence there should be a perfect match for every orientation in the created library. The inverse pole figures of the template matching orientations and the rotations between subsequent template matching orientations are shown in Figures 4.1, 4.2 and 4.3 for NCC, ZNCC and FFC, respectively. The template matching results for orientations 1 to 16 in the tilt series are shown in the **(a)** panels. Orientations 16 + n coincide with orientations 16 - n for the inverse pole figures in both x and z directions. Orientations 1 to 16 were selected because a plot of 30 orientations would be too crowded. This choice will be the discussed later.

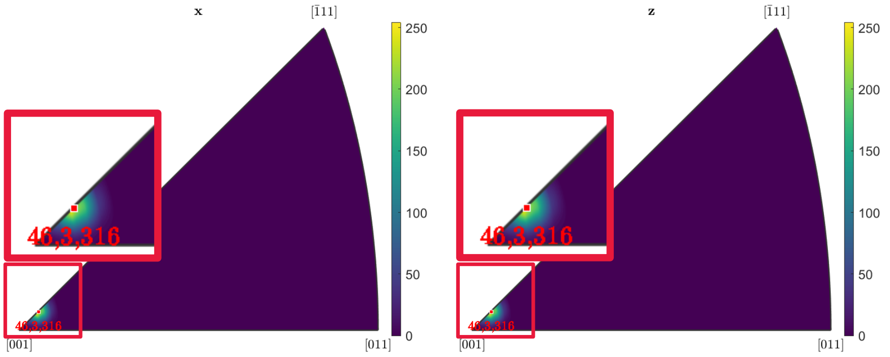
Therefore, For every orientation a unimodal ODF function was generated with a half width of one degree and a weight determined by the correlation score of the orientation. The summed unimodal ODF functions for orientations 1 to 16 gave rise to the intensity of the orientations, while the red squares show the exact position of every orientation. The colorbar represents the intensity of the summed unimodal ODF functions. The red numbers below every orientation are the positions in the tilt series. The colored rectangle insets show the template plotted together with the signal for selected orientations in the tilt series. In every insert the Euler angle of the template, the correlation score and the rank of the template are reported, where the rank of the highest scoring template is 0, the second highest scoring template is 1, and so on. Every insert has two colorbars, one representing the signal intensities and one representing the template intensities. In the **(b)** panels of Figures 4.1 4.2 and 4.3 the rotations between subsequent orientations in the tilt series are shown, for orientations 1 to 30. For every rotation an unimodal ODF function was generated with a half width of one degree and a constant weight of 1. The summed ODF functions give rise to the intensity function on the IPF, and the colorbar represent the summed intensity of the ODF functions. Every rotation is plotted as a red square. The red numbers are the Euler angle of the rotations in the Bunge Convention. The big red frame shows a magnified version of the motif in the smaller red frame. In this specific tilt series (see Table 3.1) the tilt axis was [1,1,1] and the tilt angle was 4 degrees.

For NCC and ZNCC, the orientations seen in Figures 4.1a and 4.2a form a smooth line, and the templates for selected orientations perfectly fit the signal. The rotations seen in Figures 4.1b and 4.2b are all stacked in one position on the inverse pole figure. For FFC the orientations in Figure 4.3a are perfectly aligned, apart from an interchange of the 9th and 10th orientation. In the green and yellow rectangles corresponding to the 9th and 10th orientation, respectively, the templates do not match the signal, even though a high correlation score is obtained. In Figure 4.3b, the rotations are stacked in two positions on the IPFs. One position contains the projection of Euler angles 46,3,316 and 224,3,134, while the other position contains the Euler angle 47,7,317. The 46,3,316 rotation is recognized from Figures 4.1b and 4.2b, as corresponding to the correct rotation, while 224,3,134 and 47,7,317 are not seen in either of Figures 4.1b and 4.2b.

The angular distance from the template matching orientation to the true orientation, and the angular distance between subsequent template matching orientations are shown in Figure 4.4. For NCC and ZNCC, the correct orientation is obtained at every index, as seen in Figures 4.4a and 4.4b. Furthermore, the rotation angle between orientations is constant at the expected value, as seen in Figures 4.4d and 4.4e. For FFC, template matching orientations 9 and 10 are at a distance 4 degrees from the true orientation, as seen in Figure 4.4c. For a perfect result every orientation should be at a distance 0 from the true value, as seen in Figures 4.4a and 4.4b. The rotation angle between orientations differ from the expected value of 4 degrees at indices 9 and 11 as seen in Figure 4.4f.

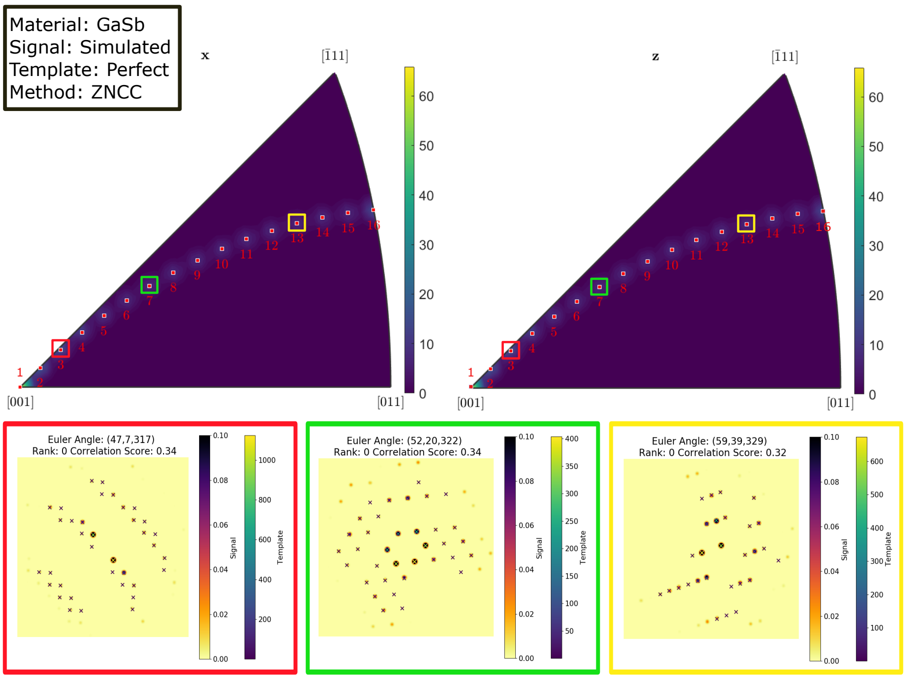


(a)

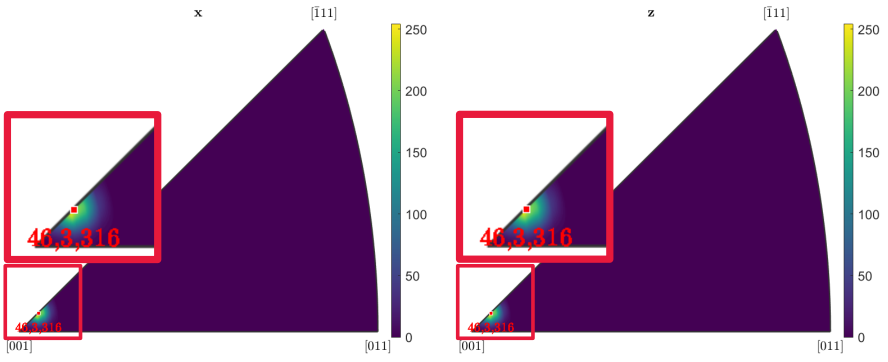


(b)

Figure 4.1: Inverse pole figures for NCC on a simulated GaSb tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

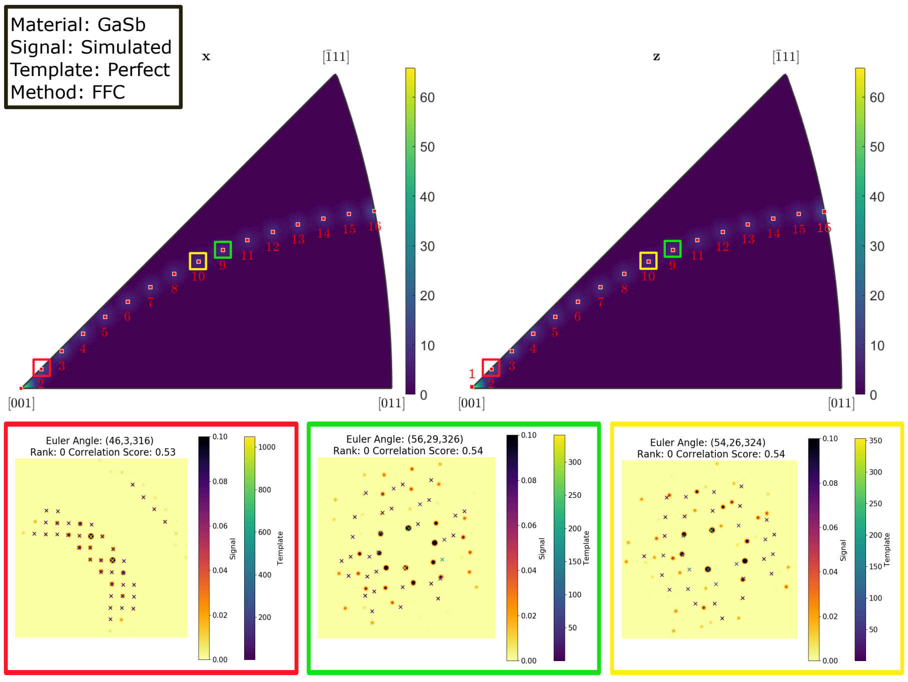


(a)

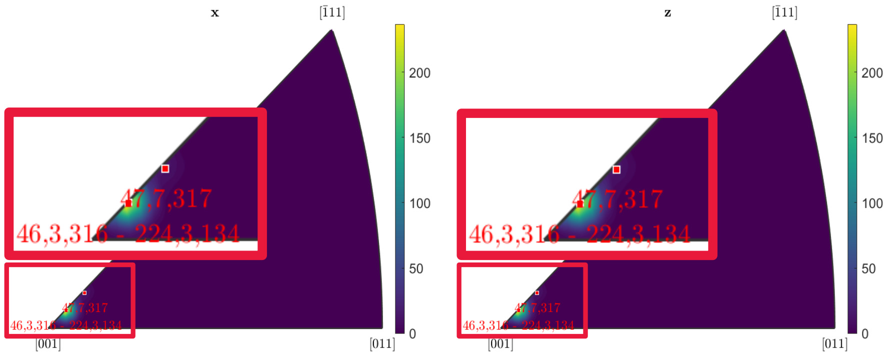


(b)

Figure 4.2: Inverse pole figures for ZNCC on a simulated GaSb tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

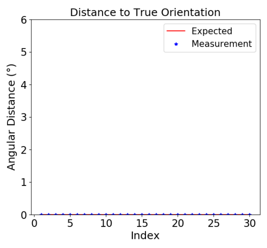


(a)

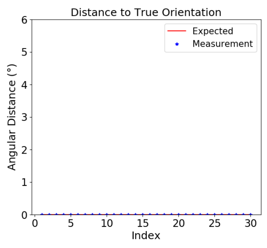


(b)

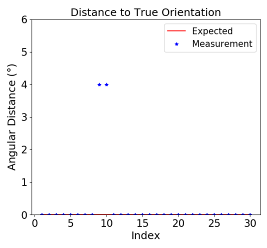
Figure 4.3: Inverse pole figures for FFC on a simulated GaSb tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.



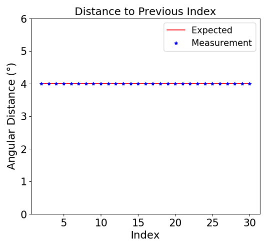
(a) NCC



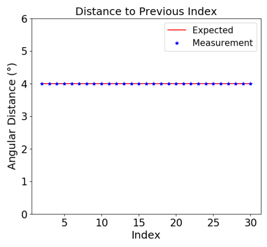
(b) ZNCC



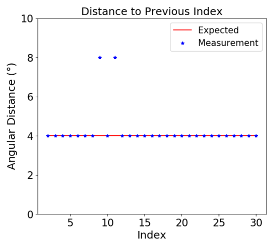
(c) FFC



(d) NCC



(e) ZNCC



(f) FFC

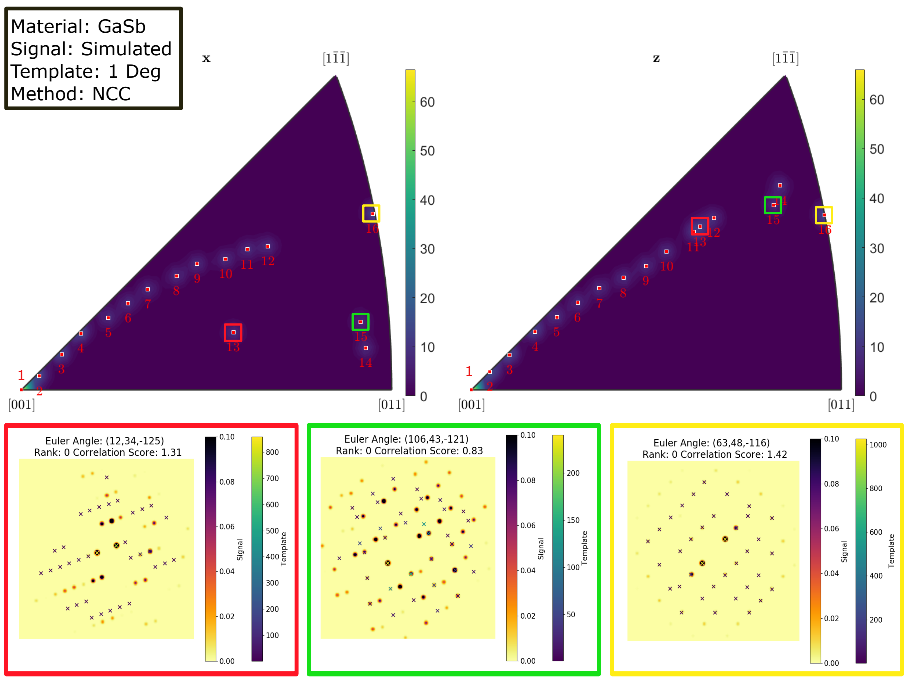
Figure 4.4: Distance of template matched orientation to true orientation for GaSb with a perfect template input for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC for 30 simulated orientations. Angular distance to the previous orientation in the template matching result for **(d)** NCC, **(e)** ZNCC and **(f)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.

4.1.2 1 Degree Grid

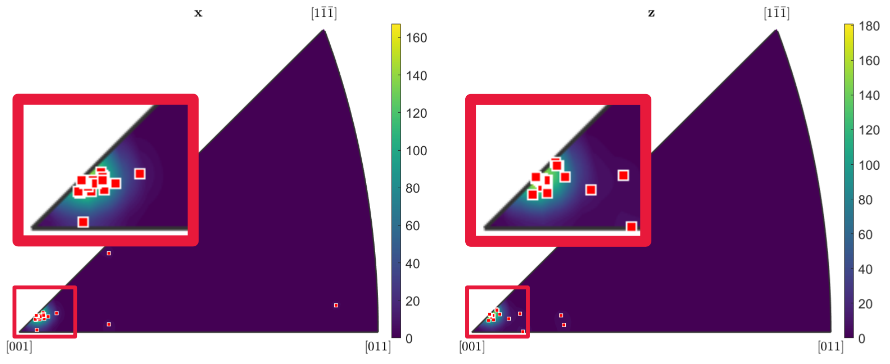
A template library was generated by creating an equally spaced grid on the fundamental zone, and the simulated data was matched against the template library. The grid spacing was 1 degree for NCC and ZNCC and 3 degrees for FFC. The inverse pole figures of the template matching orientations and the rotations between subsequent template matching orientations are seen in Figures 4.5, 4.6 and 4.7 for the methods NCC, ZNCC and FFC, respectively. For Figure 4.5a template matching orientations 1 to 12 are aligned on the inverse pole figure for both the x- and z- projections. Indices 13,14 and 15 are not following the trend from orientations 1 to 12. For the selected indices, the template coincides with the signal for a few high intensity positions for the red and green rectangles corresponding to indices 13 and 15, but the template does not match the signal. For index number 16 the template coincides perfectly with the signal. In Figure 4.5b, there is

The angular distance from the obtained orientation to the true orientation and the angular distance between subsequent indices are shown in Figure 4.8. NCC predicts orientations that are close to the true value for most indices, but fails for indices 13, 14, 15 and 25. ZNCC also predicts orientations that are generally close to the true value, but fails for 11 different indices, as seen in Figure 4.8b. For FFC, the predicted orientations are not close to the true value, and Figure 4.7a does not resemble a tilt series. The reliability index, the weighted reliability index and the correlation score maps are seen in Figure 4.9.

The reliability metrics for the template matching results are found as described in Section 2.5.3. By comparing the reliability metrics with the distance to the true orientation, the reliability metrics can be evaluated.

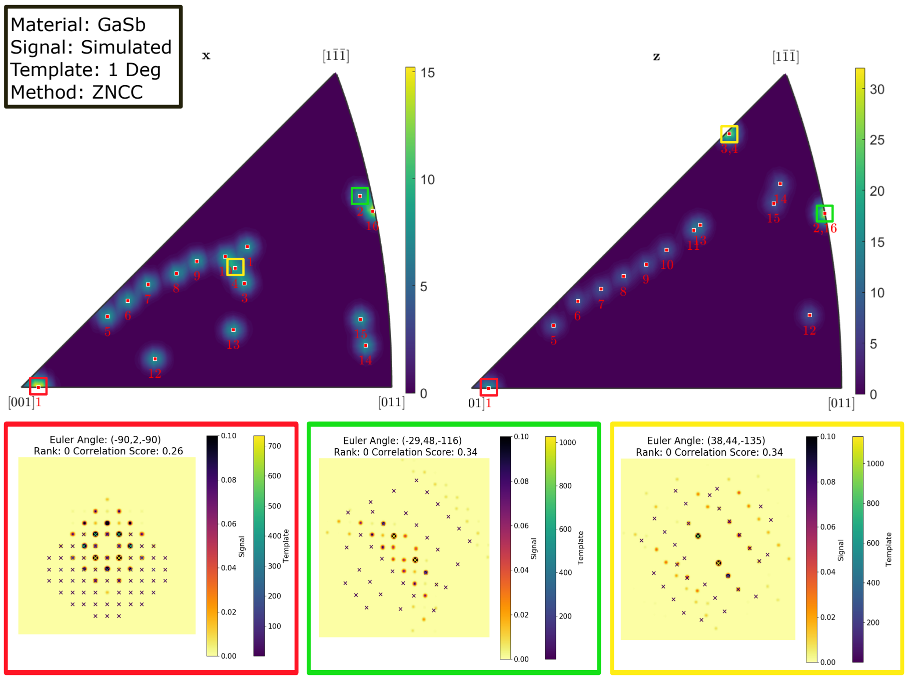


(a)

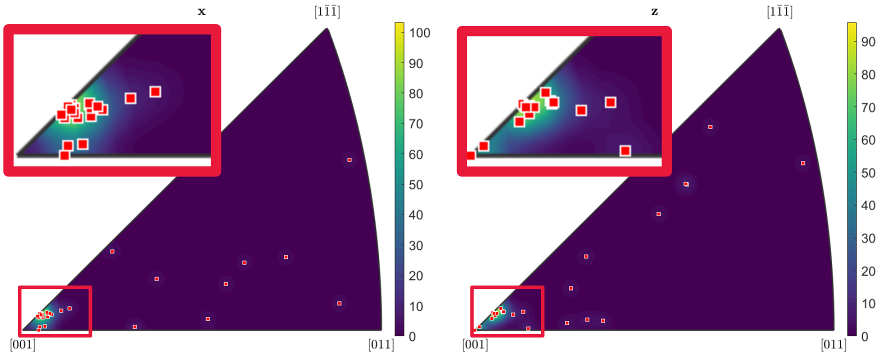


(b)

Figure 4.5: Inverse pole figures for NCC on a simulated GaSb tilt series with a template bank of 1 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

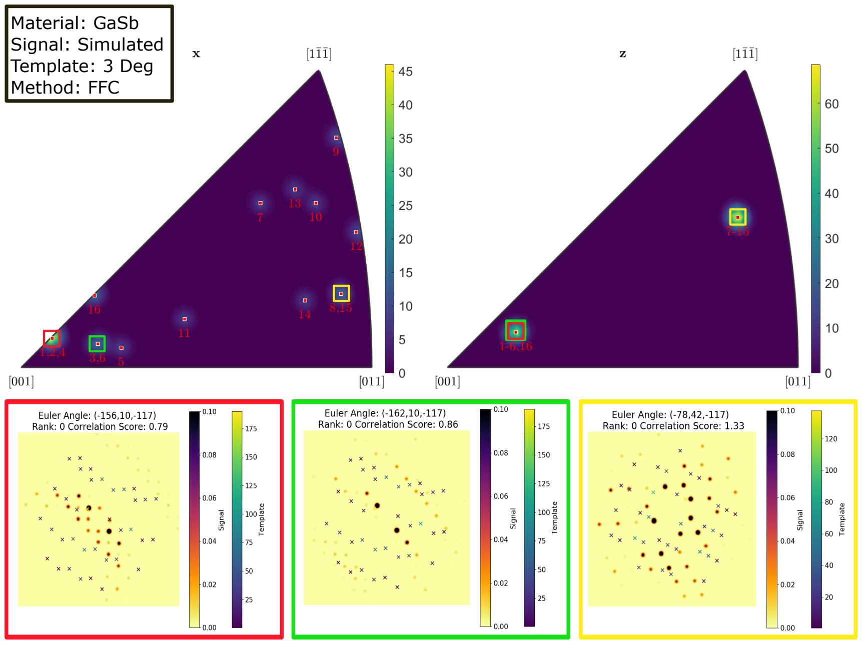


(a)

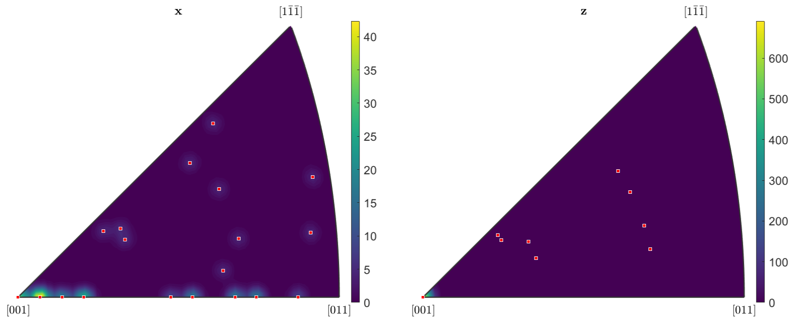


(b)

Figure 4.6: Inverse pole figures for ZNCC on a simulated GaSb tilt series with a template bank of 1 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

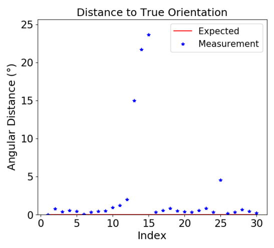


(a)

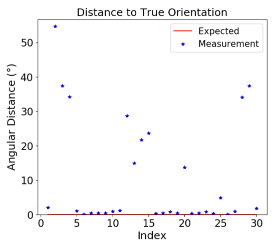


(b)

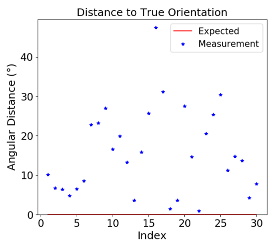
Figure 4.7: Inverse pole figures for FFC on a simulated GaSb tilt series with a template bank of 3 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.



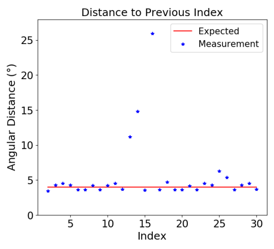
(a) NCC



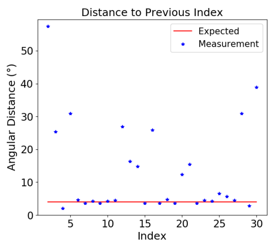
(b) ZNCC



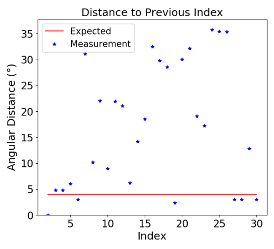
(c) FFC



(d) NCC

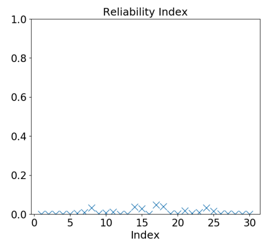


(e) ZNCC

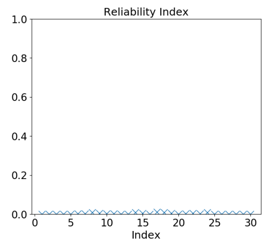


(f) FFC

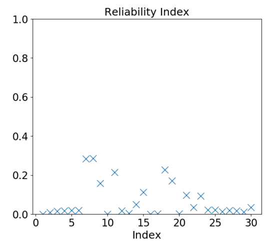
Figure 4.8: Distance of template matched orientation to true orientation for GaSb with a template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC for 30 simulated orientations. Angular distance to the previous orientation in the template matching result for **(d)** NCC, **(e)** ZNCC and **(f)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.



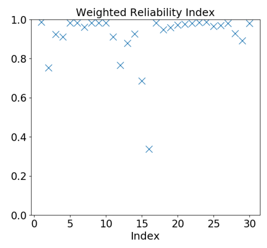
(a) NCC - Reliability Index



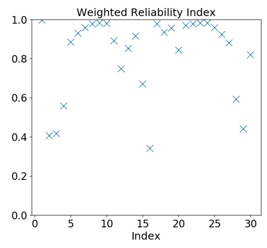
(b) ZNCC - Reliability Index



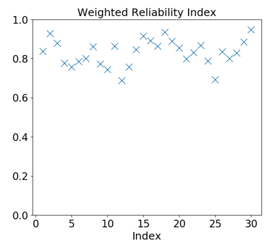
(c) FFC - RI



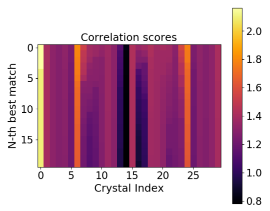
(d) NCC - WRI



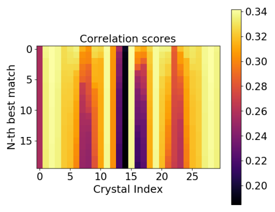
(e) ZNCC - WRI



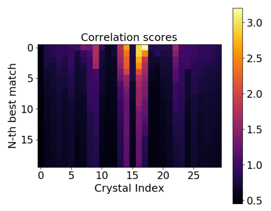
(f) FFC - WRI



(g) NCC - Correlation score-map



(h) ZNCC - Correlation score-map



(i) FFC - Correlation score-map

Figure 4.9: Top row: Reliability index for GaSb with a template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for 30 orientations in a simulated tilt series for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC. A higher score indicates a higher reliability as defined in Equation (2.54) . **Middle row:** Weighted reliability index for correlation methods **(d)** NCC, **(e)** ZNCC and **(f)** FFC. A higher score indicates a higher weighed reliability as defined by Equation (2.57), where 1 is the highest score. **Bottom row:** 20 Best correlation scores represented in a map for every orientation in the tilt series for the correlation methods **(g)** NCC, **(h)** ZNCC and **(i)** FFC, with color representing the correlation score.

4.1.3 Random Orientations

The inverse pole figure for NCC applied to a set of 1000 random orientations for simulated GaSb, with a template bank of 1 degree resolution. A local averaging algorithm was applied to the orientations to generate a smooth result and identify regions where many orientations are misindexed. The colorbar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed, while a bright area is an area where the average distance to the true value is higher. The insets show the signal together with the best fitting templates for selected positions on the IPF. Every inset contains the Euler angle for the template, the correlation score and the rank of the template. The colorbars represent signal intensity and template intensity. **(a)** Shows the x-projection of the inverse pole figure, while **(b)** shows the z-projection of the inverse pole figure. — The angular distance from the obtained value to the true orientation for the NCC method. **(a)** Shows the x-projection. **(b)** Shows the z-projection.

A new simulated dataset was generated by drawing 1000 random orientations, and simulate data with an Energy, Max excitation error, reciprocal radius and spread as defined in Table 3.1. A template library was generated from an equally spaced grid on the fundamental zone, where the grid spacing was 1 degree for NCC and ZNCC and 3 degrees for FF. The distance from the true result to the template matching result was computed. Figures 4.10, 4.11 and 4.12 show the angular distance to the true value for the x and z directions of the Inverse Pole Figure. A local averaging algorithm was applied to the set of 1000 orientations to generate the figures.

By defining a correctly indexed orientation as an orientation that has an angular distance of less than 3 to the true orientation all orientations were divided into correct and wrong results. The correct results were further analyzed by computing the standard deviation, with a zero mean, as zero represents the true orientation. The results are presented in Table 4.1.

Table 4.1: Fraction of correctly indexed orientations for the correlation methods NCC, ZNCC and FFC, computed by defining a template matching result within 3 degrees from the true value to be correct. The standard angular deviation from the true result is reported

Method	Correctly indexed (%)	Standard Deviation (°)
NCC	60.1	0.89
ZNCC	55.8	1.06
FFC	3.1	1.66

For the same data, pairs of random orientations were drawn. The distance between the true value of the orientations was found. The distance between the corresponding template matching orientations was computed, and compared to the true distance. A deviation of less than 1.8 degrees was taken to be correct. The result of the test is presented in Table 4.2.

By comparing the angular distance to the score of the reliability index and the weighted reliability index, the correlation coefficients for RI and WRI were computed. In order to test if there were a correlation between the total intensity of the simulated signal and the indexation error, the correlation coefficient between the total signal intensity and the

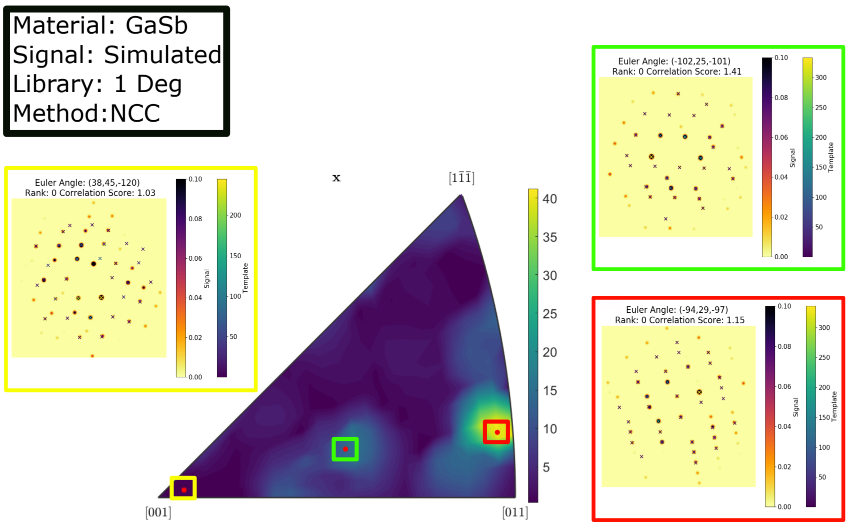
Table 4.2: Fraction of correctly indexed orientations for the correlation methods NCC, ZNCC and FFC, computed by defining a distance between template matching orientations within 1.8 degrees from the true distance to be correct.

Method	Correctly indexed (%)
NCC	47.3
ZNCC	42.6
FFC	10.0

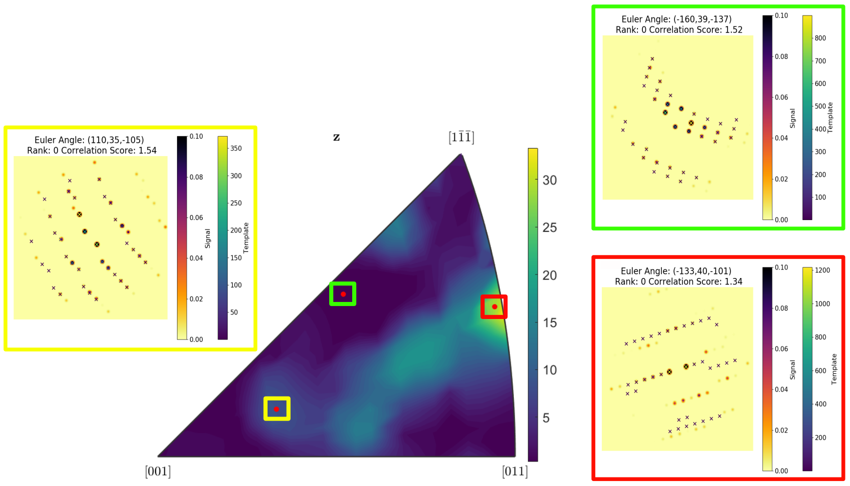
angular distance to the true value were computed.

Table 4.3: Correlation coefficients - Distance to true value vs Reliability Metric

Method	NCC	ZNCC	FFC
WRI	-0.281	-0.428	-0.372
RI	-0.137	-0.184	-0.084
I_{tot}	-0.096	-0.133	0.001

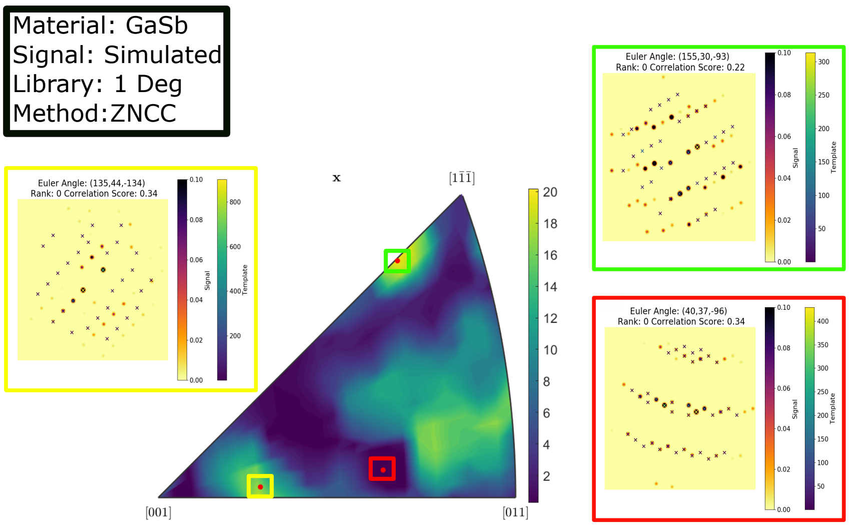


(a)

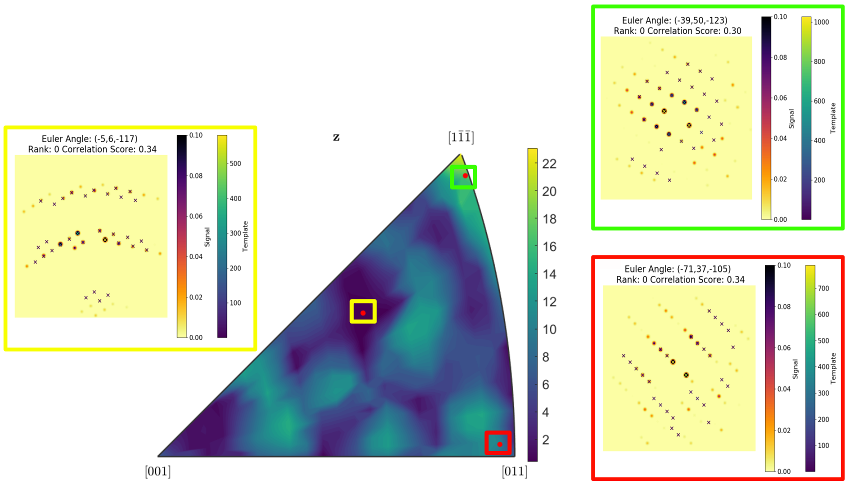


(b)

Figure 4.10: Inverse pole figure for NCC to a set of 1000 random orientations for simulated GaSb, with a template bank of 1 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.

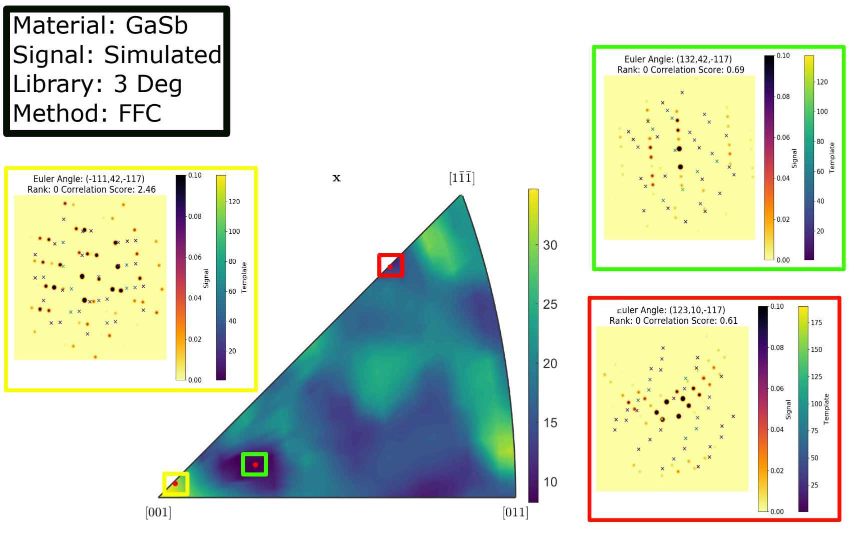


(a)

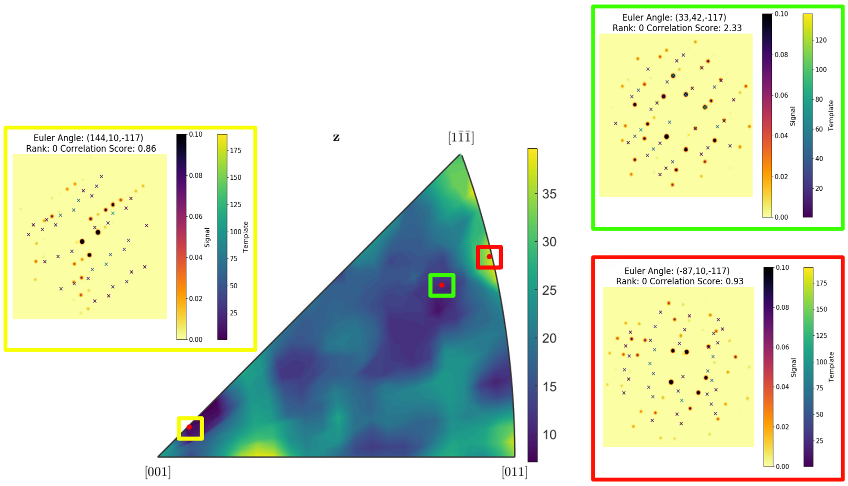


(b)

Figure 4.11: Inverse pole figure for ZNCC to a set of 1000 random orientations for simulated GaSb, with a template bank of 1 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.



(a)



(b)

Figure 4.12: Inverse pole figure for FFC to a set of 1000 random orientations for simulated GaSb, with a template bank of 3 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.

4.2 GaSb Dataset

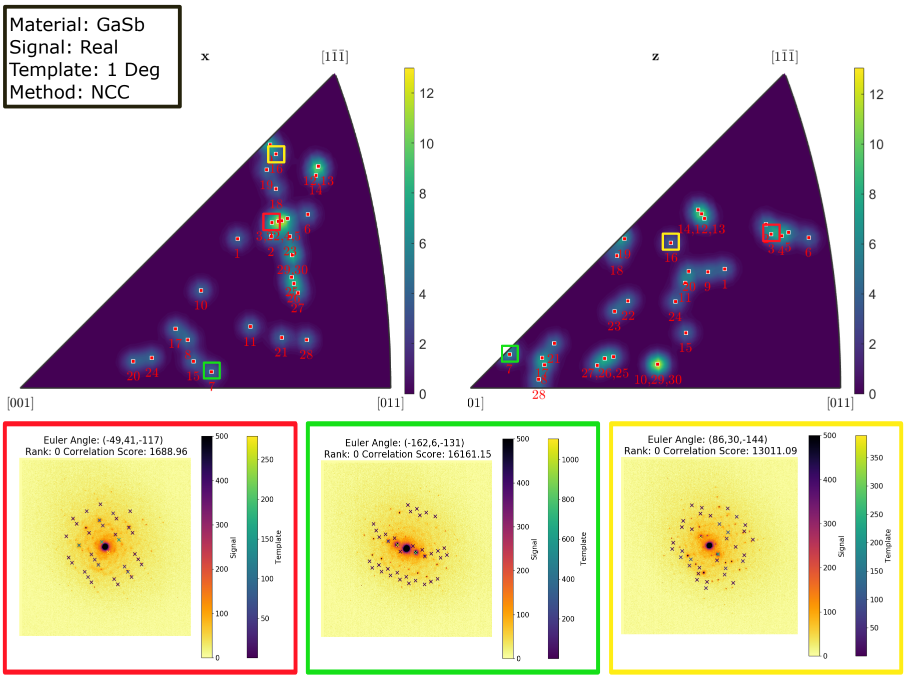
A template library was generated by creating an equally spaced grid on the fundamental zone, and the GaSb dataset was matched against the template library. The inverse pole figures of the resulting rotations and orientations are shown in Figures 4.13, 4.14 and 4.15.

While some orientations are aligned in the inverse pole figure for NCC, it is very hard to see a tilt series. By considering the selected templates plotted together with the signal, it would seem like a few, high intensity template positions correspond well with high intensity positions in the signal, while most template positions do not correspond to an intensity in the signal. For the rotations, there is a noticeable concentration of rotations around a small area in the inverse pole figure. There are also many rotations scattered over the IPF far from the area where the concentration of rotations is higher.

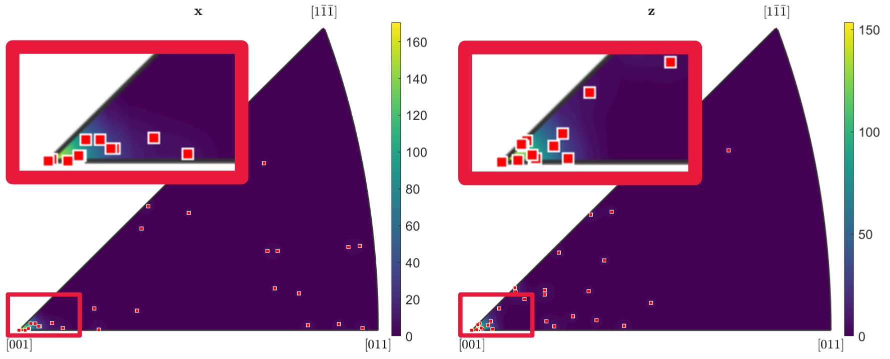
The results for the ZNCC, seen in Figure 4.14 are similar to those of NCC. The indices 2-6 seem to be aligned well, and there are other sequences of indices that seem to be aligned, but in general the result does not resemble a tilt series. For the templates plotted together with the signal, only the green rectangle, corresponding to index 19 looks correctly indexed. For the red and yellow rectangles, the template coincides with the signal in a some positions, but there are also many signal reflections that do not correspond to a template entry. Figure 4.14b shows many rotations that are spread over the inverse pole figure, but there is a region with higher intensity close to the [001] corner. By considering Figure 4.16, we observe that there are a number of rotations with a rotation angle not far from the expected oscillation angle of 1.98 degrees, reported by the group that collected the data.

For FFC, seen in Figure 4.15 the orientations are spread over the inverse pole figure for the x-projection, and concentrated in 4 positions on the z-projection. The templates plotted together with the signal do not resemble the signal they are fitted to. The rotations are also spread over the inverse pole figure for the x-projection, while the z- projection has a density of rotations close to the [001] corner of the inverse pole figure, and 14 rotations spread along a line at a distance from the [001] corner. In Figure 4.16 only a few rotations have rotation angles that are close to the expected value.

Considering the reliability indices in Figure 4.17, we observe that the scores for the Weighted reliability indices now provides lower values in general for all correlation methods. The reliability index is close to 0 for most indices, and the Correlation score maps resemble what we have seen in Figure 4.21.

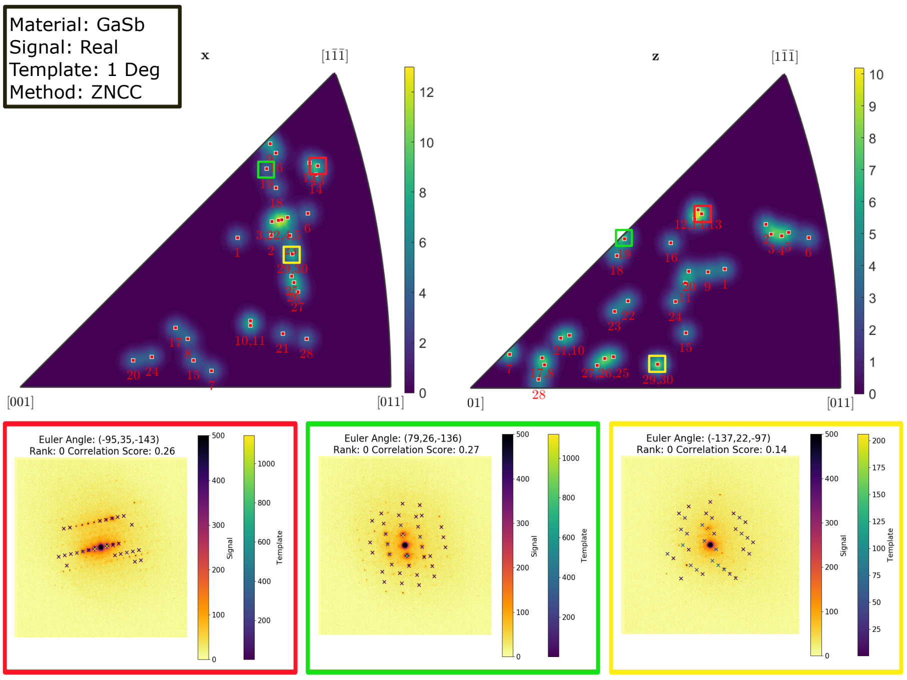


(a)

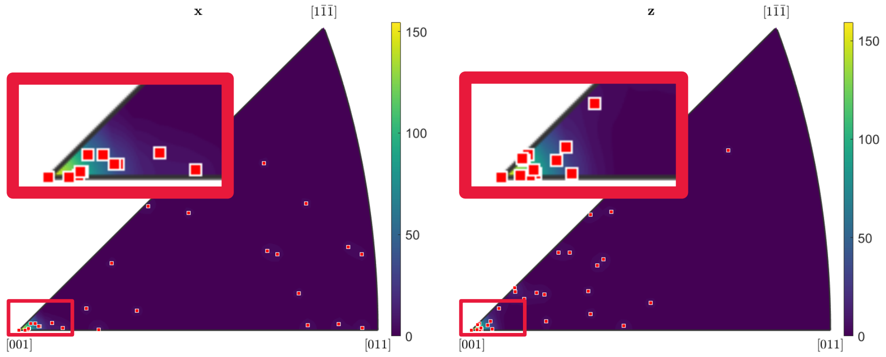


(b)

Figure 4.13: Inverse pole figures for NCC on an experimental GaSb tilt series with a template bank of 1 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

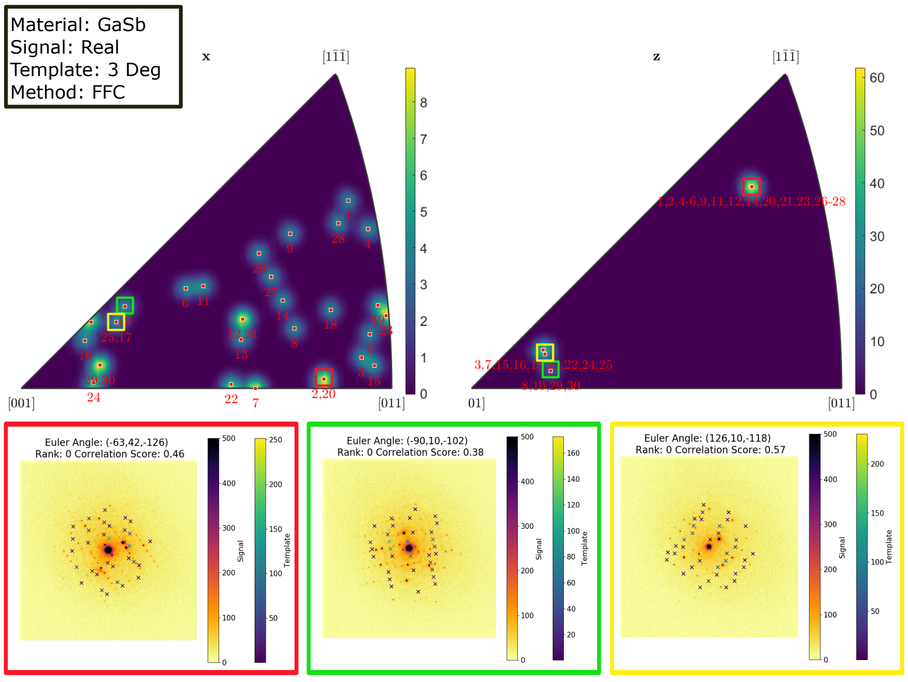


(a)

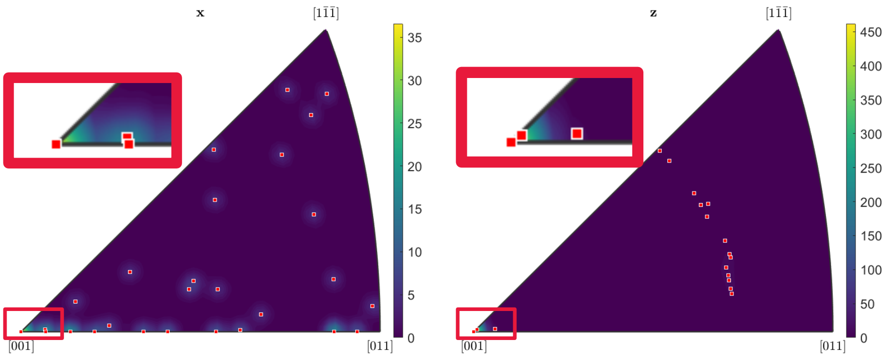


(b)

Figure 4.14: Inverse pole figures for ZNCC on an experimental GaSb tilt series with a template bank of 1 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

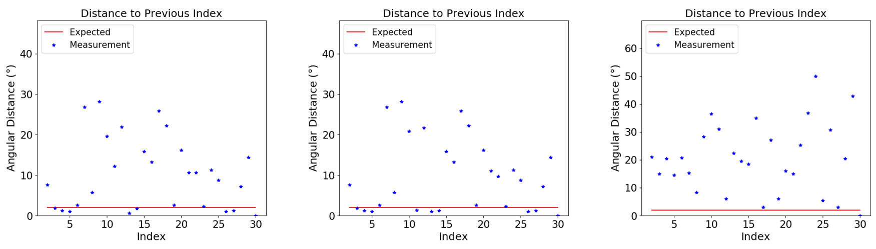


(a)



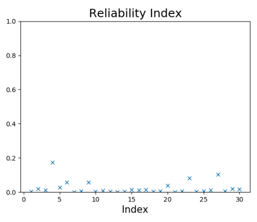
(b)

Figure 4.15: Inverse pole figures for FFC on an experimental GaSb tilt series with a template bank of 3 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

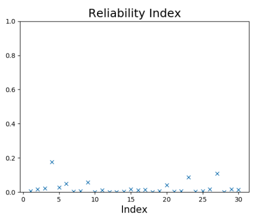


(a) NCC **(b)** ZNCC **(c)** FFC

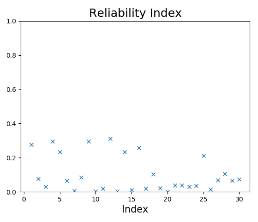
Figure 4.16: Angular distance to the previous orientation in the template matching result for experimental GaSb with a template bank of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for **(a)** NCC, **(b)** ZNCC and **(c)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.



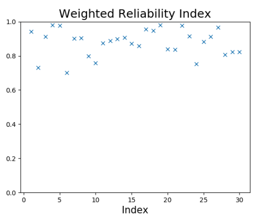
(a) NCC - Reliability Index



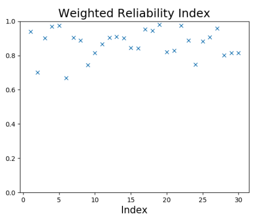
(b) ZNCC - Reliability Index



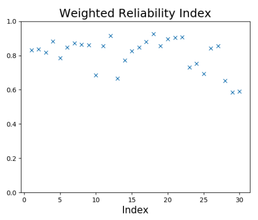
(c) FFC - Reliability Index



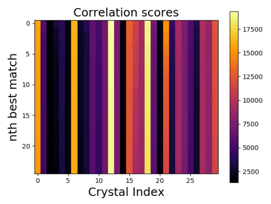
(d) NCC - Weighted Reliability Index



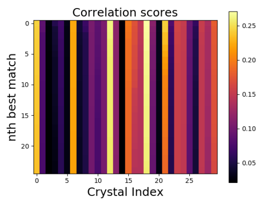
(e) ZNCC - WRI



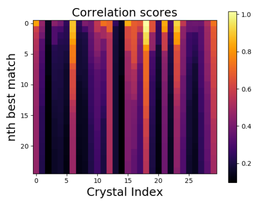
(f) FFC - WRI



(g) NCC - Correlation score-map



(h) ZNCC - Correlation score-map



(i) FFC - Correlation score-map

Figure 4.17: Top row: Reliability index for GaSb with template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for 30 orientations in an experimental tilt series for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC. A higher score indicates a higher reliability as defined in Equation (2.54) . **Middle row:** Weighted reliability index for correlation methods **(d)** NCC, **(e)** ZNCC and **(f)** FFC. A higher score indicates a higher weighed reliability as defined by Equation (2.57), where 1 is the highest score. **Bottom row:** 25 Best correlation scores represented in a map for every orientation in the tilt series for the correlation methods **(g)** NCC, **(h)** ZNCC and **(i)** FFC, with color representing the correlation score.

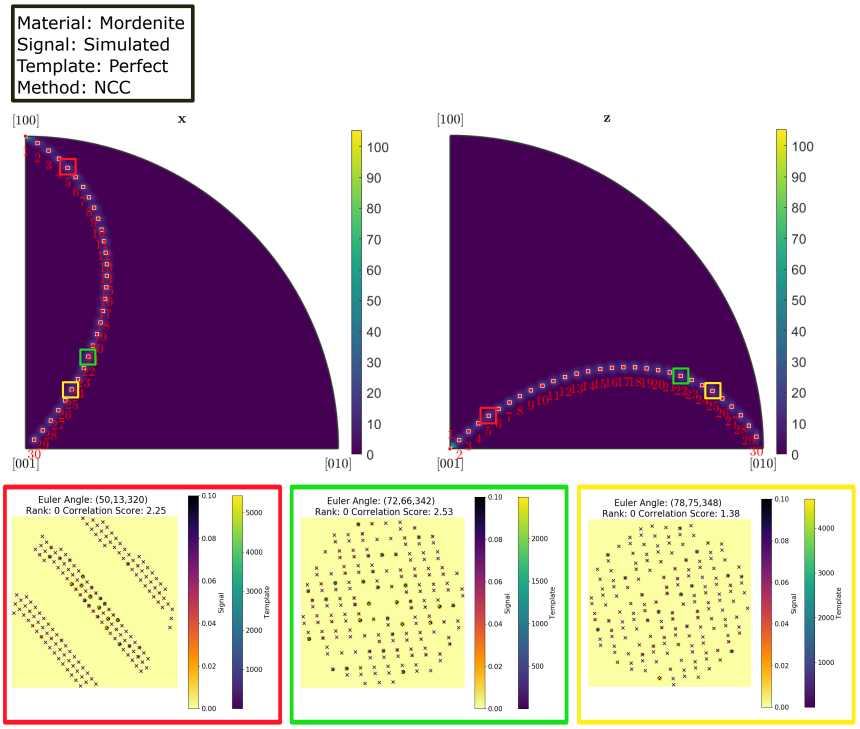
4.3 Simulated Mordnite

The parameters described in Table 3.1 were used to generate a simulated diffraction pattern for a tilt series.

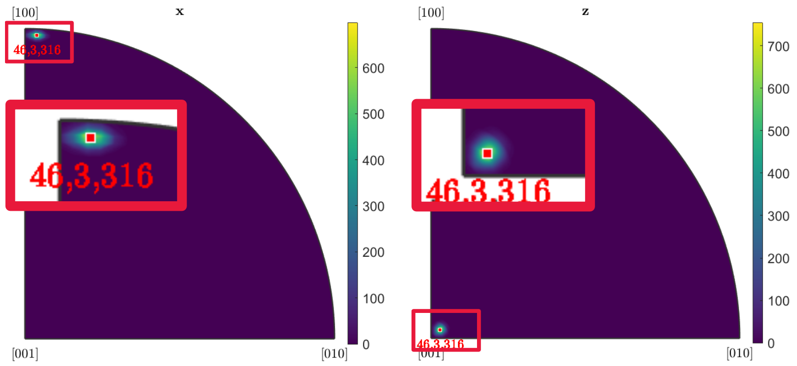
4.3.1 Perfect Template

The orientations in the tilt series were used to generate a template library, and the simulated data was matched to the template library. The inverse pole figures of the resulting orientations and rotations between subsequent orientations are shown in Figures 4.18, 4.19 and 4.20. The figures also show the best matching template together with the signal for selected indices.

Both NCC and ZNCC correctly identify all indices in the tilt series, while FFC method misses for two orientations, as seen in Figure 4.21c.

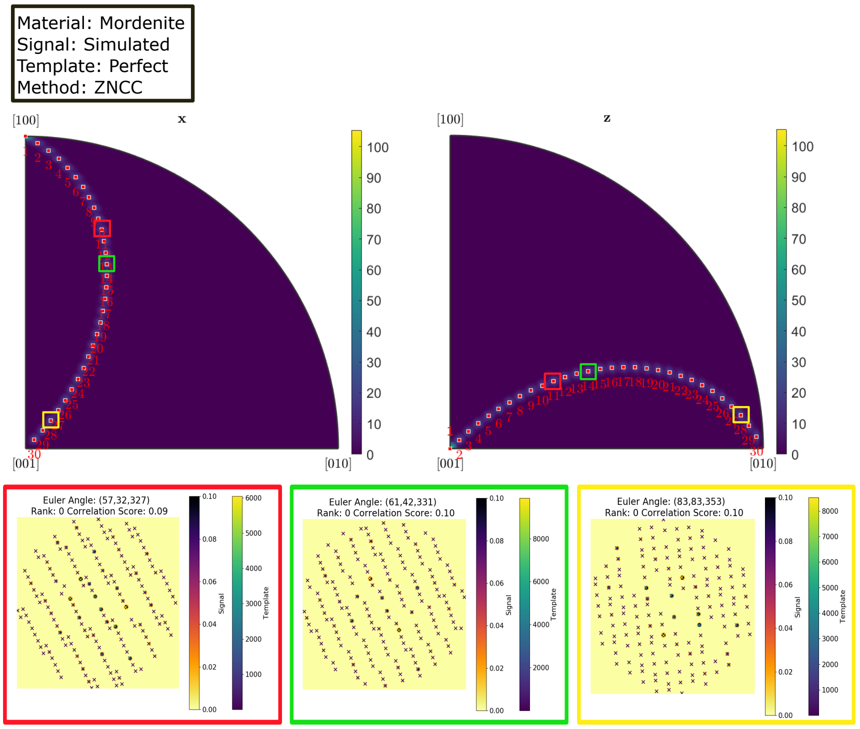


(a)

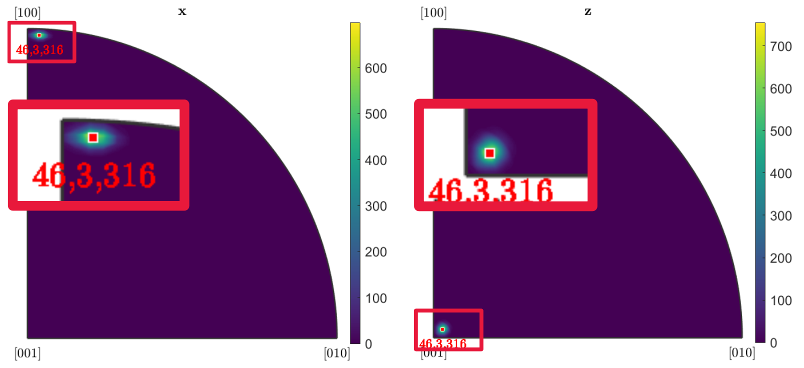


(b)

Figure 4.18: Inverse pole figures for NCC on a simulated mordenite tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

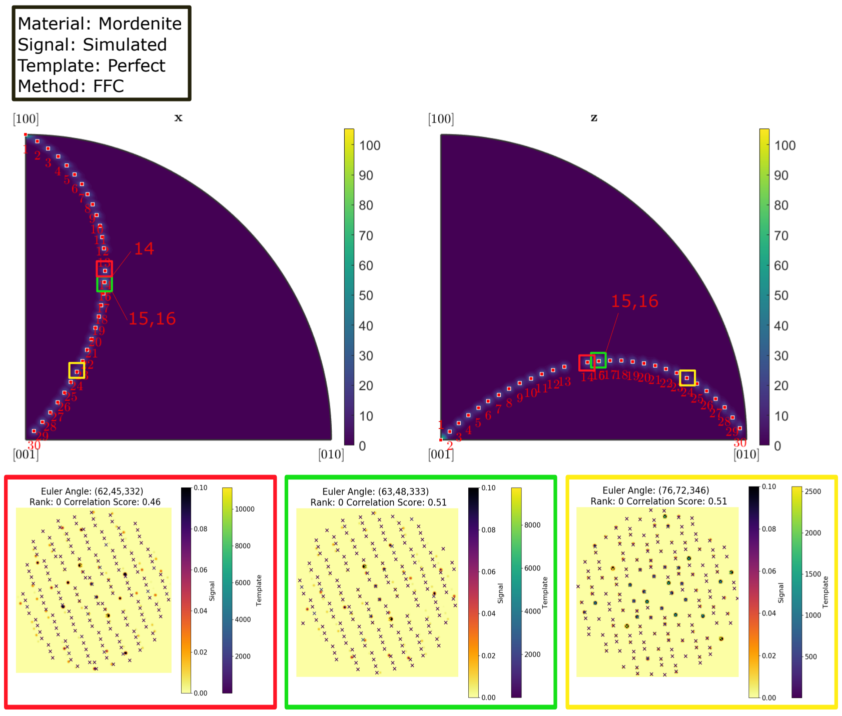


(a)

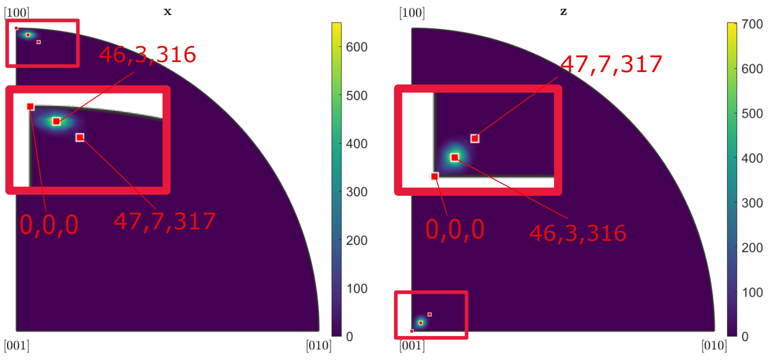


(b)

Figure 4.19: Inverse pole figures for ZNCC on a simulated mordenite tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

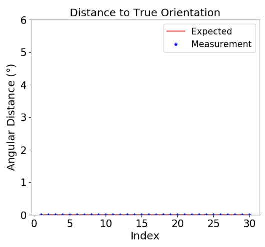


(a)

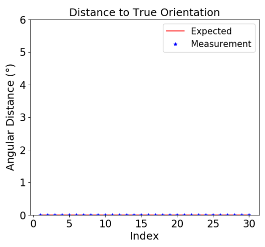


(b)

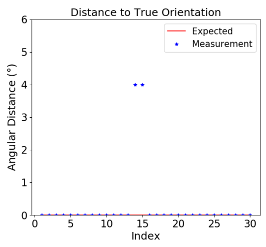
Figure 4.20: Inverse pole figures for FFC on a simulated mordenite tilt series with a perfect template input, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.



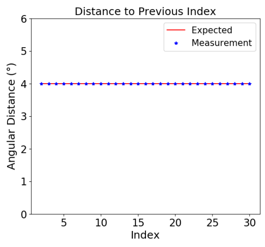
(a) NCC



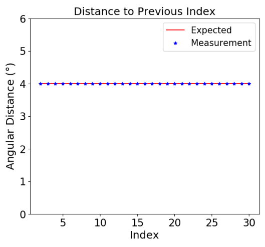
(b) ZNCC



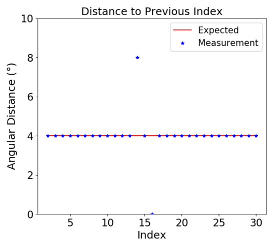
(c) FFC



(d) NCC



(e) ZNCC



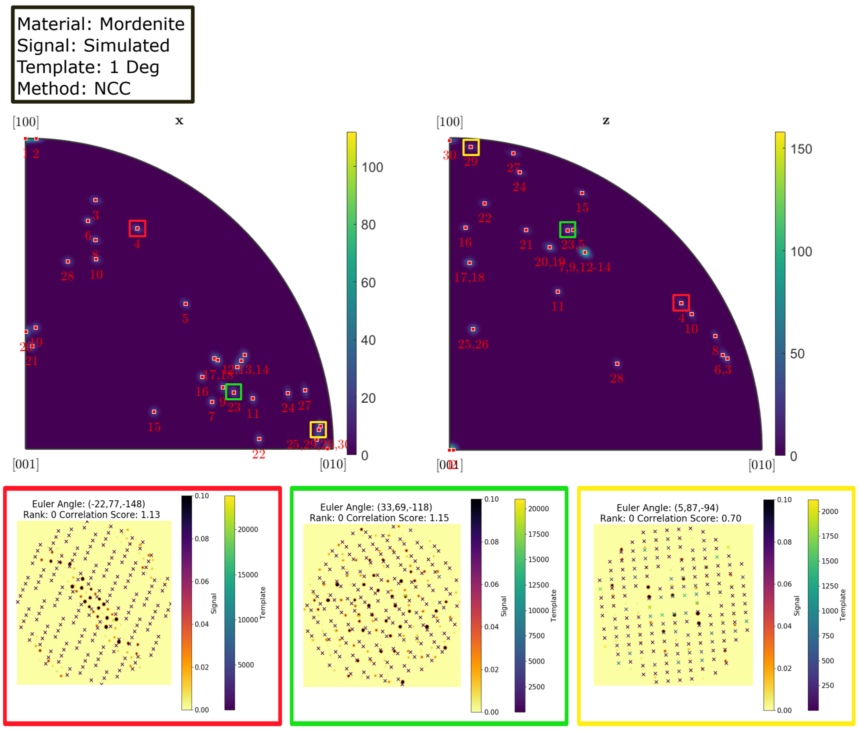
(f) FFC

Figure 4.21: Distance of template matched orientation to true orientation for mordenite with a perfect template input for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC for 30 simulated orientations. Angular distance to the previous orientation in the template matching result for **(d)** NCC, **(e)** ZNCC and **(f)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.

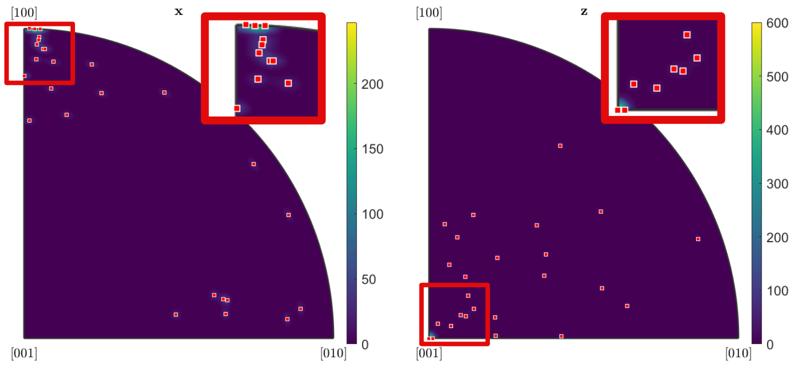
4.3.2 1 Degree Grid

A template library was generated by creating an equally spaced grid on the fundamental zone, and the simulated data was matched against the template library. The inverse pole figures of the resulting rotations and orientations are seen in Figures 4.22, 4.23 and 4.24. The angular distance from the obtained orientation to the true orientation, and the angular distance between subsequent orientations are shown in Figure 4.25. The Reliability metrics for the template matching result are shown in Figure 4.26.

None of the three correlation methods are able to correctly index the data, as seen in Figure 4.25.

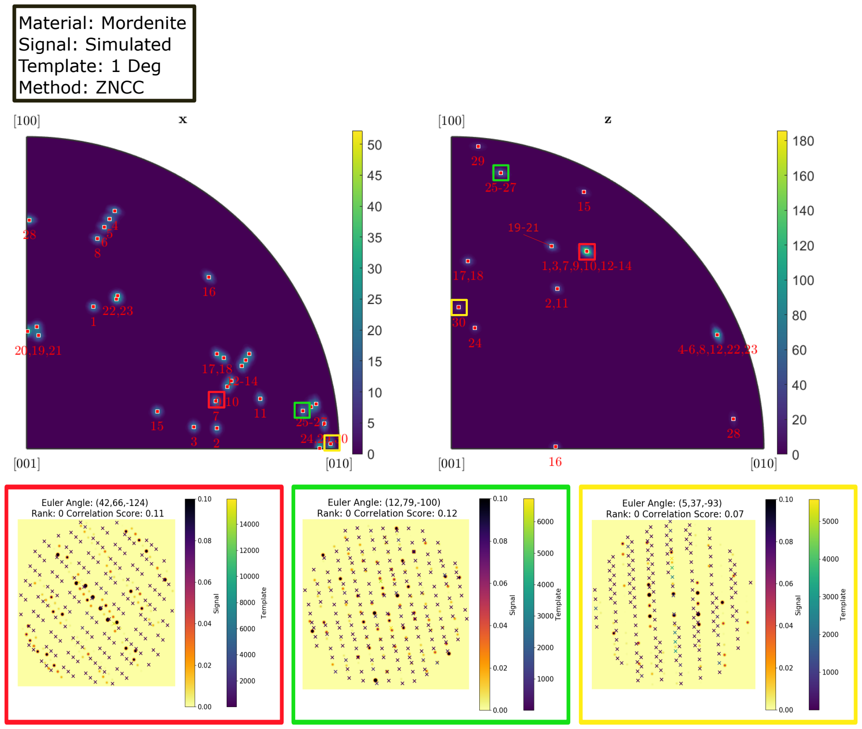


(a)

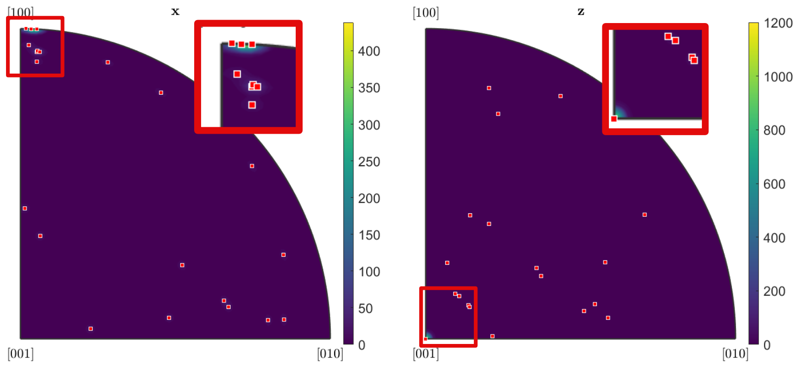


(b)

Figure 4.22: Inverse pole figures for NCC on a simulated mordenite tilt series with a template bank of 1 degree resolution, for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

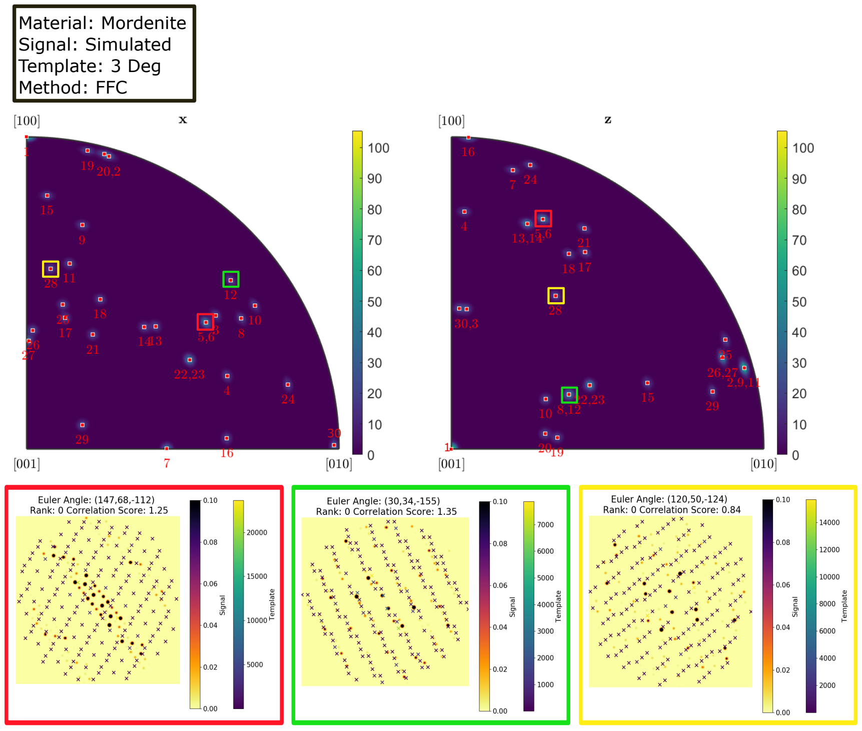


(a)

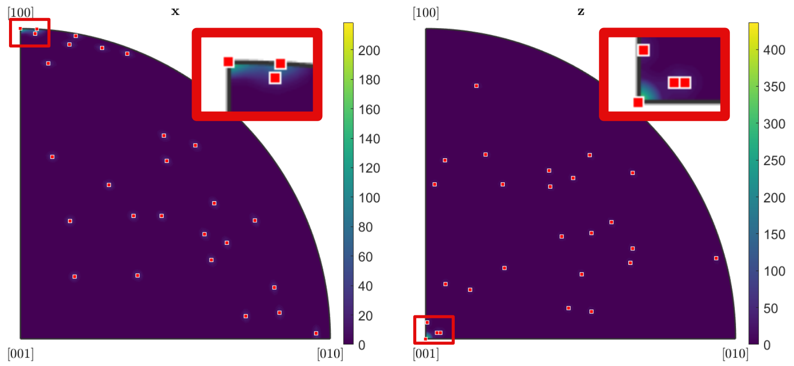


(b)

Figure 4.23: Inverse pole figures for ZNCC on a simulated mordenite tilt series with a template bank of 1 degree resolution for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

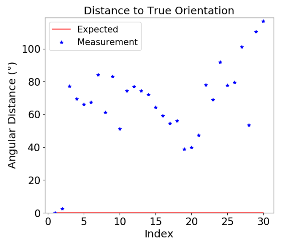


(a)

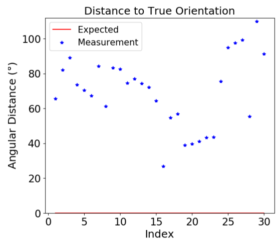


(b)

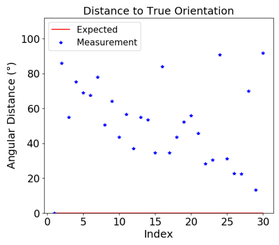
Figure 4.24: Inverse pole figures for FFC on a simulated mordenite tilt series with a template bank of 3 degree resolution for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.



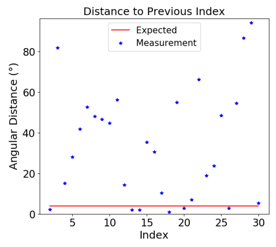
(a) NCC



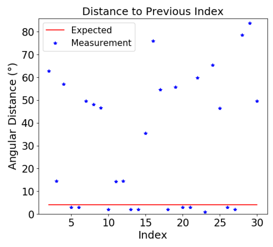
(b) ZNCC



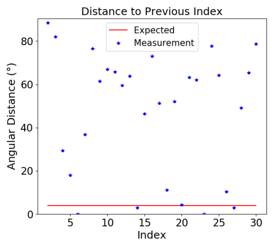
(c) FFC



(d) NCC

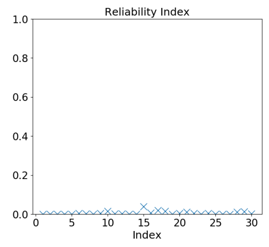


(e) ZNCC

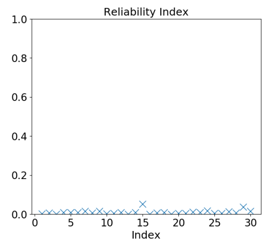


(f) FFC

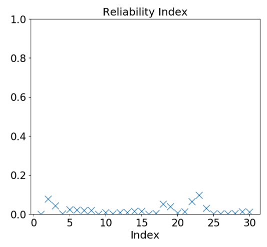
Figure 4.25: Distance of template matched orientation to true orientation for mordenite with a template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC for 30 simulated orientations. Angular distance to the previous orientation in the template matching result for **(d)** NCC, **(e)** ZNCC and **(f)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.



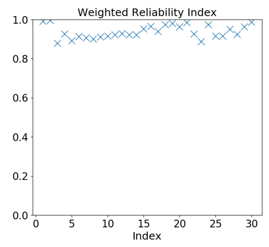
(a) NCC - RI



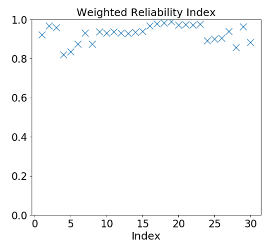
(b) ZNCC - RI



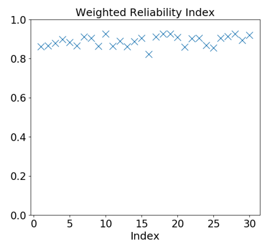
(c) FFC - RI



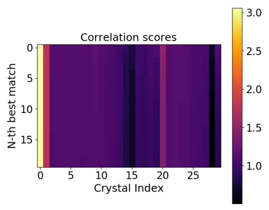
(d) NCC - WRI



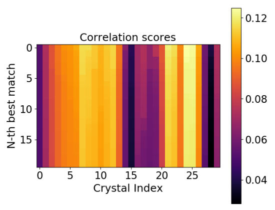
(e) ZNCC - WRI



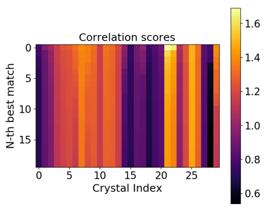
(f) FFC - WRI



(g) NCC - CSM



(h) ZNCC - CSM



(i) FFC - CSM

Figure 4.26: Top row: Reliability index for mordenite with a template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for 30 orientations in a simulated tilt series for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC. A higher score indicates a higher reliability as defined in Equation (2.54) . **Middle row:** Weighted reliability index for correlation methods **(d)** NCC, **(e)** ZNCC and **(f)** FFC. A higher score indicates a higher weighed reliability as defined by Equation (2.57), where 1 is the highest score. **Bottom row:** 20 Best correlation scores represented in a map for every orientation in the tilt series for the correlation methods **(g)** NCC, **(h)** ZNCC and **(i)** FFC, with color representing the correlation score.

4.3.3 Random Orientations

Table 4.4: Results for random orientations

Method	Correctly indexed (%)	Standard Deviation (°)
NCC	54.3	0.74
ZNCC	30.3	1.38
FFC	7.3	1.83

For the same data, pairs of random orientations were drawn. The distance between the true value of the orientations was found. The distance between the corresponding template matching orientations was computed, and compared to the true distance. A deviation of less than 1.8 degrees was taken to be correct. The result of the test is presented in Table 4.5.

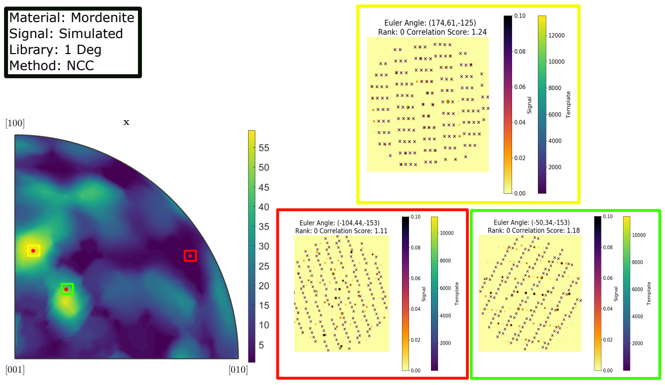
Table 4.5: Fraction of correctly indexed orientations for the correlation methods NCC, ZNCC and FFC, computed by defining a distance between template matching orientations within 1.8 degrees from the true distance to be correct.

Method	Correctly indexed (%)
NCC	39.5
ZNCC	17.0
FFC	7.7

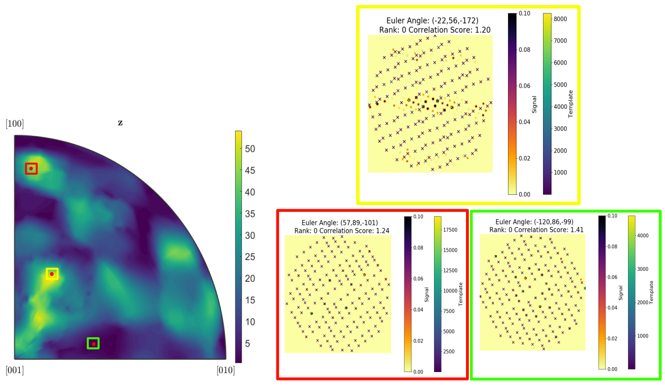
By comparing the angular distance to the score of the Reliability index and the weighted reliability index, the correlation coefficients for RI and WRI were computed. In order to test if there were a correlation between the total intensity of the simulated signal and the indexation error, the correlation coefficient between the total signal intensity and the angular distance to the true value were computed.

Table 4.6: Correlation coefficients - Distance to true value vs Reliability Metric

Method	NCC	ZNCC	FFC
WRI	-0.547	-0.371	-0.168
RI	-0.172	-0.170	-0.122
I_{tot}	-0.009	-0.082	-0.266

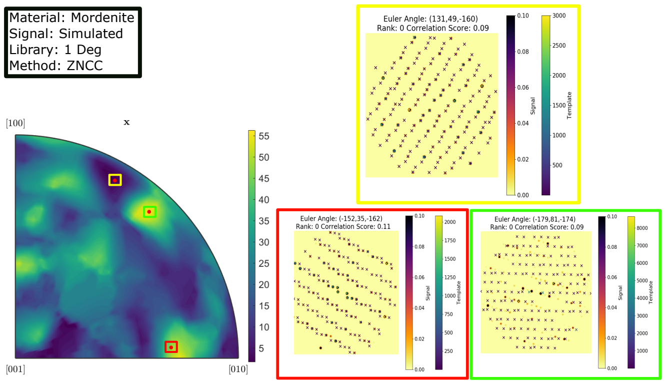


(a)

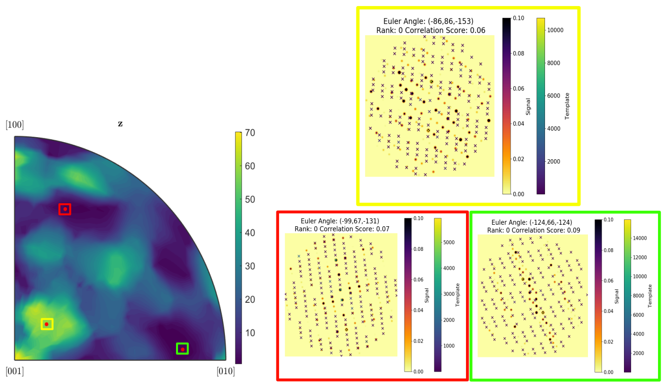


(b)

Figure 4.27: Inverse pole figure for NCC to a set of 300 random orientations for simulated mordenite, with a template bank of 1 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.

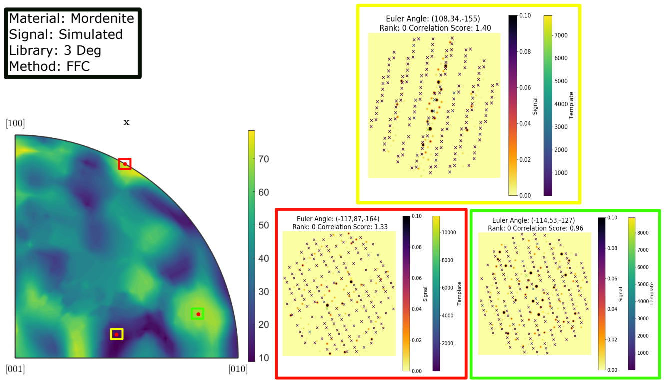


(a)

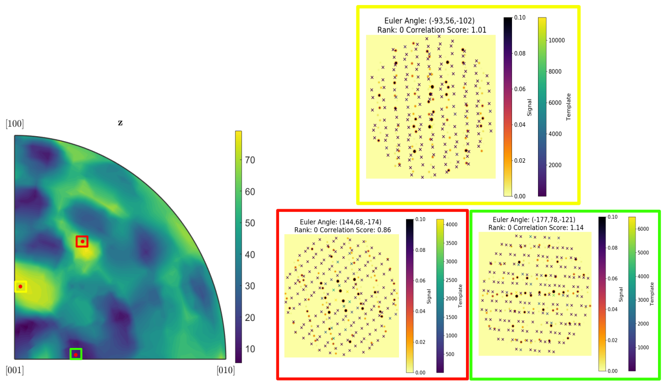


(b)

Figure 4.28: Inverse pole figure for ZNCC to a set of 300 random orientations for simulated mordenite, with a template bank of 1 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.



(a)



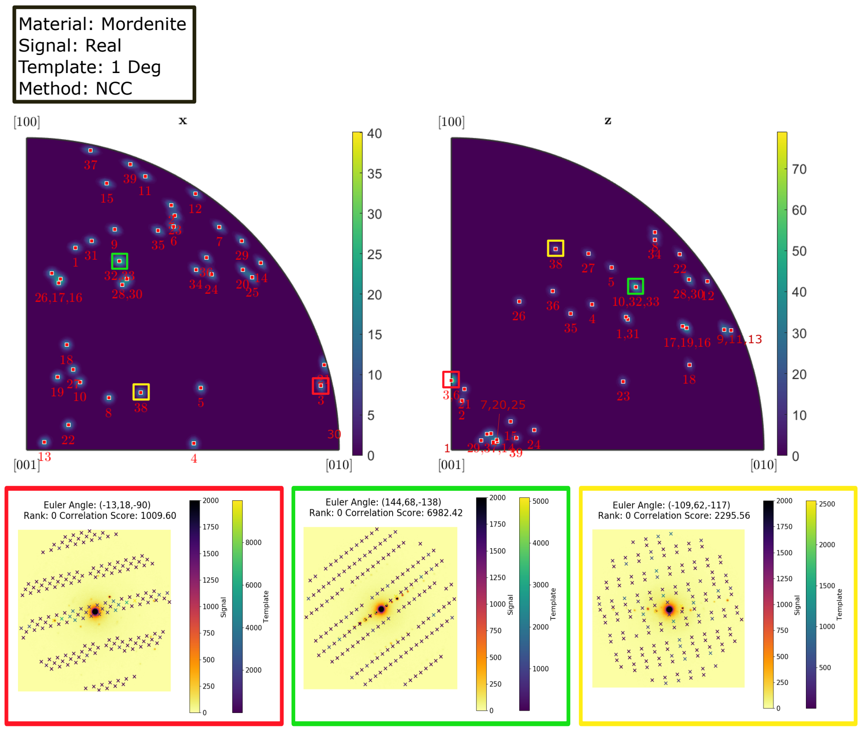
(b)

Figure 4.29: Inverse pole figure for FFC to a set of 300 random orientations for simulated mordenite, with a template bank of 1 degree resolution in **(a)** X and **(b)** Z direction. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. The insets show the signal together with the best fitting templates for selected example positions on the IPF marked by colored squares in **(a)** and **(b)**. Euler angles for the template and correlation score are indicated for the selected cases.

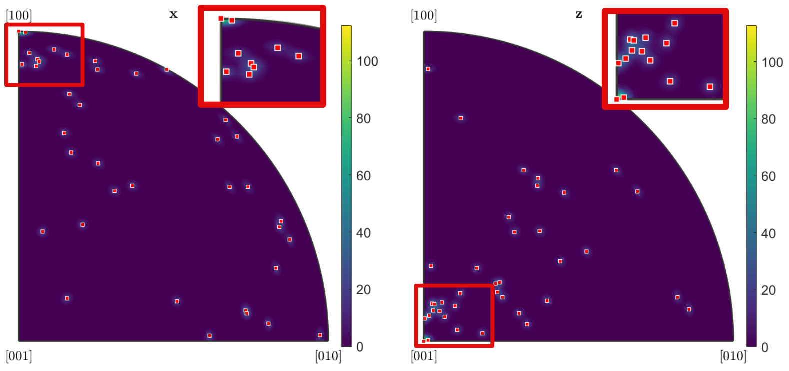
4.4 Mordenite Dataset

A template library was generated by creating an equally spaced grid on the fundamental zone, with a 1 degree spacing for NCC and ZNCC and 3 degrees spacing for FF. The mordenite dataset was matched against the template library. The inverse pole figures of the resulting rotations and orientations are shown in Figures 4.30, 4.31 and 4.32. The distance to the previous index is plotted in Figure 4.33.

The reliability metrics are shown in Figure 4.17. The scores for the Weighted reliability indices provides higher values for NCC than for ZNCC. For FFC the WRI score is low for most indices. The reliability index is close to 0 for most indices for all three correlation methods. The Correlation score maps of NCC and ZNCC look very similar, with 25 templates obtaining a similar correlation score for all indices. For FFC the difference between the best scoring templates and the subsequent templates is bigger.

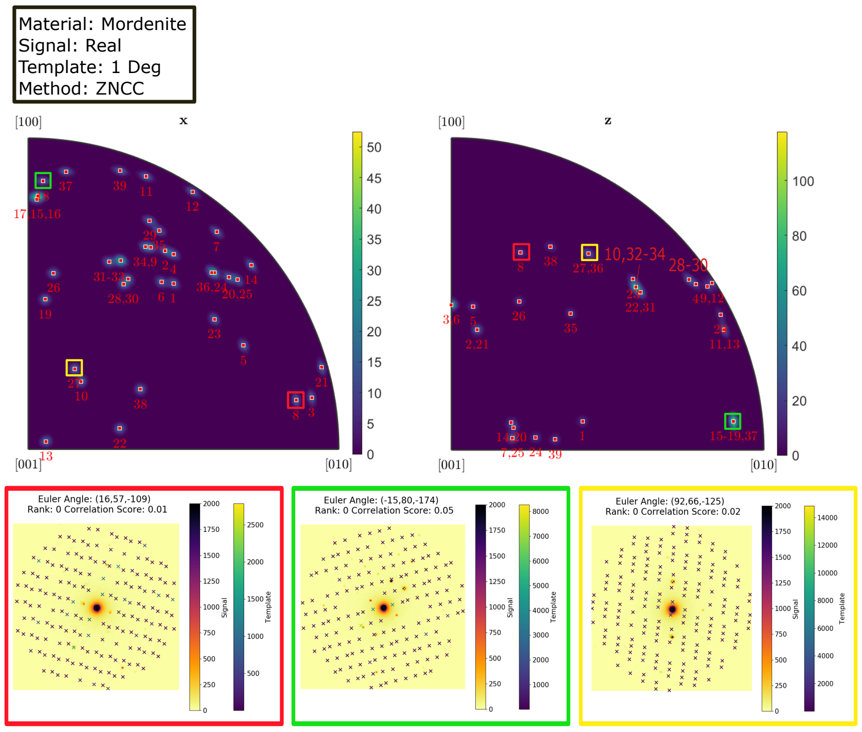


(a)

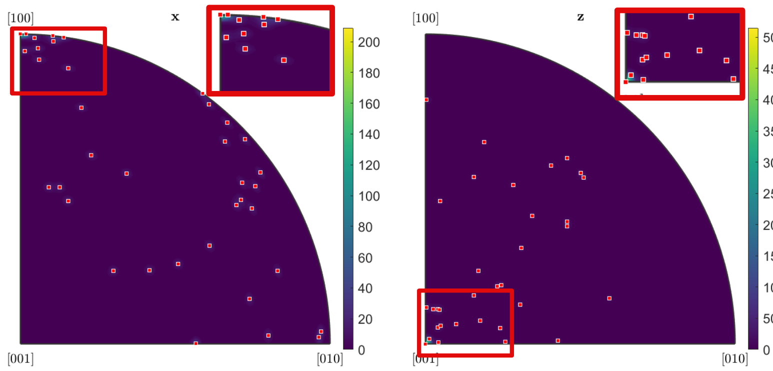


(b)

Figure 4.30: Inverse pole figures for NCC on an experimental mordenite tilt series with a template bank of 1 degree resolution for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

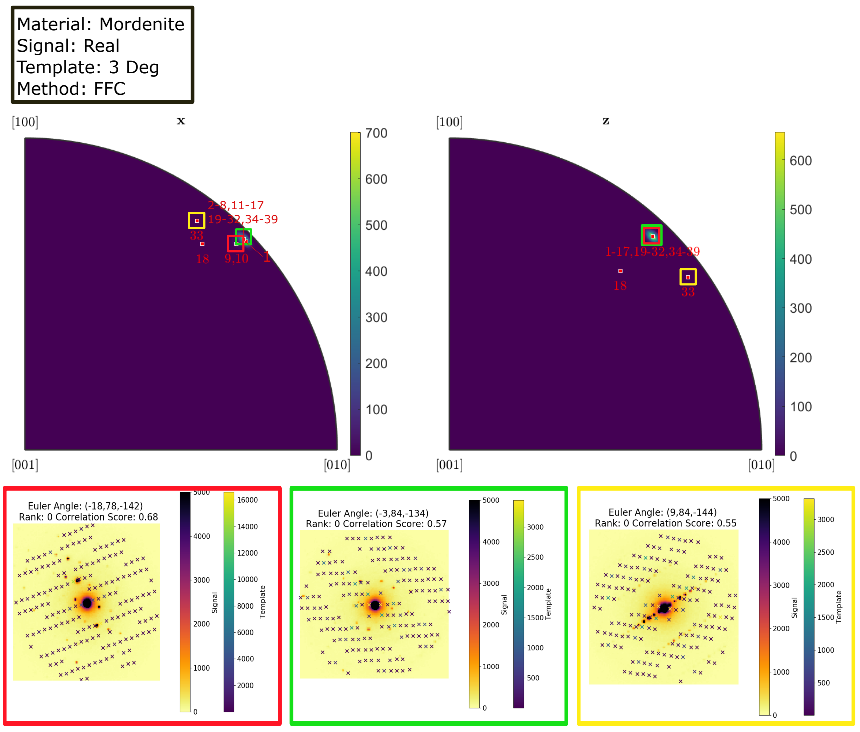


(a)

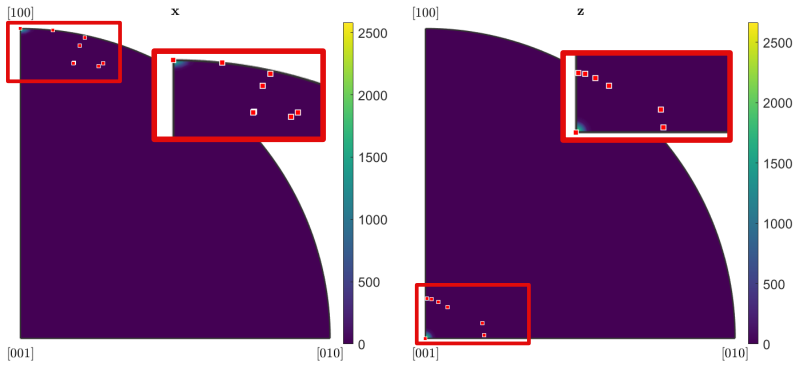


(b)

Figure 4.31: Inverse pole figures for ZNCC on an experimental mordenite tilt series with a template bank of 1 degree resolution for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.

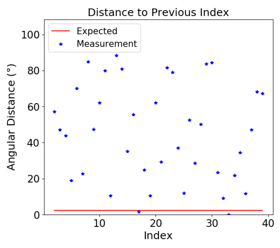


(a)

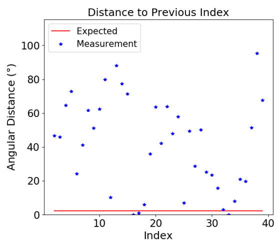


(b)

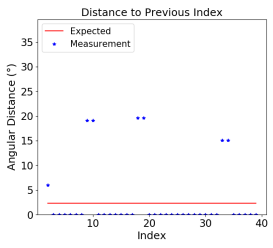
Figure 4.32: Inverse pole figures for FFC on an experimental mordenite tilt series with a template bank of 3 degree resolution for **(a)** orientations and **(b)** rotations in x and z direction. Numbers in **(a)** refer to sequential orientations in the tilt series and in insert in **(b)** to Euler angle. For the orientations in **(a)** three example cases are selected marked by colored squares. Signal and template patterns, with Euler angles, rank and correlation score, are overlaid.



(a) NCC

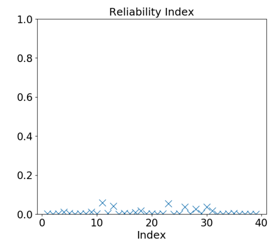


(b) ZNCC

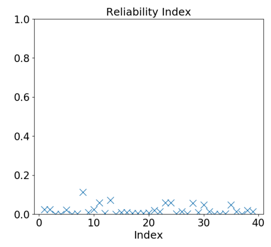


(c) FFC

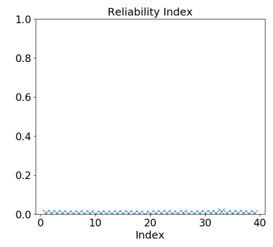
Figure 4.33: Angular distance to the previous orientation in the template matching result for experimental mordenite with a template bank of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for **(a)** NCC, **(b)** ZNCC and **(c)** FFC. The expected result is shown as a red line in every plot, and every template matched result is represented by a blue star.



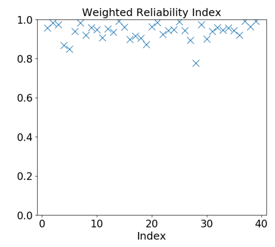
(a) NCC - RI



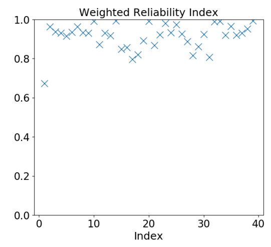
(b) ZNCC - RI



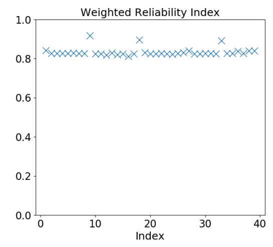
(c) FFC - RI



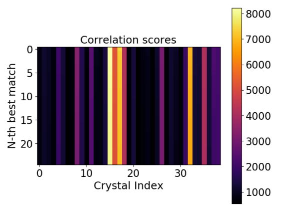
(d) NCC - WRI



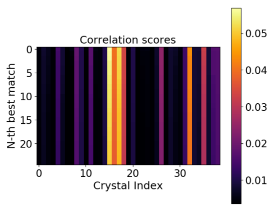
(e) ZNCC - WRI



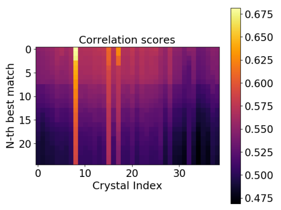
(f) FFC - WRI



(g) NCC - CSM



(h) ZNCC - CSM



(i) FFC - CSM

Figure 4.34: Top row: Reliability index for mordenite with a template library of 1 (NCC and ZNCC) or 3 (FFC) degree resolution for 39 orientations in an experimental tilt series for correlation methods **(a)** NCC, **(b)** ZNCC and **(c)** FFC. A higher score indicates a higher reliability as defined in Equation (2.54). **Middle row:** Weighted reliability index for correlation methods **(d)** NCC, **(e)** ZNCC and **(f)** FFC. A higher score indicates a higher weighed reliability as defined by Equation (2.57), where 1 is the highest score. **Bottom row:** 25 Best correlation scores represented in a map for every orientation in the tilt series for the correlation methods **(g)** NCC, **(h)** ZNCC and **(i)** FFC, with color representing the correlation score.

Chapter 5

Discussion

In this work the correlation methods NCC, ZNCC and FFC have been applied for eight different test cases:

1. A simulated GaSb tilt-series and a perfect template bank.
2. A simulated GaSb tilt-series with a template bank of equally spaced orientations on the fundamental zone.
3. A simulated GaSb signal for 1000 random orientations with a template bank of equally spaced orientations on the fundamental zone.
4. An experimental GaSb tilt-series with a template bank of equally spaced orientations on the fundamental zone.
5. A simulated mordenite tilt-series with a perfect template bank.
6. A simulated mordenite tilt-series with a template bank of equally spaced orientations on the fundamental zone.
7. A simulated mordenite signal for 300 random orientations with a template bank of equally spaced orientations on the fundamental zone.
8. An experimental mordenite tilt-series with a template bank of equally spaced orientations on the fundamental zone.

Due to memory constraints the resolution of the template banks of equally spaced orientations on the fundamental zone varied, and was 1 degree for NCC and ZNCC and 3 degrees for FFC.

5.1 Choice of tilt series for GaSb

In Figure 5.1 the full tilt series is plotted on the inverse pole figure in x and z directions. As orientations 16 - n and 16 + n coincide for both x and y directions they are symmetrically equivalent orientations, and should produce the same result. Still, in Figure 4.4c, the result for 16 + n is not the same as the result for 16 - n for orientations 10 and 22 or 9 and 23. This shows that, although the orientations in 16 + n and 16 - n are symmetrically equivalent, they are not behaving as such.

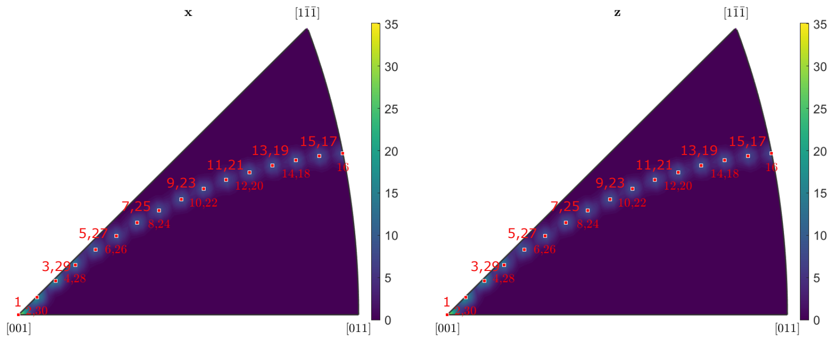


Figure 5.1: The inverse pole figure for the GaSb tilt series defined in Table 3.1, in the X and Y directions. The numbers refer to sequential orientations in the tilt series

To further examine this, the templates of orientations 10 and 22 are plotted together in Figure 5.2. The red line in the figure indicates the mirror symmetry that relates the two patterns. For the GaSb structure, the two templates should be indistinguishable for a cubic structure. A previous work by the author [68] found a bug in the simulations of diffraction patterns that lead to a discussion in the pyxem/diffsims forum[1], which in turn lead to changes in the simulation of diffraction patterns. The process of simulating the diffraction pattern of a given orientation is described in Appendix B.2.2. Rotating the structure is complicated in the pyxem code because it relies on the diffpy.structure library [71], where the active rotation convention is used. For the remainder of the pyxem code, and for this work, the passive rotation convention has been used, as discussed in Section 2.4.3. This leads to complications related to the conversion between conventions, and it is possible that the phenomenon showed in this section stems from a bug in the conversion between conventions.

The ramifications of this phenomenon will greatly impact template matching results for both simulated and experimental results. The orientations in the template library are generated from a fundamental zone, which constitutes a set of unique symmetry operations. The fundamental zone is a property of the structure symmetry, and any complete set of rotations on SO(3) will have a subset that is the fundamental zone. It is not limited to the Euler angles in the Bunge convention that are used in pyxem. Now, a rotation operations that is supposed to generate the same result as its projection in the fundamental zone, generates a different result. Therefore, either (1) there is another rotation operation within the fundamental zone that corresponds to the new result, or (2) there are two operations within the defined fundamental zone that will produce the same result. These two hypotheses follow from the realization that even if rotations are applied in an erroneous way, there is still a fundamental zone defined by the structure symmetry, that has a one-to-one

[1]pyxem/diffsims Pull request # 60.

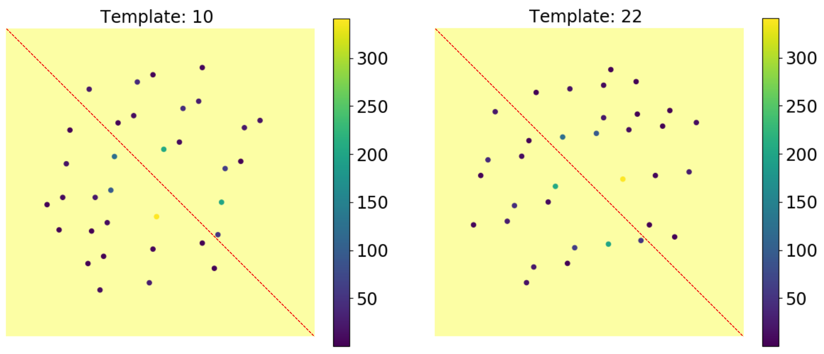


Figure 5.2: Templates for orientations 10 and 22 in the tilt series of GaSb with a template library of 1 degree resolution. The red dotted line shows the mirror line that translates the signal for orientation 10 into the signal for orientation 22.

correspondence with all other fundamental zones in SO(3) (24 for cubic structures). If (2) is true, then the defined fundamental zone is not a true fundamental zone in the rotation convention that is being applied, and there are diffraction patterns that will have no match in the template library. This is, assuming that the function used to create a grid of constant spacing for the fundamental zone function is correctly implemented, and that there are no symmetrically equivalent orientations in the grid.

Assume first that (1) is true, and consider the case where NCC method was applied with a template library of 1 degree resolution. In Figure 4.8a, the distances to the true orientation is plotted. For orientations 1-16 and 16-30 the results are close to the true value. If there was an orientation at a different place in the fundamental zone that corresponded well to one of the orientations in the tilt series, it would give a large deviation from the expected result. This is not seen for either orientations 1-16 or 16-30. To further examine the difference between orientations 1-16 and 16-30, the best fitting templates for orientations 10 and 22 are plotted in Figure 5.3. The signal red dotted line is the mirror line that translates orientation 10 to orientation 22. The best fitting templates for the two signals do not share the mirror symmetry, but we see that the templates are a near perfect fit for both orientation 10 and orientation 22. This indicates that both orientation 10 and orientation 22 belong to areas covered by the Fundamental zone of the template bank. This is not expected, as they are symmetrically equivalent orientations, and should appear only once in the Fundamental zone. The implementation is that the fundamental zone grid is not correct, which removes the premise of hypotheses (1) and (2). The finding is that the fundamental zone grid spans more than the fundamental zone, which is not a big issue unless computational speed is considered. What is a serious issue is if the fundamental zone is not spanned.

In order to confirm the result from the previous paragraph the full tilt series NCC on the simulated GaSb with a template bank of 1 degree resolution is shown in Figure 5.4. In 5.4a, the antipodal symmetry operation is applied in projecting orientations to the fundamental zone. In 5.4b the antipodal symmetry operation is not applied. In the (b) template, orientations 1-12 are missing for the z direction of the inverse pole figure, and 3-11 are

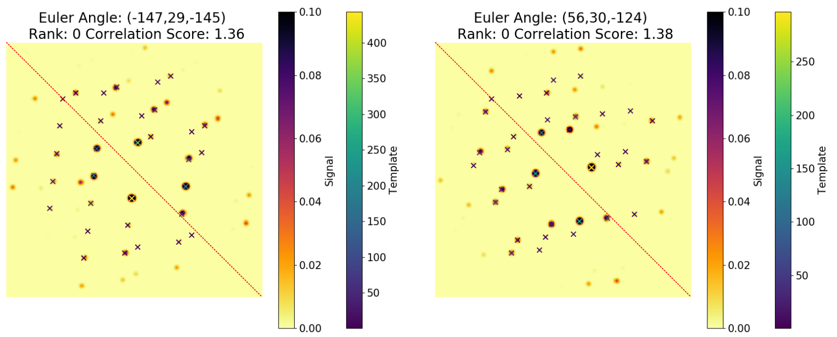
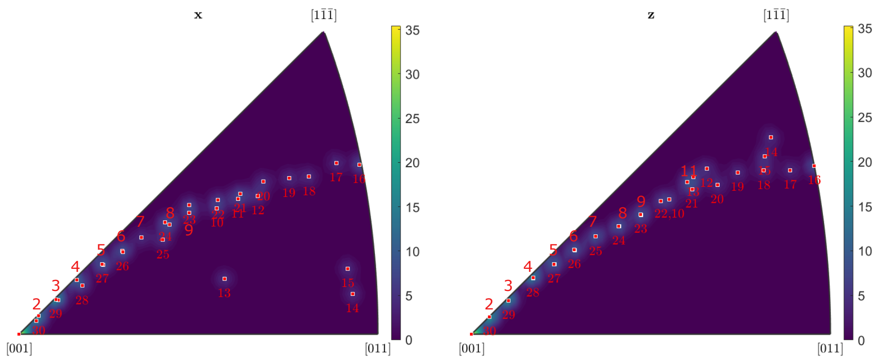


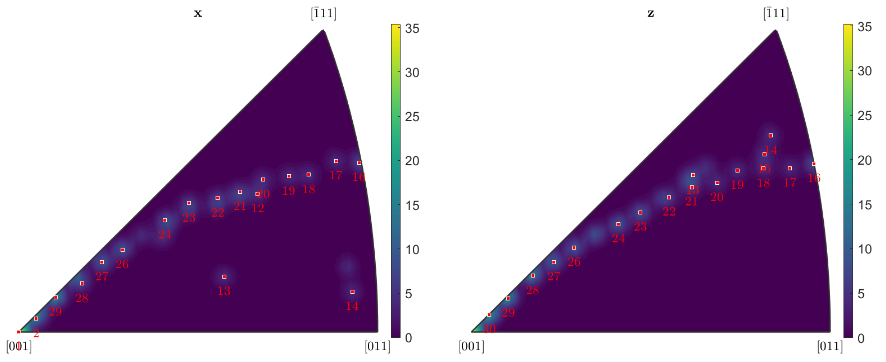
Figure 5.3: Templates together with signal for orientations 10 and 22 in the tilt series of GaSb with a template library of 1 degree resolution. The red dotted line shows the mirror line that translates the signal for orientation 10 into the signal for orientation 22.

missing for the x direction. This shows that without the antipodal symmetry operation, orientations 16 + n do not coincide with orientations 16 - n. Friedel's law, stated by Friedel in 1913 [72], says that the reflections [hkl] and [$\bar{h}\ \bar{k}\ \bar{l}$] can not be distinguished. There should therefore be no need to apply antipodal symmetry for the orientations to coincide, and the result is an indication that the simulation is wrong, and also that a set of orientations greater than the fundamental zone is covered by the template bank. Although the premise for hypotheses (1) and (2) is removed, the underlying concept, that the structure has a symmetry and a well defined fundamental zone still holds. If the rotations are applied in an erroneous way, so that patterns that should be identical are not identical, it necessarily also implies that patterns that should not be identical will be identical, as there is a one-to-one correspondence between different fundamental zones. In the following discussion of template matching, it is therefore important to consider carefully if a template matches the signal, and not only that the obtained result is sensible, when different correlation methods are evaluated. A positive aspect of the simulation error is that the template bank covers all the orientations in the tilt series, and that pairs of diffraction patterns are equal, apart from a mirror operation. This simplifies the study of what goes wrong for instance for template 10 in the tilt series with a perfect template for the FFC method, seen in Figure 4.4c, as the situation can be compared to an equivalent case in template 22, where the correct template was selected.

In order to fix the issue of missing symmetry in the simulated patterns, the code for rotating the structure must be reevaluated. In order to test that the reevaluated code works, an orientation must be chosen, and a list of all equivalent orientations from symmetry operations generated. Ideally the pattern of the orientation should not have much symmetry, so a pattern off zone-axis is to be selected. For every orientation in the list of equivalent orientations, a diffraction pattern is simulated, and the structure is rotated correctly if all the symmetry equivalent orientations produce the same simulated diffraction pattern.



(a)

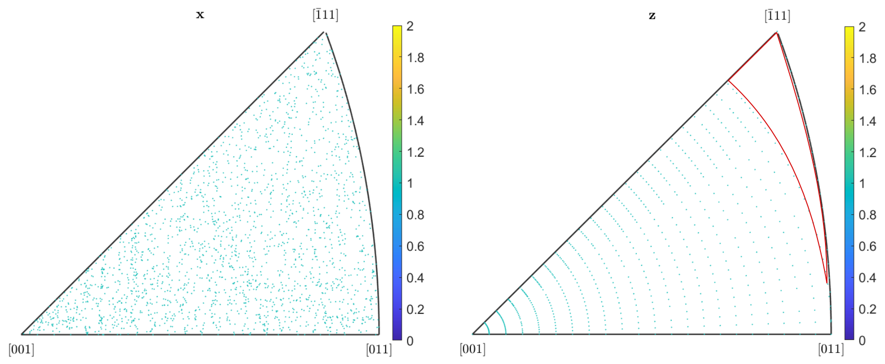


(b)

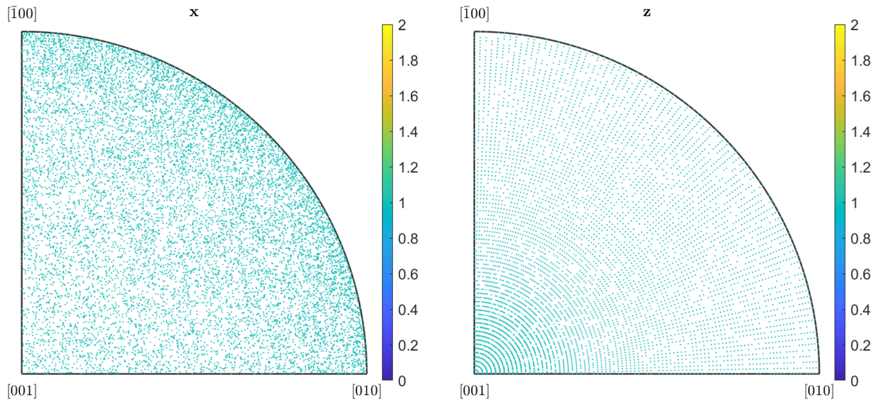
Figure 5.4: Inverse pole figure of the full tilt series for GaSb with a template bank of 1 degree resolution. In **(a)** the antipodal symmetry operation about (0,0,0) is applied, while in **(b)** it is not applied. The numbers refer to sequential orientations in the tilt series. Orientations 3 - 12 are not seen in the X direction in **(b)** and orientations 1 to 12 are not seen in the Z direction in **(b)**

5.2 Confirming that the orientation list spans the Fundamental Zone

In light of the discussion in the previous section it should be confirmed that the function for generating a grid on the fundamental zone spans the full fundamental zone. This was done in a previous project in an indirect way [68], but is repeated here in a more direct way with the plotting tools provided in MTEX [69]. In Figure 5.5 the list of orientations on the fundamental zone are plotted for cubic and orthorhombic grids. For the Z direction of the inverse pole figure for the cubic grid, there is a hole near the $[\bar{1}11]$ corner, outlined with a red outline. No orientations have been projected, and the result shows that the fundamental zone for GaSb is not fully covered, while the fundamental zone for mordenite is covered. Notice also that the inverse pole figures for both the cubic and the orthorhombic grids are more densely populated near the [001] corner of the Z direction. This is an artifact of choosing to have an equal angle between orientations in the grid, rather than having an equal area covered by each orientation. Both options are available in pyxem, but both have the same characteristic zone with no orientations for the Z direction of the inverse pole figure. This bug will have ramifications for orientations that belong to the zone where the template bank has no entries. The closest possible match for an orientation in this area will be further from the true value, and this extra distance will contribute to a worse template matching result. The procedure for finding a set of orientations spanning the fundamental zone is described in Appendix B.2.1.



(a)



(b)

Figure 5.5: Randomly drawn orientations from the list of orientations spanning the fundamental zone are projected to the X and Z directions of the inverse pole figure for **(a)** a cubic structure **(b)** an orthorhobic structure. The area marked by a red outline for the Z direction in **(a)** is an area without any orientations projected to it.

5.3 Evaluating template matching for simulated data

Consider the top rows of Figures 4.4 and 4.21. The NCC method correctly identified all indices for the simulated GaSb and mordenite tilt series, when the template bank consisted of only orientations present in the tilt series. This was also the case for ZNCC. The FFC method got 28/30 orientations correct for both GaSb and mordenite with a perfect template bank, with a four degree deviation from the true result for the misindexed orientations. This demonstrates that all three methods are able to differentiate between 30 patterns in a simulated tilt series, when every orientation in the tilt series has a perfect fit in the template bank. All three methods are therefore applicable to the correlation between a template and a signal, and can be further evaluated in a more realistic test where the orientations in the simulated signal are assumed to be unknown when generating the template bank. The template bank is then generated by selecting orientations on an equally spaced grid on the fundamental zone.

Consider cases where orientation mapping is performed with a template bank of equally spaced orientations on the fundamental zone. As a threshold value for labeling an orientation as correct, an angular distance of 3 degrees to the true value was chosen for this work. This threshold value should be selected based on the objective of the study in question. One common use of orientation mapping is for determining grain boundaries [73]. For studies of grain boundaries, the minimum misorientation for defining a grain boundary varies depending on the purpose of the study. For two example works with grain boundaries, the minimum misorientation angles to define a grain boundary were 3 degrees [74] and 4 degrees [75]. The misorientation imprecision is $\sqrt{2}$ times larger than the orientation imprecision [76], and the threshold value for labeling an orientation as correct should be selected with this in mind. The selected threshold value for this work would correspond to a minimum misorientation angle of around 4.2 degrees.

For random orientations, the standard deviation of the orientations that are defined to be correct is reported in addition to the fraction of orientations that are correctly indexed. This provides a number for the spread of the orientations taken to be correct around the true orientation. A mathematically correct treatment would take into account the orientation distribution of orientations on SO(3) for the computation of the spread. The simplification of the von Mises distribution for misorientations in Section 2.6.1 is not applicable to this problem, as the distribution is centered at zero distance from the true orientation, where the rotation axis from the observed orientation to the true orientation is not well defined. The problem can be treated with von Mises - Fisher statistics by considering the full orientation matrix treatment discussed in Appendix A.2, but for a distribution of small (Book does not provide an exact definition of small) variance around a mean orientation, the von Mises - Fisher distribution is well approximated by Gaussian statistics [61]. Although the book does not provide an exact definition of small, it stresses that even for the variance where the von Mises - Fisher distribution is furthest from the wrapped normal distribution, the differences between the two are very small, allowing the statistician to use the distribution best suited for his/her work.

Table 5.1 summarized the contents shown in the top rows of Figures 4.8 and 4.25 together with the *Correctly indexed* columns of Tables 4.1 and 4.4. For all test cases the NCC method shows the best results, by correctly indexing a larger fraction of the orientations. The overall results of ZNCC are better than those of FFC, as ZNCC correctly indexes a significantly higher fraction of orientations in three out of four tests. All methods fail for the mordenite tilt series. When the methods were applied to a simulated dataset of random orientations, with the same simulation parameters, the fraction of correctly indexed orientations is significantly higher for all methods. For GaSb and the NCC method, the opposite is the case. For the tilt series, 26/30 orientations are correctly indexed, while for random orientations the number is only 60.1%. To better understand this phenomenon, NCC is further studied as an example case. The true orientations of the tilt series are plotted together with the angular distance heatmaps from Figures 4.10 and 4.27 in Figure 5.6.

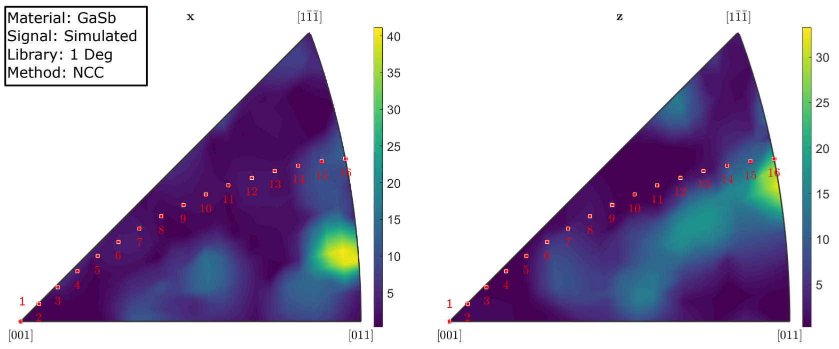
Table 5.1: Fraction of correctly indexed orientations. Numbers refer to the test index in the introduction of this section.

	NCC	ZNCC	FFC
2. GaSb - 1-Degree Template	26/30	19/30	2/30
3. GaSb Random Orientations - 1 Degree Template	60.1%	55.8%	3.1%
6. Mordenite - 1-Degree Template	2/30	0/30	1/30
7. Mordenite Random Orientations - 1 Degree Template	54.4%	30.3%	7.3%

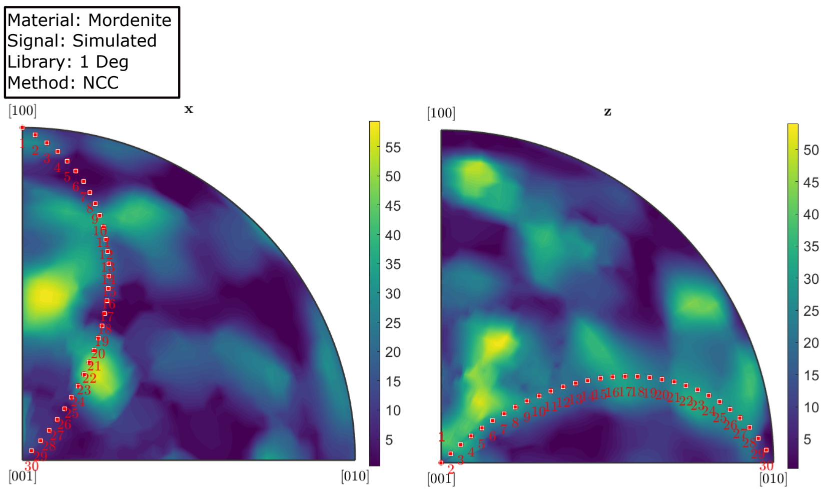
Considering Figure 5.6a for the GaSb tilt series, it is observed that indices 14-16 are in areas with a higher average distance to the true value for both the X and Z direction of the inverse pole figure. Futhermore, the indices 6-8 and 13 are in areas with slightly higher average distance to the true value. The remainder of the orientations from 1-16 are in areas with low average distance to the true value. This coincides well with what is observed in Figure 4.5, where orientations 13 to 15 are misindexed.

For Figure 5.6b for the mordenite tilt series it is observed that all orientations lie in areas of the inverse pole figure with a high average distance to the true value for the Z direction. For the X direction orientations 4-7 and 16-17 are on areas with a low average distance with the remainder of the orientations being in areas with high average distance to the true value. This could in part explain why only a few of the orientations in the tilt series are correctly indexed, while for random orientations the percentage of correct indexations is much higher.

The implications of the recent discussion of Table 5.1 are that the success rate of template matching is a local, not global parameter in the orientation space. While the general success rate, found from random orientations, is a parameter that can be used to compare correlation methods, the success rate of a specific tilt series might deviate a great deal from the general success rate. This observation is strengthened by considering Figures 4.10 to 4.12 and 4.27 to 4.29, where the average distance to the true value varies a great deal across the inverse pole figures. An issue with this realization is that the average angular distance to the true value is a property depending on the true orientation, not the template matching orientation. For situations where it is not known which orientations to expect, the general success rate will therefore be the only applicable estimate for expected success rate before running the template matching.



(a)



(b)

Figure 5.6: Inverse pole figure in X and Z direction for NCC to a set of **(a)** 1000 random orientations for simulated GaSb **(b)** 300 random orientations for simulated mordenite, with a template bank of 1 degree resolution. A local averaging algorithm was applied to the orientations to generate a smooth result. The color bar represents the angular distance from the true value. A dark area is an area where the majority of orientations have been correctly indexed. Orientations in the tilt series described in Table 3.1 are seen as red markers on the figure, and numbers refer to sequential orientations in the tilt series.

5.4 Reliability scores

WRI and RI were computed for template matching on simulated signals from a set of random orientations, when the methods NCC, ZNCC and FFC had been applied. Table 5.2 summarizes the results in Tables 4.3 and 4.6 for GaSb and mordenite, respectively. The distance from the true value has been correlated to the reliability metrics, using Equation (2.58). A large negative correlation coefficient indicates a good result, as -1 is a perfect negative correlation where a high value for the metric means that the distance to the true orientation is small. The WRI provides a stronger linear relationship to the deviation from the true value for both GaSb data and mordenite data, across all three methods. On average, the WRI provides a correlation of -0.36 while the RI provides a correlation of -0.15. So the average, based on 2 example materials and 3 different methods (that are evaluated and differently valued) gives indication that WRI to a higher degree is able to predict which template matching results are correct. Assuming that NCC is best based on the previous discussion and that it is the default method in commercial software and pyXem, comparing RI and WRI, comes to the same conclusion that WRI is a better indicator of how correct prediction is.

Table 5.2: Correlation coefficients - Distance to true value vs Reliability Metric for NCC, ZNCC and FFC applied to simulated GaSb and mordenite data from random orientations.

Method	RI	WRI
NCC - GaSb	-0.137	-0.281
ZNCC - GaSb	-0.184	-0.428
FFC - GaSb	-0.084	-0.372
NCC - mordenite	-0.172	-0.547
ZNCC - mordenite	-0.170	-0.371
FFC - mordenite	-0.122	-0.168
Average	-0.145	-0.361

Reliability metrics are used to tell if a template matching result is reliable or not. The result in Table 5.2 shows that high values for the WRI coincide with better template matching results in general terms, the matching scores themselves are difficult to interpret in physical meaning and vary in general for the three correlation methods (ie hard to compare scores directly). In practice, reliability metrics are used to answer the question: ”Is my result reliable?”, and the expected answer is either ”yes” or ”no”. To provide such an answer, a threshold value has to be set, so that any template matching result with a WRI score lower than the threshold value is labeled as wrong. The properties of the test, after setting a threshold value can be examined by a confusion matrix [77]. Considering the top row of Figure 4.8 together with the middle row of Figure 4.9. An initial test is done with course cut-off values. A threshold value for the WRI of around 0.9 was chosen so that if the WRI value is higher than this value for an orientation in Figure 4.9, then it should be close to the true value for the corresponding orientation in Figure 4.8. Table 5.3 shows the confusion matrix of a WRI threshold value of 0.9 for NCC on simulated data from 1000 random orientations. When WRI gives a value higher than 0.9, 71.8 % of the orientations are correctly indexed, while for WRI values lower than 0.9, only 32.3 % of the orientations are correctly indexed. For all orientations, 60.1 % of the orientations are correct, as seen in 4.1. It is clear that computing the WRI adds valuable information to the analysis of template matching results.

Table 5.3: Confusion table for WRI test with threshold value 0.9 for data from 1000 random orientations for NCC on simulated GaSb data.

	WRI > 0.9	WRI < 0.9
Distance < 3 deg	505	96
Distance > 3 deg	198	201

5.5 Experimental Data

For the Experimental datasets of both GaSb and mordenite, the true orientation is not known. What is known is that both stem from a tilt series, with reported oscillation angles of 1.98 degrees and 2.336 degrees, respectively. The results are therefore evaluated on whether a tilt series is observable on the inverse pole figure, and by considering the best matching template compared to the signal, to evaluate if it is a good fit. While finding the rotation axis is not the primary objective of this study, the rotation axis will be observable if a sufficient number of orientations have been correctly indexed, as seen in Figure 4.5.

By comparing Table 4.2 to Table 4.1 it is clear that although the two methods for measuring the fraction of correct data do not coincide perfectly, they are strongly related. The same is observed when considering Tables 4.5 and 4.4. As the estimates found by comparing rotations in Tables 4.2 and 4.5 require both selected orientations to be correctly indexed to give a true result, the estimates are for the squared probability of correctly indexing an orientation. For a tilt series, the results for subsequent orientations are correlated. Counting the number of correct orientations will therefore be done by considering the number of rotation in the correct angular range (As discussed in section 2.6 this is a very rough estimate). If a rotation is correct while the previous rotation was not correct, we count the rotation as +2, because two orientations are required to be correctly indexed to give a correct result. If the previous rotation is also correct, we count +1, as we have already counted the previous orientation.

Considering now the experimental tilt series, and letting a rotation that deviate less than 1.8 degrees from the reported tilt series rotation be labeled as correct. In Figure 4.16, there are a total of 10 rotations within the expected range of rotations for NCC. Counting as described above, this amounts to an estimated 15/30 orientations correctly indexed. For ZNCC the number of rotations within the expected range is 11, amounting to an estimate of 17/30 rotations within the expected range. For FFC 2 rotations, amounting to an estimate of 4 orientations are within the expected range. By considering Figure 4.17 we observe that for NCC 8 orientations have a WRI score of higher than 0.9. For ZNCC the number is also 8, while FFC has only 2 orientations of WRI higher than 0.9.

For mordenite, the distance to previous plot is shown in Figure 4.33. For NCC, 5 rotations are within the expected range, amounting to an estimate of 8 correct orientations. For ZNCC, there are 9 rotations within the expected range, amounting to an estimate of

16 correct orientations. For FFC there are three orientations within the expected range, amounting to an estimate of 6 correct orientations. The WRI values in Figure 4.34, show that NCC has 10 orientations with a WRI above 0.9, ZNCC also has 10 orientations while FFC has none.

As the WRI has demonstrated to be related to the distance from the true value, a high score makes a template matching result more reliable, but there is still a need for further evaluation. The method of counting rotations between template matching results would be much stronger if the accepted rotations were also limited to a rotational direction, not only an absolute rotation value. It is therefore necessary to consider the plot of orientations and rotations on the IPF, and consider if the results look like a tilt series.

5.5.1 NCC on experimental GaSb

The result of the NCC method applied on the experimental GaSb dataset is seen in Figure 4.13. There are areas on the inverse pole figure where subsequent indices lie in close vicinity, as expected for a tilt series. Indices 3-6, 12-14 and 25-27 resemble a tilt series locally for both the x- and z-projections. The general picture is, however, not that of a tilt series, and it is hard to identify the tilt axis from looking at the inverse pole figures. The template and the signal for selected indices do not match very well, apart from in a couple of high-intensity positions. This would indicate that for the selected indices we have not found the true answer. The plot of the rotations on the inverse pole figure show that there is a region with a large concentration of rotations, that could correspond to the constant tilt in the tilt series. Within this region the orientations are quite spread, and it is hard to determine the true tilt-axis and angle of rotation. The NCC method fails to determine the orientations from the experimental GaSb dataset.

5.5.2 ZNCC on experimental GaSb

For ZNCC on the experimental GaSb dataset the results seen in Figure 4.14 resemble those of NCC. The indices 2-6, 12-14 and 25-27 are locally aligned for both the x- and z-projections, while the remainder of the indices are scattered over the inverse pole figure, making it hard to identify the tilt axis in the tilt series. The template and the signal matches quite well for the green rectangle representing index 19, but do not match well for the red and yellow rectangles, representing indices 13 and 29, respectively. Both IPFs for rotations have a region with higher density, but again the rotations are quite spread within this region, making it difficult to determine the tilt-axis and angle of the tilt-series. The ZNCC method also fails in determining the orientations from the experimental GaSb dataset.

5.5.3 FFC on experimental GaSb

The result for FFC on the experimental GaSb dataset, seen in Figure 4.15, does not have any areas where subsequent indices are aligned for both the x- and z- projections of the inverse pole figure. For the x-projection the orientations are scattered over the IPF, while the z- projection has only 4 different points to which all orientations in the tilt-series are projected. The template does not match the signal for any of the selected indices where the template is plotted together with the signal. Neither of the IPFs for rotations have a region of noticeably higher concentration. No tilt series can be identified from the template matching result, and the FFC method fails in determining the orientations from the experimental GaSb dataset.

5.5.4 NCC on experimental mordenite

For NCC on the experimental mordenite dataset, seen in Figure 4.30, there are no areas where subsequent indices are aligned on the inverse pole figure. The plots of selected templates together with the signal show that the template matches the signal only in a few high intensity positions, sometimes just a single position, as for the yellow rectangle representing index 38. The plots of rotations on the IPF do not look like those of tilt series, as rotations are spread all over the inverse pole figure, while we would expect there to be a an area of increased density representing the constant tilt of the tilt series. The NCC method fails in determining the orientations from the experimental mordenite dataset.

5.5.5 ZNCC on experimental mordenite

Considering ZNCC on the experimental mordenite dataset, seen in Figure 4.31, the results again resemble those of NCC. There are no areas where subsequent indices are aligned on the inverse pole figure, but there are certain subsequent indices that have the same or almost the same position in both the x- and z-projection, such as indices 15-17 and 31-33. As the orientations are not aligned, but rather identified to the same area, there could be a common high intensity reflection for the indices in question, that would give a higher correlation score for templates from a given area. From the plots of templates together with signal we observe that the highest-scoring templates do not resemble the signal for any of the three selected indices. The plot of rotations on the IPF does not resemble a tilt series, as no single position has a higher concentration of rotations. The ZNCC method fails in determining the orientations from the experimental mordenite dataset.

5.5.6 FFC on experimental mordenite

Moving on to FFC on the experimental mordenite dataset, seen in Figure 4.32, all orientations are projected onto 5 points on the x-projection of the IPF and 3 points on the z-projection of the IPF. All points are in close vicinity to each other. This is a clear indication that the method has not worked as intended, as the data spans a range of 100 degrees, which would correspond to a much larger area of the IPF being covered. The plots of templates together with signal for selected indices show that the selected templates look very different from the signal. The inverse pole figures showing the rotations have a high intensity in the [100] corner for the x-projection and a high intensity in the [001] corner for the z-projection. This indicates that many subsequent indices are identified to be the same orientation. This could again indicate that many of indices share a common trait that is picked up by the FFC method, but not the trait the method is intended to identify. The FFC method fails to determine the orientations in the experimental mordenite dataset.

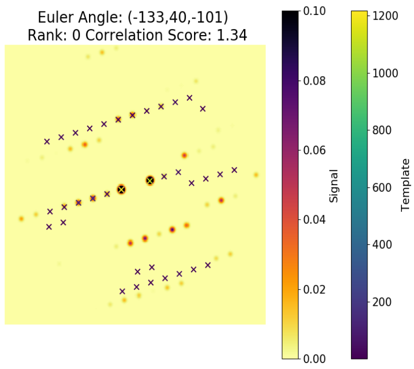
5.6 Comparison of correlation methods

Considering Table 5.1, the findings of this work show that for GaSb and mordenite, the NCC method performs well for perfect templates on simulated data and for GaSb with a simulated data input of 1 degree resolution. The ZNCC method gives in general the same or very similar result as NCC especially for 5 of 6 test cases. ZNCC performs slightly worse for simulated GaSb with a template library of 1 degree. FCC performs worse than NCC and ZNCC for all tests but can correctly identify orientations for simulated data of both GaSb and mordenite with a perfect template library. For Experimental data of both GaSb and mordenite, all tested correlation methods fail, as discussed in Section 5.5.

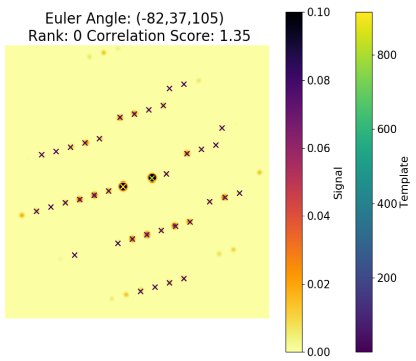
To further examine the correlation methods, specific example orientations are selected for every method, and further elaborated on to provide an understanding of what might be the underlying cause that a template is misindexed.

5.6.1 NCC

Considering now the orientation in the red frame in Figure 4.10b. This orientation lies in an area that is identified as difficult to index for the NCC algorithm, based on a set of 1000 random orientations, as the bright color in the area of the red frame on the inverse pole figure represents. By creating a template from the true orientation, and match this template against the simulated signal, the perfect match is found. The template found in template matching with a 1 degree grid and the perfect match template are plotted on the same signal in Figure 5.7. Considering the image of the 1 degree template, a line of reflections in the center of the template matches the signal well. In addition there are two points in the top half of the pattern match the underlying signal. None of the intensities in the lower half of the pattern are matched. This gives a correlation score of 1.34 for the template, as reported in the title above the template on signal plot. For the perfect template the template and the signal coincide for all reflections, by construct. This results is a correlation score of 1.35.



(a)



(b)

Figure 5.7: The orientation marked with a red square in Figure 4.10b is plotted with **(a)** the best fitting template from a template bank of 1 degree resolution and **(b)** the perfect template.

It is important to notice that the colorbar of the template from the 1 degree library has a maximum value of around 1200, while the colorbar of the perfect template has a maximum value of around 900. As the high intensity reflections of the 1 degree template coincide perfectly with the two highest intensity spots in the signal, this constitutes the major part of the correlation score. Notice also that all other intensities in the template are much weaker than the two central spots. For the perfect template, the fraction of the correlation score stemming from the two central spots will be smaller, as more points coincide with the signal. If the template is not perfect, but close to to the true orientation, the positions of the reflections will be slightly shifted, and the correlation score lower. Meanwhile, a template from another region of orientation space achieves a correlation score close to that of the perfect template by only matching the signal in the center, with spots of very high intensity, as seen in Figure 5.8. This could explain why the are on the inverse pole figure from which the signal is collected is hard to match; There are other areas on the inverse pole figure where the diffraction patterns have reflections of higher intensity in the center, and these reflections coincide with the high intensity reflections of the signal of the selected orientation. Hence, the correlation score on its own can be a poor parameter to determine if the indexation is correct or not.

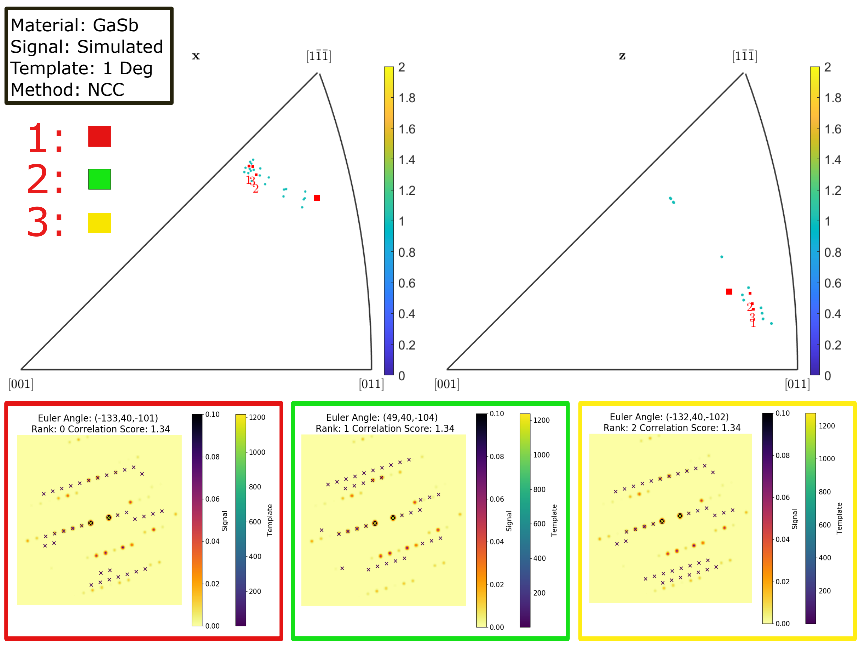
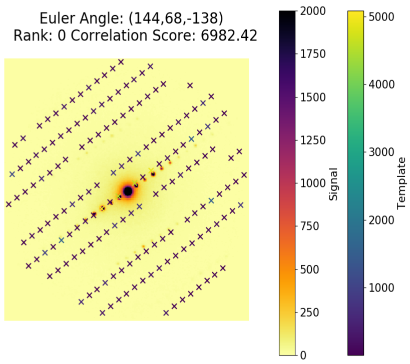
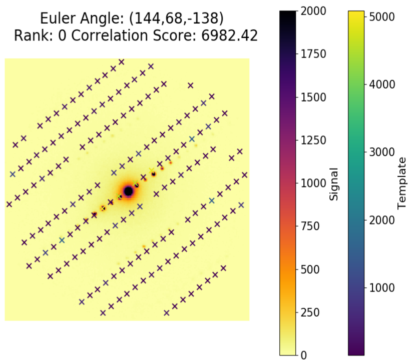


Figure 5.8: The orientation marked with a red square in Figure 4.10b is plotted as a big red square on the inverse pole figure in X and Z directions together with the 25 best fits, shown as blue dots on the IPFs. The three best fitting templates are shown in colored frames. The numbers show the position among the best fitting templates, and to which frame each template belongs.

This observation points to a general problem of both the NCC and the ZNCC methods. As the formulae, expressed in Equations (2.51) and (2.52), for computing the correlation score is a product function of the signal intensity and the template intensity, the correlation score is easily dominated by high intensity reflections in the signal that coincide with high intensity reflections in the template. The effect of matching the low intensity reflections becomes negligible. This is in part accounted for by dividing with the norm of the template intensities, but the effect is still clear for the example in Figure 5.7. To further analyze how NCC performs, Figure 5.9 shows the orientations marked by a green and a red square in Figure 4.30a. For the signal in Figure 5.9a, there are four high intensity template entries that coincide well with high intensity positions in the signal, close to the center. Apart from the line in the center, a so-called systematic row, there is not much visible signal, but the template has light blue entries with medium level intensity spread across the pattern that do not correspond to any visible underlying signal. For the signal in Figure 5.9b the template only coincides with the signal for two high intensity positions near the center. Interestingly, the scale of the colorbar representing the intensity only goes to 2500, while the colorbar of the template in Figure 5.9a goes to 5000. Thus, we can expect there to be templates with a higher intensity in the template bank, that could also coincide with reflections in the signal. This indicates that division by the template norm is indeed working as intended, for normalizing templates. Still, the effect that the high intensity template entries dominate the result is clear for both images, and a template with a larger fraction of its total intensity located at a few intensity maxima, can dominate the correlation score in another region of orientation space, as seen in Figure 5.7.



(a)

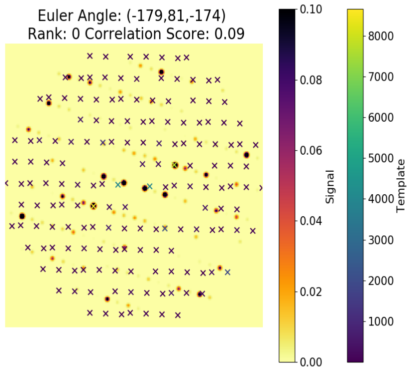


(b)

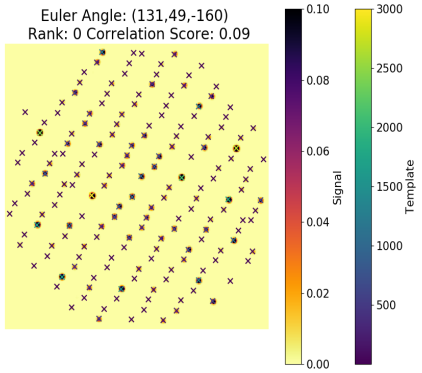
Figure 5.9: The template together with signal for the orientation marked with **(a)** a green square and **(b)** a red square in Figure 4.30a.

5.6.2 ZNCC

ZNCC shares many of the same attributes as NCC, but is invariant under constant change in image brightness [52]. This makes it easier to compare correlation scores from different experimental diffraction patterns and between different experiments. For the testing on simulated data, the NCC and ZNCC methods provided similar results for 5 of 6 tests. The greatest difference was for simulated mordenite data, where NCC was able to correctly index 54.3 % of the orientations, while ZNCC correctly indexed 30.3 %, as seen in Table 4.6. Figure 5.10 shows the template together for the signal for the orientations marked with a green (a) and yellow (b) square in Figure 4.28, which will be studied further. As is seen in the title above the diffraction patterns, both templates receive the same correlation score of 0.09, even though one is correctly indexed (b), while the other (a) is far from the true orientation. For the template and signal in Figure 5.10a the highest intensity templates match well with two strong reflections in the signal, one up and to the right from the center, and one down and to the left. Otherwise, no template reflections coincide with the signal. For the template and signal in Figure 5.10b, almost every template intensity coincides with a reflection in the signal. This shows that the ZNCC function has the same issue as NCC, as a few high intensity reflections can dominate the correlation score, while low intensity reflections become irrelevant.



(a)



(b)

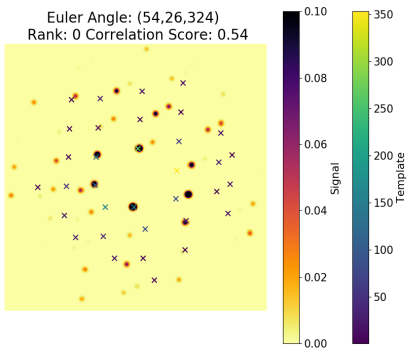
Figure 5.10: The template together with signal for the orientations marked with **(a)** a green square and **(b)** a yellow square in Figure 4.28b.

The current model of simulation, described in Appendix B.2.2, tends to provide two very strong reflections, as seen in Figure 5.10. This, together with correlation functions for NCC and ZNCC that multiply the template intensity with the signal intensity, clearly weights high intensity reflections much more than low intensity reflections. The precession technique for collecting signals ensures more illumination and also distributes the illumination more uniformly. Therefore, taking data with the PED technique could improve the results for both NCC and ZNCC, as the peaks in the signal will be stronger. The same effect can be achieved by incorporating precession in the simulation of templates. Another option is to demand from a match result that a minimum number of points match the unerlying signal. This will remove templates with high intensity matches for only two reflections from consideration. Furthermore, introducing a function such as the log function or square root function in Equations (2.51) and (2.52), to map the correlation scores from a single reflection onto a more flat distribution can be considered. Another option is to perform the cross-correlation in another space, which is done in FFC. Here, both the templates and the signal are transformed into Fourier space, where the correlation computations are performed.

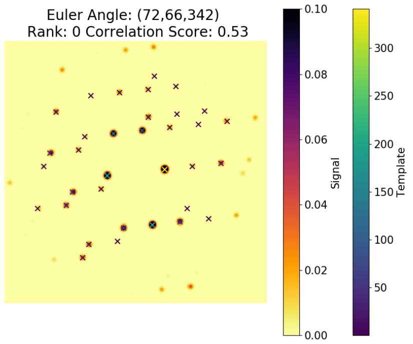
5.6.3 FFC

FFC is a completely different method from NCC and ZNCC. For the cases of perfect templates for simulated GaSb and mordenite tilt series, the FFC method performed well, and it was proved that it can distinguish between orientations in a tilt series in Figures 4.4c and 4.21c. With a template library of 3 degree resolution the performance was not good, as only 3.1 % of random GaSb orientations and 3 % of random mordenite orientations, as seen in Tables 4.1 and 4.4. This is in part expected, as the template library had a lower resolution due to memory constraints, but also because the FFC method is meant to be followed with a refinement procedure, as described in [54], where the objective of the FFC step is to match an orientation to within 7 degrees of the true orientation, as a starting point for the subsequent refinement. Considering Figure 4.8c, it is clear that only 9 out of 30 orientations are confined to within 7 degrees of the true orientation. Considering now the cases of templates 10 and 22 from the perfect template bank analysis of the GaSb tilt series, discussed in section 5.1. This is an ideal case for testing, as two patterns that are identical apart from a mirror symmetry behave differently, in that one is correctly indexed, while the other is not, as seen in Figure 4.4c.

Templates for both orientations are plotted together with the signal in Figure 5.11. Although the template in (a) does not fit the signal, it still gets a higher correlation score than the perfect template for the signal in (b). In this case, the cause of error is very different from what has been shown for NCC and ZNCC. To further examine this, the Fourier transform has to be considered.



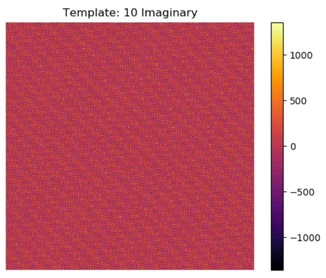
(a)



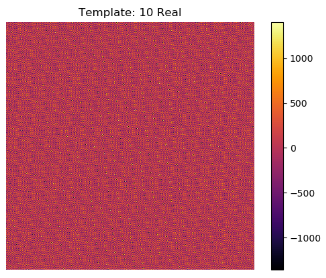
(b)

Figure 5.11: The template together with signal for orientations 10 and 22 from the tilt series of GaSb with FFC on perfect template.

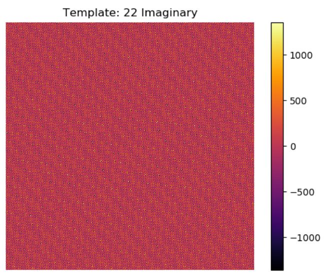
Figure 5.12 shows the real and imaginary parts of the Fourier transforms of orientations 10 and 22 in the tilt series. The intensity scale is the same for all images. There intensities rapidly oscillate between the minimum and maximum values for all the subfigures. This could be because of the sharp peaks in the templates, of only one pixel width. Comparing now with Figure 5.13 where the real and imaginary parts of the simulated signal are shown. Again the scales of the colorbars are exactly the same. It is very hard to extract relevant information from such a plot. In Equation (2.53) the Hadamard product between the Fourier transform of the template and the Fourier transform of the signal is inverse Fourier transformed to provide a correlation score, the max value in the resulting matrix. It is clear that following the steps in the process with visualization tools developed for point based correlation methods does not suffice, and new ways of portraying the process will have to be developed in order to better understand the FFC process. An educated guess is that the results would improve if the templates were made more similar to actual signals, by simulating reflections larger than just one pixel. This method of simulating patterns is very specialized towards the point based methods, NCC and ZNCC, and could be a showstopper for methods performing the correlation in other domains.



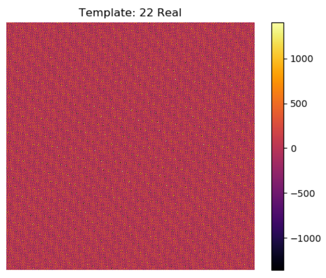
(a)



(b)



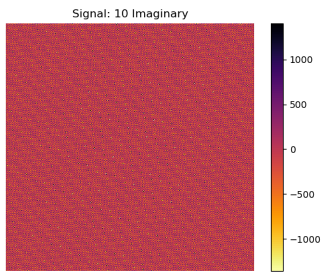
(c)



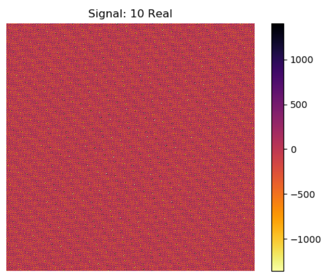
(d)

Figure 5.12: Real and imaginary parts of the Fourier transforms of the templates for orientations 10 and 22 for FFC on the tilt series of GaSb with a perfect template.

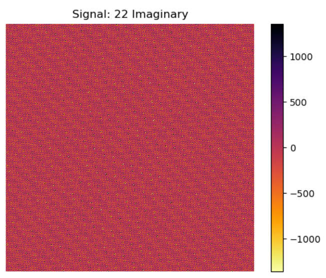
Considering that NCC is the most computationally efficient method and provides the most accurate results, this method should be applied when running template matching for GaSb and mordenite, given the current model of simulation. As all methods have been proven to work for ideal cases, it is likely that both ZNCC and FFC can be tuned to work well, by considering simulation parameters, and also the model of simulation.



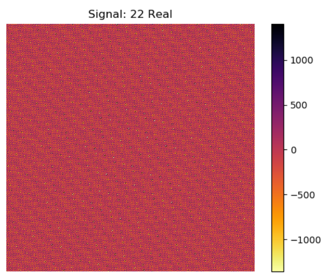
(a)



(b)



(c)



(d)

Figure 5.13: Real and imaginary parts of the Fourier transforms of the simulated signal for orientations 10 and 22 for FFC on the tilt series of GaSb with a perfect template.

Chapter 6

Conclusion

Two new methods for correlating simulated templates to signals; Zero-normalized cross-correlation (ZNCC) and full-frame correlation (FFC) were implemented and tested together with the standard method, normalized cross-correlation (NCC) that was present in pyXem at the start of the project, and is the common routine for template matching for scanning electron diffraction data stacks. The test cases were simulated and experimental tilt series data from GaSb and mordenite. This work demonstrated that all these three methods are able to distinguish between signals from different orientations for the ideal case where every signal has a perfect fit in the template bank. When the orientations were taken from a grid of constant angular spacing on the fundamental zone, the NCC method provided the best results for both GaSb and mordenite for simulated data. The NCC method is also the fastest method, and requires less memory than FFC.

Experimental data were tilt series of GaSb and mordenite taken with the RED technique. None of the methods studied in this work were able to correctly index the GaSb or the mordenite tilt series for template banks of 1 degree resolution, when the max excitation error was set to 0.025. Results can be improved by precessing the electron beam when taking the tilt series instead of using the RED technique, to reduce dynamical effects.

For the evaluation and comparison of the correlation algorithms, a new reliability metric was proposed: The weighted reliability index (WRI). It showed a significantly stronger relationship with the angular distance from the true value for simulated data than that of the well established reliability index. For all three correlation methods and for both GaSb and mordenite the weighted reliability index showed an average correlation coefficient of -0.36 with the angular distance from the true value. This means that the weighted reliability index better predicts if a template matching result is close to the true value. This reliability analysis can also be used to analyze other aspects of the total workflow than correlation, such as data acquisition, preprosessing or the simulation parameters.

The analysis of the different test cases provided in this report has revealed an issue in the functionality applied for the template matching procedure within pyXem, including the orientation list not spanning the fundamental zone. The orientation list has a bug for cubic crystal structures, where it will contain symmetrically equivalent orientations from different fundamental zones, while it does not span the full fundamental zone. The issue has been reported at the pyxem/diffsims discussion page on github[1].

There is still a need for further development and testing before the template matching procedure is applicable for tilt series. In this work, the functionality related to correlating template diffraction patterns to simulated and experimental patterns has been systematically analyzed. For different cases the results show that the NCC method already present in pyXem provides the best results for the test cases. It is clear that for all the methods certain templates in the template library are chosen as the best fit for diffraction patterns with a very different diffraction profile than that of the signal. For NCC and ZNCC this happens when the central intensities in the experimental diffraction pattern coincide best with the template, the contribution from low intensity reflections is limited. For FFC it has yet to be determined what features in a pattern frame dominate the correlation and the choice of a potentially erroneous orientation. Possible improvements to the template matching procedure include acquiring precessed electron diffraction, which is more kinematic like in appearance and afterwards account for precession in the simulation model.

The code developed in this project is made available in version 0.11.x of the open source python library pyXem. Computations of the WRI reliability metric was done within the framework of pyxem 0.11.0, but it does not have a dedicated function integrated in the library. The code for computing this metric, and all other analysis performed in this project is made available in the Appendix.

[1]Issue # 93

Chapter 7

Future Work

The results and discussion in Chapters 4 and 5, show that NCC, ZNCC and FFC all work in ideal cases, and should be continously tested and evaluated as the template matching procedure develops. This work has considered the correlation step, and has also introduced a new metric for measuring the reliability of a template matching result.

7.1 Improvements to the simulation model

For the point based correlation methods NCC and ZNCC, this work revealed that a few high intensity reflections can dominate the correlation score, and can cause templates that only coincide with the signal in for two or three reflections to be selected as the best match. One of the causes for this was that the simulation model often generates templates with two very strong reflections. By introducing precession in the simulation model, templates will have more intensity in general, and also more spread of intensity between the reflections.

7.2 Taking experimental data with PED

In this work, data collection was done with the RED technique. For template matching it might be better if the electron beam is precessed when data is taken, to give more intensity in the signal, and intensity that is distributed across more reflections. This should especially effect the point based correlation methods NCC and ZNCC.

7.3 Developing visualization tools to follow the FFC process

For signal and templates plotted in the direct domain, visualization tools such as plotting the template together with the signal are useful for evaluating the methods, and spotting sources of error. For methods like the FFC method, that makes use of other domains, it is harder to follow the steps of the algorithm, as discussed in section 5.6.3. It was demonstrated that FFC can distinguish between orientations in a tilt series, and that it fails for a equally spaced template library, but analyzing the underlying mechanisms causing the method to fail is hard without proper analysis tools. Future work should focus on developing tools that can identify and illustrate the steps of the FFC method, for both transparency in the workflow and better intuition among users.

7.4 Confusion matrix for the WRI

In section 5.4 the correlation coefficient between WRI and the angular distance from the true value was computed, and the Confusion matrix was created for a test case of 1000 random orientations, and a threshold value of 0.9. Future work should focus on doing more extensive and systematic testing, to find optimal threshold values for WRI.

7.5 Further development of tilt series testing

In section 2.6 a method for evaluating tilt series results with a roughly known tilt relation between subsequent orientations was proposed. In this work a simplified version of this method has been applied. A future project should implement the full and more correct treatment, to better evaluate template matching results from experimental tilt series data.

Bibliography

[1] V. L. Colvin. The potential environmental impact of engineered nanomaterials. *Nature Biotechnology*, 2003.

[2] J. J. Baumberg. Breaking the mould: Casting on the nanometre scale. *Nature Materials*, 2006.

[3] M. de Graef. *Introduction to conventional transmission electron microscopy*. Cambrige university press, 2003.

[4] R. Courtland. The microscope revolution that's sweeping through materials science. *Nature*, 2018.

[5] W. L. Bragg. The structure of some crystals as indicated by their diffraction of x-rays. *Proceedings of the Royal Society of London.*, 1913.

[6] J. D. Watson, F. H. C. Crick, et al. Molecular structure of nucleic acids. *Nature*, 1953.

[7] W. Dai, B. Zhang, H. Su, J. Li, Y. Zhao, X. Xie, Z. Jin, F. Liu, C. Li, Y. Li, et al. Structure-based design of antiviral drug candidates targeting the sars-cov-2 main protease. *Science*, 2020.

[8] G. Mollenstedt. 4.2 my early work on convergent-beam electron diffraction. *Advances in imaging and electron physics*, 1996.

[9] L. Palatinus, P. Brázda, P. Boullay, O. Perez, M. Klementová, S. Petit, V. Eigner, M. Zaarour, and S. Mintova. Hydrogen positions in single nanocrystals revealed by electron diffraction. *Science*, 2017.

[10] R. F. Service. 2018 breaktrough of the year. *Science*, 2018.

[11] J. Dubochet, J. Frank, and R. Henderson. Press release: The nobel prize in chemistry 2017. *NobelPrize.org*, 2018.

[12] W. Wan, J. Sun, J. Su, S. Hovmöller, and X. Zou. Three-dimensional rotation electron diffraction: software red for automated data collection and data processing. *Journal of applied crystallography*, 2013.

[13] P. A. Midgley and A. S. Eggeman. Precession electron diffraction–a topical review. *IUCrJ*, 2015.

[14] R. Vincent and P. A. Midgley. Double conical beam-rocking system for measurement of integrated electron diffraction intensities. *Ultramicroscopy*, 1994.

[15] E. F. Rauch and L. Dupuy. Rapid spot diffraction patterns idendification through template matching. *Archives of Metallurgy and Materials*, 2005.

[16] E. F. Rauch, J. Portillo, S. Nicolopoulos, D. Bultreys, S. Rouvimov, and P. Moeck. Automated nanocrystal orientation and phase mapping in the transmission electron microscope on the basis of precession electron diffraction. *Zeitschrift fur Kristallographie*, 2010.

[17] P. Moeck, S. Rouvimov, E. F. Rauch, M. Véron, H. Kirmse, I. Häusler, W. Neumann, D. Bultreys, Y. Maniette, and S. Nicolopoulos. High spatial resolution semi-automatic crystallite orientation and phase mapping of nanocrystals in transmission electron microscopes. *Crystal research and technology*, 2011.

[18] M. Gemmi, E. Mugnaioli, T. E. Gorelik, U. Kolb, L. Palatinus, P. Boullay, S. Hovmuller, and J. P. Abrahams. 3d electron diffraction: The nanocrystallography revolution. *ACS Central Science*, 2019.

[19] P. Oleynikov, S. Hovmöller, and X. D. Zou. Precession electron diffraction: Observed and calculated intensities. *Ultramicroscopy*, 2007.

[20] K. Gjønnes. On the integration of electron diffraction intensities in the vincent-midgley precession technique. *Ultramicroscopy*, 1997.

[21] R. A. Adams and C. Essex. *Calculus 2*. Pearson Education Limited, 2013.

[22] W. H. Miller. *A Treatise on Crystallography*. J & J.J Deighton, 1839.

[23] K. Rottmann. *Matematisk Formelsamling*. Spektrum Forlag, 2003.

[24] J. Linder. Intermediate quantum mechanics, 2017.

[25] F. Logiurato. Relativistic derivations of de broglie and planck-einstein equations. *arXiv preprint arXiv:1208.0119*, 2012.

[26] M. Born and E. Wolf. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*. Elsevier, 2013.

[27] P. C. Hemmer. *Kvantemekanikk*. Fagbokforlaget Vigmostad & Bjorke AS, 1980.

[28] Z. U. Yusuf. The relation between wave vector and momentum in quantum mechanics. *Optik*, 2016.

[29] A. Rother and K. Scheerschmidt. Relativistic effects in elastic scattering of electrons in tem. *Ultramicroscopy*, 2009.

[30] S. Valero, P. Lapostolle, A. M. Lombardi, N. Pichoff, and E. Tanke. An analytical approach to the poisson equation in 3-dimensional space charge problems. In *Proceedings of the 1999 Particle Accelerator Conference (Cat. No. 99CH36366)*, 1999.

[31] U. Shmueli. *International Tables for Crystallography, Volume B: Reciprocal Space*. Springer Science & Business Media, 2008.

[32] R. N. Bracewell and R. N. Bracewell. *The Fourier transform and its applications*. McGraw-Hill New York, 1986.

[33] G. A. Venema. *Foundations of geometry, second edition*. Pearson, 2012.

[34] M. J. Cooper. Compton scattering and electron momentum determination. *Reports on Progress in Physics*, 1985.

[35] M. Von Heimendahl, W. Bell, and G. Thomas. Applications of kikuchi line analyses in electron microscopy. *Journal of applied physics*, 1964.

[36] J. M. Cowley and A. F. Moodie. The scattering of electrons by atoms and crystals. iii. single-crystal diffraction patterns. *Acta Crystallographica*, 1959.

[37] H. Kohl and L. Reimer. *Transmission electron microscopy: physics of image formation*. Springer, 2008.

[38] H. Klein. Precession electron diffraction of Mn_2O_3 and $PbMnO_{2.75}$: solving structures where x-rays fail. *Acta Crystallographica Section A: Foundations of Crystallography*, 2011.

[39] A. S. Eggeman and P. A. Midgley. Precession electron diffraction. *Advances in Imaging and Electron Physics*, 2012.

[40] L. Palatinus, D. Jacob, P. Cuvillier, M. Klementová, W. Sinkler, and L. D. Marks. Structure refinement from precession electron diffraction data. *Acta Crystallographica Section A: Foundations of Crystallography*, 2013.

[41] D. Zhang, P. Oleynikov, S. Hovmöller, and X. Zou. Collecting 3d electron diffraction data by the rotation method. *Zeitschrift für Kristallographie International journal for structural, physical, and chemical aspects of crystalline materials*, 2010.

[42] D. Xie, C. Baerlocher, and L. B. McCusker. Combining precession electron diffraction data with x-ray powder diffraction data to facilitate structure solution. *Journal of Applied Crystallography*, 2008.

[43] D. Rowenhorst, A. D. Rollett, G. S. Rohrer, M. Groeber, M. Jackson, P. J. Konijnenberg, and M. de Graef. Consistent representations of and conversions between 3d rotations. *Modelling and Simulation in Materials Science and Engineering*, 2015.

[44] H. J. Bunge. *Texture analysis in materials science: mathematical methods*. Elsevier, 2013.

[45] H. Goldstein, C. Poole, and J. Safko. Classical mechanics, 2002.

[46] J. P. Snyder. *Flattening the earth: two thousand years of map projections*. University of Chicago Press, 1997.

[47] C. Hammond and C. Hammond. *The basics of cristallography and diffraction*, volume 214. Oxford, 2001.

[48] G. Nolze. Euler angles and crystal symmetry. *Crystal Research and Technology*, 2015.

[49] W. K. Pratt. Image detection and registration. *Digital image processing*, 2001.

[50] L. N. Kanal and N. C. Randall. Recognition system design by statistical analysis. In *Proceedings of the 1964 19th ACM national conference*, 1964.

[51] D. N. Johnstone, P. Crout, J. Laulainen, S. Høgås, B. Martineau, T. Bergh, S. Smeets, S. Collins, J. Morzy, H. Ånes, E. Prestat, T. Doherty, T. Ostasevicius, M. Danaie, and R. Tovey. pyxem/pyxem: pyxem 0.11.0, November 2020.

[52] A. Nakhmani and A. Tannenbaum. A new distance measure based on generalized image normalized cross-correlation for robust video tracking and image recognition. *Pattern recognition letters*, 2013.

[53] N. Roma, J. Santos-Victor, and J. Tomé. A comparative analysis of cross-correlation matching algorithms using a pyramidal resolution approach. In *Empirical Evaluation Methods in Computer Vision*. World Scientific, 2002.

[54] A. Foden, D. M. Collins, A. J. Wilkinson, and T. B. Britton. Indexing electron backscatter diffraction patterns with a refined template matching approach. *Ultramicroscopy*, 2019.

[55] G. D. Bergland. A guided tour of the fast fourier transform. *IEEE spectrum*, 1969.

[56] F. Ram, S. Wright, S. Singh, and M. De Graef. Error analysis of the crystal orientations obtained by the dictionary approach to ebsd indexing. *Ultramicroscopy*, 2017.

[57] R. E. Walpole, R. H. Myers, S. L. Myers, and K Ye. *Probability and statistics for Engineers and Scientists*. Pearson, 2012.

[58] C. G. Khatri and K. V. Mardia. The von mises–fisher matrix distribution in orientation statistics. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1977.

[59] H. Schaeben. “normal” orientation distributions. *Texture, Stress, and Microstructure*, 1992.

[60] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 2005.

[61] K. V. Mardia and P. E. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009.

[62] V. S. Teodorescu and M. G. Blanchin. Fast and simple specimen preparation for tem studies of oxide films deposited on silicon wafers. *Microscopy and Microanalysis*, 2009.

[63] S. Smeets and M. O. Cichocka. Continuous rotation electron diffraction data for Zeolite Mordenite, July 2018.

[64] M. O. Cichocka, J. Ångström, B. Wang, X. Zou, and S. Smeets. High-throughput continuous rotation electron diffraction data acquisition via software automation. *Journal of applied crystallography*, 2018.

[65] T. Hahn, U. Shmueli, and J. C. W. Arthur. *International tables for crystallography*, volume 1. Reidel Dordrecht, 1983.

[66] Gasb crystal structure: Datasheet from pauling file multinaries edition 2012 in springermaterials.

[67] V. Gramlich. Untersuchung und verfeinerung pseudosymmetrischerstrukturen. *ETH Library, Zurich, Switzerland*, 1971. [In German].

[68] E. Opheim. Evaluating template matching for simulated kinematic electron difraction. *NTNU Project Thesis*, 2019.

[69] G. Nolze and R. Hielscher. Orientations–perfectly colored. *Journal of Applied Crystallography*, 2016.

[70] M. Själander, M. Jahre, G. Tufte, and N. Reissmann. EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure, 2019.

[71] P. Juhás, C. L. Farrow, X. Yang, K. R. Knox, and S. J. L. Billinge. Complex modeling: a strategy and software program for combining multiple information sources to solve ill posed structure and nanostructure inverse problems. *Acta Crystallographica Section A*, 2015.

[72] G. Friedel. Sur les symétries cristallines que peut révéler la diffraction des rayons röntgen. *CR Acad. Sci. Paris*, 1913. In French.

[73] C. J. Boehlert, S. C. Longanbach, M. Nowell, and S. Wright. The evolution of grain-boundary cracking evaluated through in situ tensile-creep testing of udimet alloy 188. *Journal of Materials Research*, 2008.

[74] Y. Xun, M. J. Tan, and T. G. Nieh. Grain boundary characterisation in superplastic deformation of al-li alloy using electron backscatter diffraction. *Materials science and technology*, 2004.

[75] M. A. Groeber, B. K. Haley, M. D. Uchic, D. M. Dimiduk, and S. Ghosh. 3d reconstruction and characterization of polycrystalline microstructures using a fib–sem system. *Materials Characterization*, 2006.

[76] A. Morawiec, E. Bouzy, H. Paul, and J. J. Fundenberger. Orientation precision of tem-based orientation mapping techniques. *Ultramicroscopy*, 2014.

[77] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 2006.

[78] A. Morawiec. *Orientations and rotations*. Springer, 2003.

[79] T. Lee. Bayesian attitude estimation with the matrix fisher distribution on so (3). *IEEE Transactions on Automatic Control*, 2018.

[80] P. Deuflhard. *Newton methods for nonlinear problems: affine invariance and adaptive algorithms*, volume 35. Springer Science & Business Media, 2011.

[81] A. L. Schwab and J. P. Meijaard. How to draw euler angles and utilize euler parameters. In *ASME 2006 international design engineering technical conferences and computers and information in engineering conference*. American Society of Mechanical Engineers, 2006.

[82] R. Krakow, R. J. Bennett, D. N. Johnstone, Z. Vukmanovic, W. Solano-Alvarez, S. J. Lainé, J. F. Einsle, P. A. Midgley, C. M. F. Rae, and R. Hielscher. On three-dimensional misorientation spaces. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2017.

[83] I. Lobato and D. van Dyck. An accurate parameterization for scattering factors, electron densities and electrostatic potentials for neutral atoms that obey all physical constraints. *Acta Crystallographica Section A: Foundations and Advances*, 2014.

Appendices

Appendix A

Mathematical proofs

In this section we will outline some proofs omitted in the text because they might be of interest to a limited fraction of the readers, and also of a certain length.

A.1 Equation (2.12)

We start of examining Equation (2.12). We have stated that for parallel planes the left hand side of the Equation will remain the same, while the right hand side changes. For the purposes of this proof we will consider parallel planes to be planes with parallel normal vectors. We consider two parallel planes A and B as seen in Figure A.1. Plane A intersects the basis vectors $\mathbf{a}_i$ in the points s_1, s_2 and s_3. Plane B intersects the basis vectors in the points s'_1, s'_2 and s'_3. The distance d from the origin to a plane along the plane normal, **n** can be found by

$$d_A = \boldsymbol{n} \cdot [Os_1] = \boldsymbol{n} \cdot [Os_2] = \boldsymbol{n} \cdot [Os_3]$$

and

$$d_B = \boldsymbol{n} \cdot [Os'_1] = \boldsymbol{n} \cdot [Os'_2] = \boldsymbol{n} \cdot [Os'_3]$$

where O is the origin, and **n** is a unit vector in the direction of the plane normal. We make use of Equation (2.3) to write

$$\frac{d_A}{d_B} = \frac{|Os_1|\cos\theta_1}{|Os'_1|\cos\theta_1} = \frac{|Os_1|}{|Os'_1|} = \frac{s_1}{s'_1}$$

where θ_1 is the angle between the normal vector and the basis vector in direction 1, that is, the axis on which s_1 and s'_1 lies. Likewise it can be shown that

$$\frac{s_1}{s'_1} = \frac{s_2}{s'_2} = \frac{s_3}{s'_3} = \frac{d_A}{d_B} \tag{A.1}$$

We now consider Equation (2.12) for plane A. We let $\frac{d_A}{d_B} = C$, substitute s_i for Cs'_i and get

$$\frac{x}{s'_1} + \frac{y}{s'_2} + \frac{z}{s'_3} = C. \tag{A.2}$$

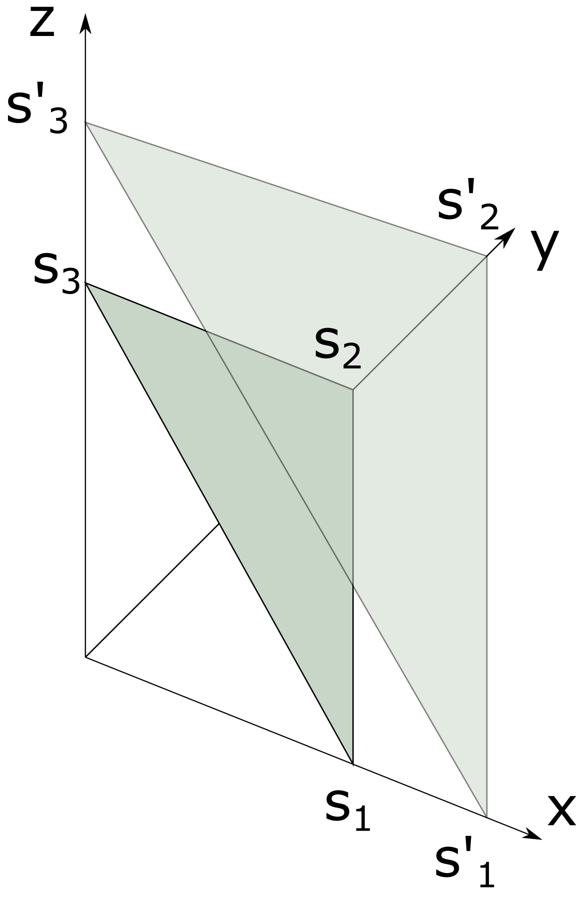


Figure A.1: Two planes, with s_i and s'_i defined along the coordinate axes.

The only assumption we have made for the planes is that they are parallel. This proof holds as long as $d_A \neq 0$. Should this be the case we can interchange the names of planes A and B and proceed with our proof. We have shown that for translation of parallel planes along the plane normal the left side of Equation (2.12) remains unchanged, whereas the right side is changed by a factor depending on the relative change in the distance from the origin to the plane.

A.2 Maximum likelihood estimate for the Fisher - Matrix distribution

The von Mises - Fisher distribution is the analogue of the normal distribution for modelling the spread of random misorientations [76]. Given a set of rotation matrices R_1, R_2, ..., R_k, the objective is to determine the maximum likelihood estimate for the parameters of the Fisher Matrix distribution.

The probability density function for the generalized Fisher Matrix distribution is given by

$$F_{vMF}(R; F) = \frac{exp\{tr(F^T R)\}}{c(F)} \tag{A.3}$$

where R is a 3x3 rotation matrix, F is a parameter matrix and c(F) is a normalizing constant [78]. Both [78] and [58] report the normalizing constant to be

$$c(F) = {}_0F_1(\frac{d}{2}; \frac{FF^T}{4}) = \int_{SO(3)} exp\{tr(F^T R)\}dR \tag{A.4}$$

where ${}_0F_1$ is the hypergeometric function of matrix argument and d is the dimension of the rotation matrix. For crystallographic textures we are interested in the case d = 3.

We now define the parameter

$$\bar{R} = \frac{\sum_{i=1}^{k} R_i}{N}. \tag{A.5}$$

Following section four of [58], the parameter F can be uniquely decomposed as

$$F = \Delta D_\phi \Gamma \tag{A.6}$$

given that the first row of Δ is non-negative and F is of rank d. It is also assumed that the d largest eigenvalues of $\bar{R}\bar{R}^T$ are g_i^2 so that $g_1^2 > g_2^2 > g_3^2$. Let D_g be the diagonal matrix with entries g_1, g_2 and g_3. The maximum likelihood estimators of Δ and Γ are then given by the singular value decomposition of $\bar{R}$:

$$\bar{R} = \Delta D_g \Gamma. \tag{A.7}$$

The matrix D_ϕ is a diagonal matrix, where the maximum likelihood estimate for the i-th entry is given by:

$$\frac{1}{c(D_\phi)} \frac{\partial c(D_\phi)}{\partial \phi_i} - g_i = 0. \tag{A.8}$$

In order to solve Equation A.8, the following equations will be useful, given that Δ and Γ are proper rotation matrices [79]:

$c(D_\phi)$ can be evaluated by a 1-D integral as

$$\begin{aligned} c(D_\phi) = \int_{-1}^{1} & \frac{1}{2} I_0[\frac{1}{2}(\phi_i - \phi_j)(1-u)] \\ & \times I_0[\frac{1}{2}(\phi_i + \phi_j)(1+u)] exp\{\phi_k u\} du, \end{aligned} \tag{A.9}$$

where I_i is the Bessel function of the first kind and order i.

Furthermore, the derivatives of $c(D_\phi)$ can be evaluated as

$$\begin{aligned}\frac{\partial c(D_\phi)}{\partial \phi_i} &= \int_{-1}^{1} \frac{1}{4}(1-u)I_1[\frac{1}{2}(\phi_i-\phi_j)(1-u)] \\ &\times I_0[\frac{1}{2}(\phi_i+\phi_j)(1+u)]exp\{\phi_k u\} \\ &+ \frac{1}{4}(1+u)I_0[\frac{1}{2}(\phi_i-\phi_j)(1-u)] \\ &\times I_1[\frac{1}{2}(\phi_i+\phi_j)(1+u)]\, exp\{\phi_k u\}\, du,\end{aligned} \tag{A.10}$$

$$\begin{aligned}\frac{\partial c(D_\phi)}{\partial \phi_j} &= \int_{-1}^{1} -\frac{1}{4}(1-u)I_1[\frac{1}{2}(\phi_i-\phi_j)(1-u)] \\ &\times I_0[\frac{1}{2}(\phi_i+\phi_j)(1+u)]exp\{\phi_k u\} \\ &+ \frac{1}{4}(1+u)I_0[\frac{1}{2}(\phi_i-\phi_j)(1-u)] \\ &\times I_1[\frac{1}{2}(\phi_i+\phi_j)(1+u)]\, exp\{\phi_k u\}\, du\end{aligned} \tag{A.11}$$

and

$$\begin{aligned}\frac{\partial c(D_\phi)}{\partial \phi_k} &= \int_{-1}^{1} \frac{1}{2}I_0[\frac{1}{2}(\phi_i-\phi_j)(1-u)] \\ &\times I_0[\frac{1}{2}(\phi_i+\phi_j)(1+u)]\, u\, exp\{\phi_k u\}\, du\end{aligned} \tag{A.12}$$

for $(i, j, k) \in (1, 2, 3), (2, 3, 1), (3, 1, 2)$. Finally, the second derivatives of $C(D_\phi)$ are given by

$$\begin{aligned}\frac{\partial^2 c(D_\phi)}{\partial^2 \phi_i} &= \int_{-1}^{1} \frac{1}{2}I_0[\frac{1}{2}(\phi_j-\phi_k)(1-u)] \\ &\times I_0[\frac{1}{2}(\phi_j+\phi_k)(1+u)]\, u^2\, exp\{\phi_i u\}\, du.\end{aligned} \tag{A.13}$$

and

$$\begin{aligned}\frac{\partial^2 c(D_\phi)}{\partial \phi_i \partial \phi_j} &= \int_{-1}^{1} \frac{1}{4}I_1[\frac{1}{2}(\phi_j-\phi_k)(1-u)] \\ &\times I_0[\frac{1}{2}(\phi_j+\phi_k)(1+u)]\, u(1-u)\, exp\{\phi_i u\} \\ &+ \frac{1}{4}I_0[\frac{1}{2}(\phi_j-\phi_k)(1-u)] \\ &\times I_1[\frac{1}{2}(\phi_j+\phi_k)(1+u)] \\ &\times u(1+u)\, exp\{\phi_i u\}\, du\end{aligned} \tag{A.14}$$

for $i, j \in (1, 2, 3)$.

In order to find D_ϕ, define three functions from Equation (A.8).

$$a(\phi_1, \phi_2, \phi_3) = \frac{1}{c(D_\phi)} \frac{\partial c(D_\phi)}{\partial \phi_1} - g_1, \tag{A.15}$$

$$b(\phi_1, \phi_2, \phi_3) = \frac{1}{c(D_\phi)} \frac{\partial c(D_\phi)}{\partial \phi_2} - g_2 \tag{A.16}$$

and

$$c(\phi_1, \phi_2, \phi_3) = \frac{1}{c(D_\phi)} \frac{\partial c(D_\phi)}{\partial \phi_3} - g_3. \tag{A.17}$$

Define a vector function $G(D_\phi)$, so that

$$G(D_\phi) = a(\phi_1, \phi_2, \phi_3)\hat{\phi_1} + b(\phi_1, \phi_2, \phi_3)\hat{\phi_2} + c(\phi_1, \phi_2, \phi_3)\hat{\phi_3}, \tag{A.18}$$

with a, b and c as defined above. Now, solving Equation (A.8) is equivalent to finding the values (ϕ_1, ϕ_2, ϕ_3) so that $G(D_\phi) = [0, 0, 0]$. This is a problem that can be solved numerically, making use of the ordinary newtons method [80]

$$G'(x^k)\Delta x^k = -G(x^k), \qquad x^{k+1} = x^k + \Delta x^k, \tag{A.19}$$

where $F(x^k)$ is a vector function, $F'(x^k)$ is the Jacobi matrix and x^k is a vector of parameters. The Jacobi matrix for $G(D_\phi)$ is [21]

$$G'(D_\phi) = \begin{pmatrix} \frac{\partial a}{\partial \phi_1} & \frac{\partial a}{\partial \phi_2} & \frac{\partial a}{\partial \phi_3} \\ \frac{\partial b}{\partial \phi_1} & \frac{\partial b}{\partial \phi_2} & \frac{\partial b}{\partial \phi_3} \\ \frac{\partial c}{\partial \phi_1} & \frac{\partial c}{\partial \phi_2} & \frac{\partial c}{\partial \phi_3} \end{pmatrix} \tag{A.20}$$

with

$$
\begin{aligned}
\frac{\partial a}{\partial \phi_1} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_1^2} - \left(\frac{1}{c(D_\phi)}\frac{\partial c(D_\phi)}{\partial \phi_1}\right)^2 \\
\frac{\partial b}{\partial \phi_2} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_2^2} - \left(\frac{1}{c(D_\phi)}\frac{\partial c(D_\phi)}{\partial \phi_2}\right)^2 \\
\frac{\partial c}{\partial \phi_3} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_3^2} - \left(\frac{1}{c(D_\phi)}\frac{\partial c(D_\phi)}{\partial \phi_3}\right)^2 \\
\frac{\partial a}{\partial \phi_2} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_2 \partial \phi_1} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_1}\frac{\partial c(D_\phi)}{\partial \phi_2} \\
\frac{\partial a}{\partial \phi_3} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_3 \partial \phi_1} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_1}\frac{\partial c(D_\phi)}{\partial \phi_3} \\
\frac{\partial b}{\partial \phi_1} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_1 \partial \phi_2} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_1}\frac{\partial c(D_\phi)}{\partial \phi_2} \\
\frac{\partial b}{\partial \phi_3} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_3 \partial \phi_2} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_2}\frac{\partial c(D_\phi)}{\partial \phi_3} \\
\frac{\partial c}{\partial \phi_1} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_1 \partial \phi_3} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_1}\frac{\partial c(D_\phi)}{\partial \phi_3} \\
\frac{\partial c}{\partial \phi_2} &= \frac{1}{c(D_\phi)}\frac{\partial^2 c(D_\phi)}{\partial \phi_2 \partial \phi_3} - \frac{1}{c(D_\phi)^2}\frac{\partial c(D_\phi)}{\partial \phi_2}\frac{\partial c(D_\phi)}{\partial \phi_3}.
\end{aligned}
\tag{A.21}
$$

Initially the values of D_ϕ can be set to D_g. A numerical estimation is obtained using Equation (A.18), with $G'(D_\phi)$ as defined in Equation (A.20), with entries found from Equation (A.21) together with eqs. (A.9) to (A.14). In order to solve the integrals in eqs. (A.9) to (A.14), numerical integration is applied.

Appendix B

Code description

B.1 Preprocessing

The most important part of the Preprocessing step is to make sure that the direct beam is centered for every diffraction pattern from the SPED scan. Other procedures that might be considered are affine transforming the image, correcting for stretching effects in the camera setup, and noise reduction in which random noise is removed.

The process of centering the direct beam can be performed by an interpolation procedure. For every diffraction pattern two arrays are created, one for the x-axis and one for the y-axis. All intensity for a line in the diffraction pattern corresponding to a given value for the x- or y- coordinate is summed up, and stored in the corresponding position in the x or y array. This procedure results in two sets of discrete data points. By interpolating the data sets and independently finding the coordinate corresponding to the highest intensity, the (x,y) coordinate of the center is approximated, and the diffraction pattern is shifted to ensure that this point is in its center. By providing additional information, such as a window in which we are sure that the center beam is found, the accuracy and computation time of the procedure can be improved.

B.2 Simulating diffraction patterns

B.2.1 Finding a proper set of orientations

In order to obtain a good library of diffraction pattern templates for a sample for which we have no prior knowledge of possible orientations, we need a list of Euler Angles that span the set of possible orientations. For a given resolution the library should have the property that the distance between two neighbouring orientations never exceeds the stated resolution. The library should also have the property that rotations that are symmetrically equivalent only appear once, meaning that orientations that can be translated into each other by the symmetry operations of the structure appear only once. In finding the list of orientations, we will be following the conventions described in Rowenhorsts article from 2015[43].

Euler angles in the Bunge convention are denoted (ϕ_1, θ, ϕ_2). Following Rowenhosts article the starting point of obtaining the orientation list should be to generate three lists of numbers, one for each of the three Euler angles. For ϕ_1 and ϕ_2 the list goes from 0 to 360 (degrees) and for θ the list goes from 0 to 180, with a step-size determined by the stated resolution. A list of triplets is generated by systematically combining one number from each list in every way. A simple procedure isolates a single realization of the euler angles subject to *gimbal lock*[81], by demanding that when $\theta = 0$, we also have $\phi_2 = 0$.

Rotations can equivalently be described by the axis-angle convention [43]. In this convention the angle ω is limited to a maximum degree for structures of a given point group [82]. By transforming the list of euler angles to the axis-angle convention and limiting the accepted angles to the ones where $\omega < \omega_{max}$ we exclude a portion of the rotations. A final filter to accept a rotation is found in the Rodrigues-Frank representation[43]. All rotations that lie outside the asymetric domain in Rodrigues-Frank space can be removed [78]. This is done by performing a filtering routine in Rodrigues-Frank space, where every orientation in the orientation list is projected to the Rodrigues-Frank space, and the orientations within the asymetric domain are kept. The remaining rotations are transformed back to the Euler representation, and constitute a minimum set spanning the Euler space for point group.

B.2.2 Simulating data from a given orientation

In order to correctly determine an electron diffraction pattern from a crystal structure with a given orientation we rely on information on a number of quantities. As seen in Equation (2.17), scattering depends on the reciprocal lattice vector, which again depends on the lattice parameters as seen in Section 2.1.4. Furthermore, the specifics of the unit cell provides additional scattering criteria, as seen in Equations (2.31) and (2.34). Another important parameter is the set of atomic form factors, expressed in Equation (2.25), which, together with the specifics of the unit cell, tells us how strong the intensity of the scattering will be. In Equation (2.39), and in Figure 2.5 the effects of finite size effects are seen for a thin foil. Although a sample in general is not a thin foil, we will approximate it to be so, and consider the one dimensional relrod length l_{rel} to be a parameter for our simulation. Figure 2.4 illustrates how the diffraction pattern also depends on the length of the wave vectors, and thereby the electron energy or acceleration voltage. Finally, we have two parameters corresponding to the zoom or *camera length* in a SPED scan, telling how far off the direct beam we should look for reflections and how much the relative distances in the resulting diffraction pattern should be scaled. We will call these parameters the *reciprocal radius* and the *calibration*.

For a given structure, represented by a set of atoms, their positions in the unit cell and the lattice parameters, the first step in simulating the diffraction pattern is rotating the structure according to a given rotation. This can be done by expressing the orientation in terms of a rotation matrix relative to an external reference frame, and letting the matrix operate on the lattice vectors, expressed by the direct matrix tensor g_{ij} in Equation (2.5). The atoms of the unit cell will have the same relative coordinates in the rotated unit cell. By expressing these coordinates in terms of the original lattice coordinates, we find the

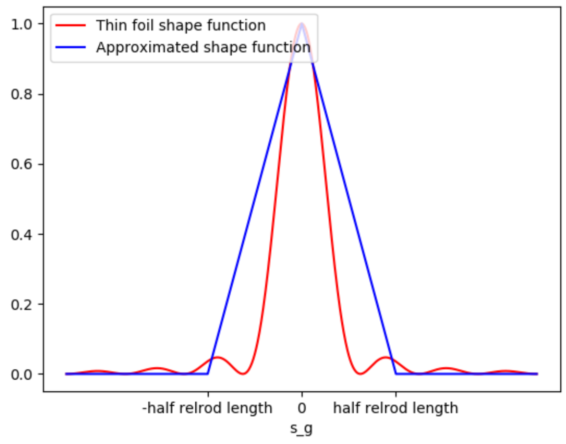


Figure B.1: The shape function of a thin foil and the function used to approximate the shape function. Noteably there is a diffrence between the two, and certain points that are extinct in the shape function have a non-zero value in the approximated function.

rotated structure coordinates. From the rotated lattice, the reciprocal lattice is found by performing a matrix inversion.

Potential reflections to be considered are points on the reciprocal lattice that fall within a sphere centered at the origin, of radius given by the reciprocal radius parameter. By definition, the Ewald sphere intersects the origin of reciprocal space. The radius of the Ewald sphere is calculated from the reported acceleration voltage applied to incoming electrons, using the relativistic formula for electron wavelength. The distance from a point that has fallen within the reciprocal radius to the Ewald sphere, the *excitation error*, is found by considering the distance along the z-axis from the point to the Ewald sphere. If the excitation error is shorter than the relrod length, the intensity of the reflection is calculated. The atomic form factor is then estimated by evaluating Equation (2.25) as described in Lobato and van Dycks article from 2014[83]. When the atomic form factor is known, the positions of the atoms in the rotated unit cell are translated to the corresponding reciprocal lattice sites, and the Fourier transform of the unit cell potential $\mathcal{F}[V_{Cell}(\mathbf{r})]$ is calculated using Equation (2.31). By Equation (2.44) the intensity at a given point is given by the Fourier transform of the unit cell potential multiplied by its complex conjugate times the shape function of a thin foil. We will approximate this thin foil shape function with a linear function, symmetric around zero, as seen in Figure B.1. The coordinates of points with non-zero intensities, along with the corresponding intensities are stored and used as the simulated diffraction pattern for the given orientation. The final step is to scale the relative distances to fit the requested calibration. This is done by scaling the distances in reciprocal space, and keeping points that stay within the field of view.

The described process is implemented in the method ”get_diffraction_library()” in the DiffractionLibraryGenerator class in pyXem.

Appendix C

Computer code

C.1 Python code for analyzing random orientations

Analyze data from a list of random orientations together with corresponding template matching results.

Analyze a template matching result with random orientations

In [ ]:

```
#Standard Libraries and pyxem + hyperspy
%matplotlib qt
import numpy as np
import math
import hyperspy.api as hs
import pyxem as pxm
import diffpy.structure
from matplotlib import pyplot as plt

#Orix imports
from orix.quaternion.orientation import Orientation, Misorientation, Rotation
from orix.quaternion.symmetry import O,C1,D6h,Oh,D2h
from orix.quaternion.orientation_region import OrientationRegion
from orix.quaternion.rotation import von_mises
from orix import plot
from orix.vector.neo_euler import AxAngle

#Diffsims imports
from diffsims.generators.diffraction_generator import DiffractionGenerator
from diffsims.libraries.structure_library import StructureLibrary
from diffsims.generators.library_generator import DiffractionLibraryGenerator
import diffsims.generators.rotation_list_generators as rotation_list_generators

#Pyxem imports
from pyxem.utils.sim_utils import sim_as_signal
from pyxem.generators.indexation_generator import IndexationGenerator
```

In [ ]:

```
import matplotlib.pylab as pylab
params = {'legend.fontsize': 'xx-large',
          'figure.figsize': (7, 6),
         'axes.labelsize': 'xx-large',
         'axes.titlesize':'xx-large',
         'xtick.labelsize':'xx-large',
         'ytick.labelsize':'xx-large'}
pylab.rcParams.update(params)
```

In [ ]:

```python
from pyxem.signals.indexation_results import TemplateMatchingResults
import numpy as np
def load_template_matching_results(filename):
    """
    ---
    Reads a .npy file and returns a TemplateMatchingResults object.

    Parameters
    ----------
    filename : str
        The name of the .npy file to be read

    Returns
    -------
    match_results : TemplateMatchingResults
        A TemplateMatchingResults object containing the loaded data
    ---
    """

    try:
        loaded_numpy_array = np.load(filename)
    except ValueError:
        raise ValueError('Could not open file: {}'.format(filename))

    dimensions = loaded_numpy_array.shape
    if len(dimensions) == 3:
        match_results_data = np.zeros((dimensions[0], dimensions[1], dimensions[2] - 2), dtype = object)
        for i in range(dimensions[0]):
            for j in range(dimensions[1]):
                match_results_data[i, j, 0] = int(loaded_numpy_array[i, j, 0])
                match_results_data[i, j, -1] = loaded_numpy_array[i, j, -1]
                my_array = np.asarray([loaded_numpy_array[i, j, 1],
                                       loaded_numpy_array[i, j, 2],
                                       loaded_numpy_array[i, j, 3]])
                match_results_data[i, j, 1] = my_array

        return TemplateMatchingResults(match_results_data)

    elif len(dimensions) == 4:
        match_results_data = np.zeros((dimensions[0], dimensions[1], dimensions[2], dimensions[3] - 2), dtype = o
bject)
        for i in range(dimensions[0]):
            for j in range(dimensions[1]):
                for k in range(dimensions[2]):
                    match_results_data[i, j, k, 0] = int(loaded_numpy_array[i, j, k, 0])
                    match_results_data[i, j, k, -1] = loaded_numpy_array[i, j, k, -1]
                    my_array = np.asarray([loaded_numpy_array[i, j, k, 1],
                                           loaded_numpy_array[i, j, k, 2],
                                           loaded_numpy_array[i, j, k, 3]])
                    match_results_data[i, j, k, 1] = my_array

        return TemplateMatchingResults(match_results_data)

    else:
        raise ValueError('Could not load file of dimensions {}'.format(dimensions))
```

Load template matching results and corresponding true orientations

In [ ]:

```python
match_results = load_template_matching_results("../Results/RandomOri/IdunDeg1sigma0.03GaSbRandomOrientations_Fast
.npy")
```

In [ ]:

```python
true_orientations = np.load("../Results/RandomOri/random_orientation_list_fast.npy")
```

Reliability Index

$R = \left(1 - \frac{\gamma_2}{\gamma_1}\right)$

In [ ]:

```
correlation_scores = match_results.data[:, :2, 2]
N = correlation_scores.shape[0]
reliability_indices = np.zeros((N))
for i in range (N):
    reliability_indices[i] = 1 - correlation_scores[i, 1] / correlation_scores[i, 0]
```

In [ ]:

```
np.save("../Results/RandomOriMord/RI_FF_Mord", reliability_indices)
```

Weighted Average Reliability Index

$R_w = \frac{1}{N-1} \sum_{i=2}^{N} \left(1 - \frac{\gamma_i}{\gamma_1} \frac{\theta_{i,1}}{\theta_{max}}\right)$

In [ ]:

```
N_best = 20
best_orientations = match_results.data[:, :N_best, 1]
correlation_scores = match_results.data[:, :N_best, 2]
symmetry = O
theta_max = 120 #Is known from crystal symmetry.
#See https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2017.0274 Fig 5, and find omega_max. (Here theta)
```

In [ ]:

```
def find_distance_rad (orientation1, orientation2, symmetry):
    """
    Takes two input Euler orientations and the symmetry of the system, and computes the distance between them
    in radians.
    """
    orientations = list([orientation1, orientation2])
    ori = Orientation.from_euler(np.deg2rad(orientations))
    misori_base = Misorientation(ori).set_symmetry(symmetry,symmetry)
    distlist = misori_base.distance()

    return distlist[0,1]
```

In [ ]:

```
N = correlation_scores.shape[0]
weighted_average_reliability_index = np.zeros(N)

for i in range (N):
    if i%10 == 0:
        print(i)
    score = 0
    for j in range (1, N_best):
        score += (1 - (correlation_scores[i, j] / correlation_scores[i, 0])
                  * (find_distance_rad(best_orientations[i, 0], best_orientations[i, j], D2h)
                  / np.deg2rad(theta_max)))

    weighted_average_reliability_index[i] = score / (N_best - 1)
```

In [ ]:

```
np.save("../Results/RandomOriMord/WARI_FF_Mord", weighted_average_reliability_index)
```

Distance to truth

In [ ]:

```
def set_distances_to_truth (orientation_list, truth_list, symmetry):
    """
    Finds the distance from every orientation in an orientation_list to the true orientation.
    """
    distances = np.zeros(len(orientation_list))
    for i in range(len(orientation_list)):
        if i % 100 == 0:
            print(i)
        distances[i] = find_distance(orientation_list[i], truth_list[i], symmetry)

    return distances

def find_distance (orientation1, orientation2, symmetry):
    """
    Takes two input Euler orientations and the symmetry of the system, and computes the distance between them
    in degrees.
    """
    orientations = list([orientation1, orientation2])
    ori = Orientation.from_euler(np.deg2rad(orientations))
    misori_base = Misorientation(ori).set_symmetry(symmetry,symmetry)
    distlist = misori_base.distance()

    return np.rad2deg(distlist[0,1])
```

In [ ]:

```
from orix.quaternion.symmetry import D2h
```

In [ ]:

```
predictions = match_results.data[:, 0, 1]
```

In [ ]:

```
angular_distances = set_distances_to_truth(predictions, true_orientations, D2h)
```

In [ ]:

```
np.save("../Results/RandomOriMord/angular_distances_FF_Mord", angular_distances)
```

Save angular distances and corresponding orientations to MTEX readable format

In [ ]:

```
true_orientations = np.load("../Results/RandomOriMord/random_orientation_list_fast_Mord.npy")
```

In [ ]:

```
angular_distances = np.load('../Results/RandomOriMord/angular_distances_Fast_Mord.npy')
```

In [ ]:

```
N = len(angular_distances)
ret_arr_AD = np.zeros((N,7))
for i in range (N):
    ret_arr_AD[i,:] = [0, true_orientations[i][0],
                true_orientations[i][1], true_orientations[i][2], angular_distances[i],i,i]
```

In [ ]:

```
np.savetxt("../Results/RandomOriMord/to_MTEX_angular_distances_FF_Mord.csv", ret_arr_AD)
```

Plot simulated data from selected indices

In [ ]:

```
true_orientations = np.load("../Results/RandomOri/random_orientation_list_FullFrame.npy")
```

In [ ]:

```
match_results = load_template_matching_results("../Results/RandomOri/IdunDeg3sigma0.03GaSbRandomOrientations_Full
Frame.npy")
```

Reconstruct the templates that matched the simulated data

Extract all Euler angles for matching templates

In [ ]:

```
angle_list = [ ]
shape = match_results.data.shape
best_orientations = match_results.data[:,:,1]
for i in range(shape[0]):
    for j in range(shape[1]):
        if not any((best_orientations[i, j] == x).all() for x in angle_list):
            angle_list.append(best_orientations[i,j])
```

In [ ]:

```
N = len(angle_list)
ret_arr_AD = np.zeros((N,7))
for i in range (N):
    ret_arr_AD[i,:] = [0, angle_list[i][0],
                       angle_list[i][1], angle_list[i][2], 1,i,i]
```

Set simulation parameters, must be the same as the ones used when computing the template matching results

In [ ]:

```
#Parameters, known from about file
phase_name = ['GaSb']
beam_energy = 200
ediff = DiffractionGenerator(beam_energy, 0.025)
pattern_size = 256
half_pattern_size = pattern_size // 2
reciprocal_radius = 2.0
calibration = 1.0 * reciprocal_radius / half_pattern_size
sigma = 0.03
```

In [ ]:

```
structure_GaSb = diffpy.structure.loadStructure("./Cif/GaSb.cif")
```

In [ ]:

```
orientation_list = true_orientations
```

Simulate diffraction patterns

In [ ]:

```
sample_lib = StructureLibrary(phase_name, [structure_GaSb], [orientations_list])

diff_gen = DiffractionLibraryGenerator(ediff)
library = diff_gen.get_diffraction_library(sample_lib,
                                           calibration=calibration,
                                           reciprocal_radius=reciprocal_radius,
                                           half_shape=(half_pattern_size, half_pattern_size),
                                           with_direct_beam=False)

data_GaSb = []

for angle in orientations_list:
    pattern = sim_as_signal(library.get_library_entry(phase=phase_name[0], angle=angle)['Sim'],
                            pattern_size,sigma = sigma,max_r = reciprocal_radius)

    data_GaSb.append(pattern)

data = [x.data for x in data_GaSb]

test_data = pxm.ElectronDiffraction2D(np.asarray(data).reshape(len(orientations_list), pattern_size, pattern_size
))
test_data.set_diffraction_calibration(calibration)
```

Simulate templates from the list of Euler Angles of the highest scoring templates

In [ ]:

```
structure_library = StructureLibrary.from_orientation_lists(['GaSb'], [structure_GaSb], [angle_list])
diff_gen = DiffractionLibraryGenerator(ediff)
template_library = diff_gen.get_diffraction_library(structure_library,
                                                    calibration=calibration,
                                                    reciprocal_radius=reciprocal_radius,
                                                    half_shape=(half_pattern_size, half_pattern_size),
                                                    with_direct_beam=False)
```

Define functionality for plotting templates together with signals

In [ ]:

```
class IndexTracker(object):
    def __init__(self, ax1, ax2, signal, storage, kwargs_for_signal = {}, kwargs_for_template_scatter = {}):

        self.ax1 = ax1
        self.ax2 = ax2

        self.signal = signal

        self.slices = signal.shape[0]
        self.ind = self.slices // 2
        self.rank = 0
        self.max_rank = len(storage[0])
        self.storage = storage

        phase = self.storage[self.ind][self.rank][0]
        angle = self.storage[self.ind][self.rank][1]
        correlation_score = self.storage[self.ind][self.rank][2]
        coordinates = self.storage[self.ind][self.rank][3]
        intensities = self.storage[self.ind][self.rank][4]
        x,y = zip(*coordinates)

        self.im = ax1.imshow(self.signal[self.ind], **kwargs_for_signal)
        self.im2 = ax2.imshow(self.signal[self.ind], **kwargs_for_signal)
        self.line1 = ax2.scatter(x = x, y = y, c = intensities, **kwargs_for_template_scatter)
        self.cbar = plt.colorbar(self.line1)
        self.cbar2 = plt.colorbar(self.im)
        self.cbar.set_label("Template", size = 15)
        self.cbar2.set_label("Signal", size = 15)
        self.cbar.ax.tick_params(labelsize=15)
        #cbar.ax.set_autoscale_on(True)
        self.cbar2.ax.tick_params(labelsize=15)
        self.update()

    def onscroll(self, event):
        if event.button == 'up':
            self.ind = (self.ind + 1) % self.slices
        else:
            self.ind = (self.ind - 1) % self.slices
        self.update()

    def click(self, event):
        self.rank = (self.rank + 1) % self.max_rank
        self.update()

    def update(self):

        self.im.set_data(self.signal[self.ind,:,:])
        self.ax1.set_ylabel('slice %s' % (self.ind + 1))
        self.ax1.set_title('Signal')
        self.im.axes.figure.canvas.draw()

        phase = self.storage[self.ind][self.rank][0]
        angle = self.storage[self.ind][self.rank][1]
        correlation_score = self.storage[self.ind][self.rank][2]
        coordinates = self.storage[self.ind][self.rank][3]
        intensities = self.storage[self.ind][self.rank][4]
        x,y = zip(*coordinates)

        self.im2.set_data(self.signal[self.ind,:,:])
        self.line1.set_offsets(coordinates)
        self.line1.set_array(intensities)
        self.ax2.set_title('Euler Angle: ({0:.0f},{1:.0f},{2:.0f}) \n Rank: {3} Correlation Score: {4:.2f}'.forma
t(angle[0],
                                                                                     angle[1],
                                                                                     angle[2],
                                                                                     self.rank,
                                                                                     correlation_score), fontsize
= 20)
```

```
        self.line1.set_clim(vmin=intensities.min(),vmax=intensities.max())
        self.cbar.update_normal(self.line1)
        self.cbar.draw_all()
        self.im2.axes.figure.canvas.draw()


from pyxem.utils.indexation_utils import get_nth_best_solution

def peaks_from_best_n_templates(tm_results, library, n = None):
    number_of_templates = tm_results.data.shape[0]
    if n == None:
        n = tm_results.data.shape[1]

    storage = []
    for i in range(number_of_templates):
        peaks_and_intensities = []

        for j in range(n):
            best_fit = get_nth_best_solution(tm_results.data[i], mode = 'template', rank=j)
            phase_names = list(library.keys())
            phase_index = int(best_fit[0])
            phase = phase_names[phase_index]
            simulation = library.get_library_entry(
                phase=phase,
                angle=tuple(best_fit[1]))
            peaks = simulation['pixel_coords']
            intensities = simulation['intensities']
            peaks_and_intensities.append([phase, best_fit[1], best_fit[2], peaks, intensities])
        storage.append(peaks_and_intensities)
    return storage

def plot_results_on_signal(tm_results, signal, library, *args, **kwargs):
    storage = peaks_from_best_n_templates(tm_results, library)

    fig, [ax1,ax2] = plt.subplots(1, 2, gridspec_kw={'width_ratios': [5, 8]})
    plt.axis('off')


    tracker = IndexTracker(ax1, ax2, signal, storage, kwargs_for_signal = {"cmap" : "inferno_r", "vmax" : 0.1, "v
min": 0}, **kwargs)


    fig.canvas.mpl_connect('scroll_event', tracker.onscroll)
    fig.canvas.mpl_connect('button_press_event', tracker.click)
    plt.show()
    return tracker
```

Plot signal and templates together

Scroll to go to next index, click to go to next rank (the next best fitting template)

In [ ]:

```
plot_results_on_signal(match_results, test_data.data, template_library, kwargs_for_template_scatter = {"cmap" : "
viridis", "s": 50, "marker" : "x"})
```

Compute number of correctly indexed orientations and STD of correctly indexed orientations

In [ ]:

```
distance_from_truth = angular_distances
```

In [ ]:

```
len(distance_from_truth)
```

In [ ]:

```
distance_from_truth = np.load('../Results/RandomOriMord/angular_distances_ZNCC_Mord.npy')
```

In [ ]:

```
correct_results = [x for x in distance_from_truth if x < 3]
```

In [ ]:

```
len(correct_results)
```

In [ ]:

```
len(correct_results) / len(distance_from_truth)
```

In [ ]:

```
correct_results = np.asarray(correct_results)
std = np.sqrt( sum(correct_results * correct_results) / len(correct_results) )
```

In [ ]:

```
std
```

Finding number of rotations within the correct rotation range

In [ ]:

```
import time
import random
```

In [ ]:

```
random.seed(time.time())
```

In [ ]:

```
random.randint(0,max_index)
```

In [ ]:

```
true_orientations = np.load("../Results/RandomOriMord/random_orientation_list_fast_Mord.npy")
```

In [ ]:

```
match_results = load_template_matching_results("../Results/RandomOriMord/IdunDeg1sigma0.008GaSbRandomOrientations
_Fast_Mord.npy")
```

In [ ]:

```
number_of_estimates = 1000
symmetry = 0
max_index = len(true_orientations) - 1
```

In [ ]:

```
distance_error = np.zeros((number_of_estimates))
for i in range(number_of_estimates):
    if i%100 == 0:
        print(i)
    index2 = random.randint(0,max_index)
    index1 = random.randint(0,max_index)
    while index1 == index2:
        index1 = random.randint(0,max_index)
    true_distance = find_distance(true_orientations[index1], true_orientations[index2], symmetry)
    estimated_distance = find_distance(match_results.data[index1,0,1], match_results.data[index2,0,1], symmetry)
    distance_error[i] = np.abs(true_distance - estimated_distance)
```

In [ ]:

```
correct_distances = [dist for dist in distance_error if dist < 1.8]
```

In [ ]:

```
len(correct_distances)
```

Compute correlation coefficient between reliability metrics and distance from truth

In [ ]:

```
I = np.sum(test_data.data, axis = (1,2))
```

In [ ]:

```
distance_from_truth = np.load('../Results/RandomOriMord/angular_distances_ZNCC_Mord.npy')
```

In [ ]:

```
distance_from_truth[50]
```

In [ ]:

```
WARI = np.load('../Results/RandomOriMord/WARI_ZNCC_Mord.npy')
```

In [ ]:

```
RI = np.load('../Results/RandomOriMord/RI_ZNCC_Mord.npy')
```

In [ ]:

```
max_angle = 120
```

Normalize angular distances

In [ ]:

```
distance_from_truth = distance_from_truth / max_angle
```

In [ ]:

```
avg_dist = np.average(distance_from_truth)
std_dist = np.std(distance_from_truth)
standard_distance = (distance_from_truth - avg_dist) / std_dist
```

In [ ]:

```
avg_I = np.average(I)
std_I = np.std(I)
standard_I = (I - avg_I) / std_I
```

In [ ]:

```
avg_WARI = np.average(WARI)
std_WARI = np.std(WARI)
standard_WARI = (WARI - avg_WARI) / std_WARI
```

In [ ]:

```
avg_RI = np.average(RI)
std_RI = np.std(RI)
standard_RI = (RI - avg_RI) / std_RI
```

In [ ]:

```
corr_WARI = np.sum(standard_distance * standard_WARI) / (len(WARI) - 1)
```

In [ ]:

```
corr_RI = np.sum(standard_distance * standard_RI) / (len(RI) - 1)
```

In [ ]:

```
corr_I = np.sum(standard_distance * standard_I) / (len(I) - 1)
```

In [ ]:

```
corr_I
```

In [ ]:

```
corr_RI
```

In [ ]:

```
corr_WARI
```

In [ ]:

```
plt.figure()
plt.plot(distance_from_truth, RI, '*')
plt.show()
```

C.2 Python code tilt series

Generate a tilt series for simulated data in Python.

Template matching on simulated tilt series data, with subsequent data analysis using the von Mises - Fisher distribution

In this notebook a tilt series is specified by defining a tilt axis, a rotation angle and the number of tilts. For every orientation in the tilt series, a diffraction pattern is simulated. Thereafter, a template library is generated and used to identify the orientations in the tilt series through template matching.

Import libraries

In [ ]:

```
#Standard Libraries and pyxem + hyperspy
%matplotlib qt
import numpy as np
import math
import hyperspy.api as hs
import pyxem as pxm
import diffpy.structure
from matplotlib import pyplot as plt

#Orix imports
from orix.quaternion.orientation import Orientation, Misorientation, Rotation
from orix.quaternion.symmetry import O,C1,D6h,Oh
from orix.quaternion.orientation_region import OrientationRegion
from orix.quaternion.rotation import von_mises
from orix import plot

#Diffsims imports
from diffsims.generators.diffraction_generator import DiffractionGenerator
from diffsims.libraries.structure_library import StructureLibrary
from diffsims.generators.library_generator import DiffractionLibraryGenerator
import diffsims.generators.rotation_list_generators as rotation_list_generators

#Pyxem imports
from pyxem.utils.sim_utils import sim_as_signal
from pyxem.generators.indexation_generator import IndexationGenerator
```

Create a list of orientations from a tilt series.

In [ ]:

```
from orix.vector.neo_euler import AxAngle
from orix.quaternion.symmetry import O
```

Set tilt series parameters

In [ ]:

```
rotation_axis = [1, 1, 1] #Axis on which the rotation is performed
rotation_angle = 4 #Degrees
initial_orientation_euler = [0., 0., 0.] #Euler angles
nb_tilts = 30 #Number of tilts in the tilt series.
```

Define a constant rotation and the initial orientation

In [ ]:

```
axis_angle = AxAngle.from_axes_angles(rotation_axis, np.radians(rotation_angle))
constant_rotation = Rotation.from_neo_euler(axis_angle)
initial_orientation = Orientation.from_euler(np.radians(initial_orientation_euler))
initial_orientation = initial_orientation.set_symmetry(O)
```

Generate tilt series

In [ ]:

```
tilt_series = [initial_orientation_euler]
current_orientation = initial_orientation

for _ in range (nb_tilts - 1):
    current_orientation = constant_rotation * current_orientation
    current_euler = current_orientation.to_euler()
    tilt_series.append(*np.rad2deg(current_euler))
```

Simulate data

Set simulation parameters

In [ ]:

```
phase_name = ['GaSb']
beam_energy = 200
max_exitation_error = 0.025
ediff = DiffractionGenerator(beam_energy, max_exitation_error)
pattern_size = 256
half_pattern_size = pattern_size // 2
reciprocal_radius = 2.0
calibration = 1.0 * reciprocal_radius / half_pattern_size
sigma = 0.03
```

Import structure from .cif file

In [ ]:

```
structure_GaSb = diffpy.structure.loadStructure("Cif/GaSb.cif")
```

Simulate diffraction data

In [ ]:

```
sample_lib = StructureLibrary(phase_name, [structure_GaSb], [tilt_series])

diff_gen = DiffractionLibraryGenerator(ediff)
library = diff_gen.get_diffraction_library(sample_lib,
                                           calibration=calibration,
                                           reciprocal_radius=reciprocal_radius,
                                           half_shape=(half_pattern_size, half_pattern_size),
                                           with_direct_beam=False)

data_GaSb = []

for angle in tilt_series:
    pattern = sim_as_signal(library.get_library_entry(phase=phase_name[0], angle=angle)['Sim'],
                            pattern_size,sigma = sigma,max_r = reciprocal_radius)

    data_GaSb.append(pattern)

data = [x.data for x in data_GaSb]

test_data = pxm.ElectronDiffraction2D(np.asarray(data).reshape(len(tilt_series), pattern_size, pattern_size))
test_data.set_diffraction_calibration(calibration)
```

Generate a template library

Create template library

In [ ]:

```
structure_library = StructureLibrary.from_orientation_lists(['GaSb'],[structure_GaSb],[tilt_series])
diff_gen = DiffractionLibraryGenerator(ediff)
template_library = diff_gen.get_diffraction_library(structure_library,
                                                    calibration=calibration,
                                                    reciprocal_radius=reciprocal_radius,
                                                    half_shape=(half_pattern_size, half_pattern_size),
                                                    with_direct_beam=False)
```

Run template matching

In [ ]:

```
indexer = IndexationGenerator(test_data, template_library)
match_results = indexer.correlate(n_largest=5, method = "full_frame_correlation")#, method = "zero_mean_normalize
d_correlation"
```

Computation of Reliability metrics

Reliability Index

$R = \left(1 - \frac{\gamma_2}{\gamma_1}\right)$

In [ ]:

```
correlation_scores = match_results.data[:, :2, 2]
```

For ZNCC, as correlation score is in range (-1,1)

In [ ]:

```
correlation_scores = (correlation_scores + 1) / 2
```

In [ ]:

```
N = correlation_scores.shape[0]
reliability_indices = np.zeros((N))
for i in range (N):
    reliability_indices[i] = 1 - correlation_scores[i, 1] / correlation_scores[i, 0]
```

In [ ]:

```
plt.figure()
xValues = np.array(range(len(reliability_indices)))
plt.plot(xValues + 1,reliability_indices, 'x')
plt.ylim(0,1)
plt.title("Reliability Index", fontsize = 18)
plt.xlabel("Index", fontsize = 15)
plt.ylabel("", fontsize = 30)
plt.show()
```

Near-Match Similarity Index

$\eta_i = \frac{1}{2}(\#(S_{i-1} \cap S_i) + \#(S_i \cap S_{i+1}))$

In [ ]:

```
N_best = 5 #Number of best matches to be considered
best_orientations = match_results.data[:, :N_best, 1]
```

In [ ]:

```
N = correlation_scores.shape[0]
near_match_similarity_index = np.zeros(N)

#General
for i in range (1, N-1):
    overlap = 0
    for j in range (N_best):
        for k in range (N_best):
            if all(best_orientations[i, j] == best_orientations[i-1, k]): #Replace with np.approx_equal?
                overlap += 1
            if all(best_orientations[i, j] == best_orientations[i+1, k]):
                overlap += 1
    near_match_similarity_index[i] = overlap/2

#Edges
overlap_start = 0
overlap_end = 0
for j in range (N_best):
    for k in range (N_best):
        if all (best_orientations[0, j] == best_orientations[1, k]):
            overlap_start += 1
        if all (best_orientations[-1, j] == best_orientations[-2, k]):
            overlap_end += 1
near_match_similarity_index[0] = overlap_start
near_match_similarity_index[-1] = overlap_end

near_match_similarity_index
```

In [ ]:

```
plt.figure()
plt.plot(near_match_similarity_index)
plt.ylim(0,N_best)
plt.show()
```

Weighted Average Reliability Index

$$R_w = \frac{1}{N-1} \sum_{i=2}^{N} \left(1 - \frac{\gamma_i}{\gamma_1} \frac{1-\cos\theta_{i,1}}{1-\cos\theta_{max}}\right)$$

In [ ]:

```
N_best = 5
best_orientations = match_results.data[:, :N_best, 1]
correlation_scores = match_results.data[:, :N_best, 2]
symmetry = O
theta_max = 61.86 #Is known from crystal symmetry.
#See https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2017.0274 Fig 5, and find omega_max. (Here theta)
```

In [ ]:

```
def find_distance_rad (orientation1, orientation2, symmetry):
    """
    Takes two input Euler orientations and the symmetry of the system, and computes the distance between them
    in radians.
    """
    orientations = list([orientation1, orientation2])
    ori = Orientation.from_euler(np.deg2rad(orientations))
    misori_base = Misorientation(ori).set_symmetry(symmetry,symmetry)
    distlist = misori_base.distance()

    return distlist[0,1]
```

In [ ]:

```
N = correlation_scores.shape[0]
weighted_average_reliability_index = np.zeros(N)

for i in range (N):
    score = 0
    for j in range (1, N_best):
        score += (1 - (correlation_scores[i, j] / correlation_scores[i, 0])
                  * ((1 - np.cos(find_distance_rad(best_orientations[i, 0], best_orientations[i, j], symmetry)))
                  / (1 - np.cos(np.deg2rad(theta_max)))))

    weighted_average_reliability_index[i] = score / (N_best - 1)

weighted_average_reliability_index
```

In [ ]:

```
plt.figure()
plt.plot(weighted_average_reliability_index)
plt.ylim(0,1)
plt.show()
```

Plotting and data analysis

Define plotting functions

Allows plotting distance to true value and plotting distance to previous orientation in the tilt series

In [ ]:

```
def find_distance (orientation1, orientation2, symmetry):
    """
    Takes two input Euler orientations and the symmetry of the system, and computes the distance between them
    in degrees.
    """
    orientations = list([orientation1, orientation2])
    ori = Orientation.from_euler(np.deg2rad(orientations))
    misori_base = Misorientation(ori).set_symmetry(symmetry,symmetry)
    distlist = misori_base.distance()

    return np.rad2deg(distlist[0,1])

def set_distances_to_first (orientation_list,symmetry):
    """
    Finds the distance from every orientation in an orientation_list to the first orientation.
    """
    distances = np.zeros((len(orientation_list)))
    for i in range(len(orientation_list)):
        distances[i] = find_distance(orientation_list[i],orientation_list[0],symmetry)

    return distances

def set_distances_to_previous (orientation_list,symmetry):
    """
    Finds the distance from every orientation in an orientation_list to the previous orientation.
    """
    distances = np.zeros((len(orientation_list)-1))
    for i in range(1,len(orientation_list)):
        distances[i-1] = find_distance(orientation_list[i],orientation_list[i-1],symmetry)

    return distances

def set_distances_to_origin (orientation_list,symmetry):
    """
    Finds the distance from every orientation in an orientation_list to the (0,0,0) orientation.
    """
    distances = np.zeros((len(orientation_list)-1))
    for i in range(1,len(orientation_list)):
        distances[i-1] = find_distance(orientation_list[i],[0,0,0],symmetry)

    return distances

def set_distances_to_truth (orientation_list, truth_list, symmetry):
    """
    Finds the distance from every orientation in an orientation_list to the true orientation.
    """
    distances = np.zeros(len(orientation_list))
    for i in range(len(orientation_list)):
        distances[i] = find_distance(orientation_list[i], truth_list[i], symmetry)

    return distances

def linear_regression(angle_list):
    """
    Computes coefissients for a linear regression.
    """
    n = len(angle_list)
    sum_x = 0
    sum_y = 0
    sum_xy = 0
    sum_y2 = 0
    sum_x2 = 0
    for i in range(n):
        sum_x += (i)
        sum_y += angle_list[i]
        sum_xy += (i)*angle_list[i]
        sum_y2 += pow(angle_list[i],2)
        sum_x2 += pow(i,2)

    b = (sum_y*sum_x2-sum_x*sum_xy)/(n*sum_x2-pow(sum_x,2))
    a = (n*sum_xy - sum_x*sum_y)/(n*sum_x2-pow(sum_x,2))

    return a,b

def linear_regression_plot(orientation_list,symmetry,method):
    """
    Plots the linear regression along with the points used to compute the regression.
    """

    if method == 'to_first':
        angle_list = set_distances_to_first(orientation_list,symmetry)
    elif method == 'to_previous':
        angle_list = set_distances_to_previous(orientation_list,symmetry)
    elif method == 'to_origin':
        angle_list = set_distances_to_origin(orientation_list,symmetry)

    a,b = linear_regression(angle_list)
    regPlot = []
    xValues = []

    for i in range(len(angle_list)):
        regPlot.append(a*i+b)
        xValues.append(i)

    plt.figure()
    plt.plot(xValues,regPlot,'r-',xValues,angle_list,'b*')
    plt.ylim(0,max(max(regPlot),max(angle_list))+2)
    plt.show()

def linear_regression_plot_to_truth(orientation_list, truth_list, symmetry):
    """
    Plots the linear regression along with the points used to compute the regression.
    """
    angle_list = set_distances_to_truth(orientation_list, truth_list, symmetry)

    a,b = linear_regression(angle_list)
    regPlot = []
    xValues = []

    for i in range(len(angle_list)):
        regPlot.append(a*i+b)
        xValues.append(i)

    plt.figure()
    plt.plot(xValues,regPlot,'r-',xValues,angle_list,'b*')
    plt.ylim(0,max(max(regPlot),max(angle_list))+2)
    plt.show()

def constant_plot_to_truth(orientation_list, truth_list, symmetry, constant):
    angle_list = set_distances_to_truth(orientation_list, truth_list, symmetry)
    constant_list = np.ones((len(angle_list))) * constant
    xValues = np.array(range(len(angle_list)))

    plt.figure()
    plt.plot(xValues + 1, constant_list, 'r-', label = 'Expected')
    plt.plot(xValues + 1, angle_list, 'b*', label = 'Measurement')
    plt.legend(fontsize = 15, loc = 'upper right')
    plt.title("Distance to True Orientation", fontsize = 18)
    plt.xlabel("Index", fontsize = 18)
    plt.ylabel("Angular Distance " + r'$(\degree)$', fontsize = 18)
    plt.ylim(0,max(max(constant_list),max(angle_list))+2)
    plt.show()


def constant_plot_to_previous(orientation_list, symmetry, constant):
    angle_list = set_distances_to_previous(orientation_list,symmetry)
    constant_list = np.ones((len(angle_list))) * constant
    xValues = np.array(range(len(angle_list)))

    plt.figure()
    plt.plot(xValues + 2,constant_list,'r-', label = 'Expected')
    plt.plot(xValues + 2,angle_list,'b*', label = 'Measurement')
    plt.ylim(0,max(max(constant_list),max(angle_list))+2)
    plt.legend(fontsize = 15, loc = 'upper right')
    plt.title("Distance to Previous Index", fontsize = 18)
    plt.xlabel("Index", fontsize = 18)
    plt.ylabel("Angular Distance " + r'$(\degree)$', fontsize = 18)
    plt.show()
```

Set parameters for plots in matplotlib

In [ ]:

```
import matplotlib.pylab as pylab
params = {'legend.fontsize': 'xx-large',
          'figure.figsize': (7, 6),
         'axes.labelsize': 'xx-large',
         'axes.titlesize':'xx-large',
         'xtick.labelsize':'xx-large',
         'ytick.labelsize':'xx-large'}
pylab.rcParams.update(params)
```

In [ ]:

```
array = match_results.data[:,0,1]
constant_plot_to_truth(array, tilt_series, 0, 0)
constant_plot_to_previous(array, 0, 4)
```

In [ ]:

```
array = match_results.data[:,0,1]
linear_regression_plot(array, 0, 'to_previous')
linear_regression_plot_to_truth(array, tilt_series, 0)
```

To MTEX

orientations

In [ ]:

```
N = match_results.data.shape[0]
ret_arr = np.zeros((N,7))
for i in range (N):
    ret_arr[i,:] = [0, match_results.data[i,0,1][0],
                    match_results.data[i,0,1][1], match_results.data[i,0,1][2], match_results.data[i, 0, 2],i,i]
```

In [ ]:

```
np.savetxt("..//to_MTEX_orientations_correlation_score_FF.csv", ret_arr)
```

rotations

In [ ]:

```
euler_angles_from_best_matches = match_results.data[:,0,1]
euler_angles_from_best_matches = euler_angles_from_best_matches.tolist()


orientations_1 = Orientation.from_euler(np.radians(euler_angles_from_best_matches[:-1]))
orientations_2 = Orientation.from_euler(np.radians(euler_angles_from_best_matches[1:]))
orientations_1 = orientations_1
orientations_2 = orientations_2

rotation_list = Rotation(orientations_2 * ~orientations_1)
rotation_array = np.rad2deg(rotation_list.to_euler())
```

In [ ]:

```
N = len(rotation_array)
ret_arr_rot = np.zeros((N,7))
for i in range (N):
    ret_arr_rot[i,:] = [0, rotation_array[i][0],
                    rotation_array[i][1], rotation_array[i][2], 1, i, i]
```

In [ ]:

```
np.savetxt("..//to_MTEX_rotations_FF.csv", ret_arr_rot)
```

Other

In [ ]:

```
N = match_results.data.shape[0]
ret_arr = np.zeros((N,7))
for i in range (N):
    ret_arr[i,:] = [0, match_results.data[i,0,1][0],
                    match_results.data[i,0,1][1], match_results.data[i,0,1][2], match_results.data[i,0,2], i, i]
```

In [ ]:

```
np.savetxt("..//101GaSbFF.csv", ret_arr)
```

Plot result on signal functions

In [ ]:

```
class IndexTracker(object):
    def __init__(self, ax1, ax2, signal, storage, kwargs_for_signal = {}, kwargs_for_template_scatter = {}):

        self.ax1 = ax1
        self.ax2 = ax2

        self.signal = signal

        self.slices = signal.shape[0]
        self.ind = self.slices // 2
        self.rank = 0
        self.max_rank = len(storage[0])
        self.storage = storage

        phase = self.storage[self.ind][self.rank][0]
        angle = self.storage[self.ind][self.rank][1]
        correlation_score = self.storage[self.ind][self.rank][2]
        coordinates = self.storage[self.ind][self.rank][3]
        intensities = self.storage[self.ind][self.rank][4]
        x,y = zip(*coordinates)

        self.im = ax1.imshow(self.signal[self.ind], **kwargs_for_signal)
        self.im2 = ax2.imshow(self.signal[self.ind], **kwargs_for_signal)
        self.line1 = ax2.scatter(x = x, y = y, c = intensities, **kwargs_for_template_scatter)
        self.cbar = plt.colorbar(self.line1)
        self.cbar2 = plt.colorbar(self.im)
        self.cbar.set_label("Template", size = 15)
        self.cbar2.set_label("Signal", size = 15)
        self.cbar.ax.tick_params(labelsize=15)
        self.cbar2.ax.tick_params(labelsize=15)
        self.update()

    def onscroll(self, event):
        if event.button == 'up':
            self.ind = (self.ind + 1) % self.slices
        else:
            self.ind = (self.ind - 1) % self.slices
        self.update()

    def click(self, event):
        self.rank = (self.rank + 1) % self.max_rank
        self.update()

    def update(self):

        self.im.set_data(self.signal[self.ind,:,:])
        self.ax1.set_ylabel('slice %s' % (self.ind + 1))
        self.ax1.set_title('Signal')
        self.im.axes.figure.canvas.draw()

        phase = self.storage[self.ind][self.rank][0]
        phase = self.storage[self.ind][self.rank][0]
        angle = self.storage[self.ind][self.rank][1]
        correlation_score = self.storage[self.ind][self.rank][2]
        coordinates = self.storage[self.ind][self.rank][3]
        intensities = self.storage[self.ind][self.rank][4]
        x,y = zip(*coordinates)

        self.im2.set_data(self.signal[self.ind,:,:])
        self.line1.set_offsets(coordinates)
        self.line1.set_array(intensities)
        self.ax2.set_title('Euler Angle: ({0:.0f},{1:.0f},{2:.0f}) \n Rank: {3} Correlation Score: {4:.2f}'.forma
t(angle[0],
                                                                                     angle[1],
                                                                                     angle[2],
                                                                                     self.rank,
                                                                                     correlation_score), fontsize
= 20)

        self.line1.set_clim(vmin=intensities.min(),vmax=intensities.max())
        #self.cbar2.set_ticks(cbar_ticks)
        self.cbar.update_normal(self.line1)
        self.cbar.draw_all()
        self.im2.axes.figure.canvas.draw()



from pyxem.utils.indexation_utils import get_nth_best_solution

def peaks_from_best_n_templates(tm_results, library, n = None):
    number_of_templates = tm_results.data.shape[0]
    if n == None:
        n = tm_results.data.shape[1]

    storage = []
    for i in range(number_of_templates):
        peaks_and_intensities = []

        for j in range(n):
            best_fit = get_nth_best_solution(tm_results.data[i], mode = 'template', rank=j)
            phase_names = list(library.keys())
            phase_index = int(best_fit[0])
            phase = phase_names[phase_index]
            simulation = library.get_library_entry(
                phase=phase,
                angle=tuple(best_fit[1]))
            peaks = simulation['pixel_coords']
            intensities = simulation['intensities']
            peaks_and_intensities.append([phase, best_fit[1], best_fit[2], peaks, intensities])
        storage.append(peaks_and_intensities)
    return storage

def plot_results_on_signal(tm_results, signal, library, *args, **kwargs):
    storage = peaks_from_best_n_templates(tm_results, library)

    fig, [ax1,ax2] = plt.subplots(1, 2, gridspec_kw={'width_ratios': [5, 8]})
    plt.axis('off')


    tracker = IndexTracker(ax1, ax2, test_data.data, storage, kwargs_for_signal = {"cmap" : "inferno_r", "vmax" :
0.1}, **kwargs)


    fig.canvas.mpl_connect('scroll_event', tracker.onscroll)
    fig.canvas.mpl_connect('button_press_event', tracker.click)
    plt.show()
    return tracker
```

In [ ]:

```
plot_results_on_signal(match_results, test_data, template_library, kwargs_for_template_scatter = {"cmap" : "virid
is", "s": 50, "marker" : "x"})
```

Plot Template Library index FT

In [ ]:

```
index = 9
orientation = tilt_series[index]
coordinates = template_library.get('GaSb')['pixel_coords'][index]
intensities = template_library.get('GaSb')['intensities'][index]

picture = np.zeros((test_data.data.shape[1:]))
fsize = np.asarray(picture.shape) * 2 - 1

for i in range (len(coordinates)):
    picture[coordinates[i,0], coordinates[i,1]] = intensities[i]

picture_FT = np.fft.fftshift(np.fft.fftn(picture, fsize))
```

In [ ]:

```
plt.figure()
plt.axis('off')
plt.imshow(np.imag(picture_FT), cmap = 'inferno')
plt.colorbar()
plt.title('Template: {} Imaginary'.format(index+1))
plt.show()
```

In [ ]:

```
x,y = zip(*coordinates)

plt.figure()
plt.axis('off')
plt.imshow(np.zeros(test_data.data.shape[1:]), cmap = 'inferno_r')
plt.scatter(x = x, y = y, c = intensities, s = 20,cmap = 'viridis')
plt.colorbar()
plt.title('Template: {} '.format(index+1))
plt.show()
```

In [ ]:

```
index = 21
test_data.data[index]

signal_FT = np.fft.fftshift(np.fft.fftn(picture, fsize))
```

In [ ]:

```
plt.figure()
plt.axis('off')
plt.imshow(np.imag(signal_FT), cmap = 'inferno_r')
plt.colorbar()
plt.title('Signal: {} Imaginary'.format(index+1))
plt.show()
```

Correlation Score Map plot

In [ ]:

```
image = np.asarray(match_results.data[:, :, 2], dtype = float)
image = image.T
plt.figure()
plt.imshow(image, cmap = 'inferno')
plt.colorbar()
plt.title("Correlation scores")
plt.xlabel("Crystal Index")
plt.ylabel("nth best match")
plt.plot()
```

C.3 Matlab Code

The following pages contain the matlab script in which all inverse pole figure plots presented in this report have been generated.

```
% Load pyxem results from .csv
datadir = 'C:\Users\eirik\Documents\PerfectTemplate';
fname = 'to_MTEX_orientations_correlation_score_Fast.csv';
file = fullfile(datadir, fname);

% Set crystal symmetry
cs = crystalSymmetry('m-3m', 'mineral', 'au'); %'-43m' 'mmm'
column_names = {'phase', 'euler1', 'euler2', 'euler3', 'score', 'y', 'x'};

%Load first file
sped = EBSD.load(file, 'cs', cs, 'ColumnNames', column_names);

%Load second file (for comparison)
datadir2 = 'C:\Users\eirik\Documents\simulert_data1deg';
fname2 = 'to_MTEX_orientations_correlation_score_0p03sigFast1degGaSb.csv';
file2 = fullfile(datadir2, fname2);

sped2 = EBSD.load(file2, 'cs', cs, 'ColumnNames', column_names);

%Set colormap preference
setMTEXpref('defaultColorMap', 'viridis')

ebsd_au = sped;
colors = ebsd_au.score;

% Import orientations and set symmetry
o = ebsd_au.orientations;%(1:30);
o2 = sped2.orientations;
o2.CS = crystalSymmetry('m-3m');
o.CS = crystalSymmetry('m-3m');

%Create unimodal ODF function from orientation data
odf = 0.7*unimodalODF(o, 'halfwidth', 1*degree);

%Define indices for plot
indices = cell(0,0);

for i = 1:length(o)
    number = ebsd_au.id(i);
    indices{end+1} = int2str(ebsd_au.id(i));
end

%Code below allows for extraction of Euler angle indices in place of
%numbers

%for i = 1:29
%    phi1 = o.phi1(i);
%    Phi = o.Phi(i);
%    phi2 = o.phi2(i);
%    indices{end+1} = strcat(sprintf('%.0f', phi1 * 180 / pi), ',',...
%    sprintf('%.0f', Phi * 180 / pi), ',', ...
%    sprintf('%.0f', phi2 * 180 / pi));
%end
```

1

```
% Plot IPF and annotate indices
figure
plotIPDF(o, vector3d.Z, 'antipodal')%, 'smooth')%, 'complete')%,

hold on
annotate(o, 'marker','s','MarkerSize', 10, 'MarkerFaceColor', 'r',...
        'label', indices, 'Color', 'r')
%hold on

%Plot second set of orientations, if interesting.
%plotIPDF(o2, vector3d.Z, 'markerfacecolor', 'r','markersize'...
%         ,5, 'antipodal')
%annotate(o2, 'marker','s','MarkerSize', 5, 'MarkerFaceColor', 'r', ...
% 'label', indices, 'Color', 'r')
mtexColorbar
```