Fakultät für
Mathematik, Informatik
und Naturwissenschaften

Magnetization distribution in superparamagnetic iron oxide nanoparticles

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Tobias Köhler
Master of Science
aus München

Berichter:	Professor Dr. Thomas Brückel
	Professor Dr. Joachim Mayer

Tag der mündlichen Prüfung: 03.05.2022

Abstract

Superparamagnetic iron oxide nanoparticles (SPIONs) have attracted considerable attention in the past due to their unique properties allowing for a wide range of applications in engineering and medicine, including adaptive dampers and cancer treatment methods. The mechanisms leading to the observed macroscopic properties are complex and depend on multiple factors, such as the particle size, the chemical composition and disorder and defects in the crystalline structure. Recently antiphase boundaries (APBs) have been suggested to strongly influence the macroscopic magnetic properties of iron oxide nanoparticles. In this work particles in a size range from 5 to 20 nm were investigated experimentally and the results were compared. Numerical simulations were used to study the impact of crystal lattice defects on the atomic spin structure and on X-ray powder diffraction patterns.

The combination of a variety of complementary techniques, including small-angle X-ray (SAXS) and neutron scattering (SANS), X-ray powder diffraction (XRD) with pair distribution function (PDF) analysis, transmission electron microscopy (TEM), Mössbauer spectroscopy and magnetometry, allowed the detailed investigation of the macroscopic and microscopic properties of the nanoparticles. A decrease in saturation magnetization with increasing particle size was found, in contrast to most of the existing literature on SPIONs. SANS showed the presence of a magnetically dead surface layer with similar thickness for particles with sizes of approximately 12 and 16 nm. Comparison with literature values suggested that this surface layer thickness is rather independent of the particle size. Thus the trend in the magnetization could not be explained by the existence of the surface layer alone. Further studies using XRD and PDF analysis were performed to assess the number of vacancies and the degree of vacancy ordering in the nanoparticles. While for the amount of vacancies no correlation with the particle size was found the vacancy ordering is more pronounced for the larger particles. The presence of APBs in the particles studied for this work was confirmed with XRD and high-resolution TEM. For the detection and quantification of APBs with XRD a model was used that was developed in this work on the basis of theoretical considerations and Debye scattering equation simulations. It could be shown that a larger number of APBs is directly related to the observed drop in magnetization. As shown in this work via Monte Carlo simulations this reduction in magnetization can be explained by strong spin disorder near the planar defect. The combination of the magnetically dead surface layer, the amount and ordering of vacancies as well as the presence and quantity of APBs was thus concluded to be the origin of the peculiar size dependence of the saturation magnetization observed for the particles used in this work.

Zusammenfassung

Superparamagnetische Eisenoxid-Nanopartikel haben aufgrund ihrer einzigartigen Eigenschaften in der Vergangenheit beträchtliche Aufmerksamkeit erfahren und finden breite Anwendung sowohl im technischen als auch im medizinischen Bereich, wie etwa in adaptiven Dämpfern und Methoden zur Behandlung von Krebs. Die mikroskopischen Ursachen für die makroskopischen Eigenschaften sind komplex und hängen von vielen Faktoren ab. Hierzu zählen neben der Partikelgröße und der chemischen Zusammensetzung auch Fehlordnungen und Defekte in der kristallinen Struktur. Sogenannte Antiphasengrenzen (APBs) wurden in neueren Arbeiten als ein wichtiger Einfluss auf die magnetischen Eigenschaften von Eisenoxid Nanopartikeln erkannt. In der vorliegenden Arbeit wurden Partikel mit Durchmessern von 5 bis 20 nm experimentell untersucht und die Ergebnisse verglichen. Numerische Simulationen dienten der Untersuchung des Einflusses von Kristallgitterdefekten auf die atomare Spinstruktur und auf Röntgen-Pulverdiffraktogramme.

Durch die Kombination verschiedener komplementärer Techniken, darunter Kleinwinkelstreuung mit Röntgenstrahlung (SAXS) und Neutronen (SANS), Röntgenpulverdiffraktometrie (XRD) mit Paarverteilungsfunktions-Analyse (PDF), Transmissionselektronenmikroskopie (TEM), Mössbauer-Spektroskopie und Magnetometrie, konnten die makroskopischen und mikroskopischen Eigenschaften der Nanopartikel eingehend untersucht werden. Es wurde eine Abnahme der Sättigungsmagnetisierung mit zunehmender Teilchengröße festgestellt, was im Gegensatz zur Mehrzahl der veröffentlichten Literatur steht. Mit SANS wurde eine magnetisch tote Oberflächenschicht mit ähnlicher Dicke für Partikel mit Durchmessern von etwa 12 und 16 nm beobachtet. Der Vergleich mit Literaturwerten deutet darauf hin, dass diese Oberflächenschichtdicke eher unabhängig von der Teilchengröße ist. Somit konnte der Trend in der Magnetisierung nicht allein durch die Existenz der Oberflächenschicht erklärt werden. Weitere Untersuchungen mittels XRD und PDF-Analyse wurden durchgeführt, um die Anzahl der Leerstellen und den Grad der Leerstellenordnung in den Nanopartikeln zu ermitteln. Während für die Anzahl der Leerstellen keine Korrelation mit der Partikelgröße festgestellt werden konnte, ist die Ordnung der Leerstellen bei größeren Partikeln stärker ausgeprägt. Die Existenz von APBs in den für diese Arbeit untersuchten Partikeln wurde mit XRD und hochauflösender TEM bestätigt. Für den Nachweis und die Quantifizierung von APBs mit XRD wurde ein Modell verwendet, das in dieser Arbeit auf der Grundlage von theoretischen Überlegungen und Simulationen der Debye-Streugleichung entwickelt wurde. Es konnte gezeigt werden, dass eine größere Anzahl von APBs in direktem Zusammenhang mit dem beobachteten Abfall der Magnetisierung steht. Wie in dieser Arbeit anhand von Monte Carlo Simulationen gezeigt wurde, lässt sich dieser Rückgang der Magnetisierung durch starke Spin-Unordnung in der Nähe des planaren Defekts erklären. Somit konnte als Ursache für die Größenabhängigkeit der Sättigungsmagnetisierung der in dieser Arbeit verwendeten Nanopartikel gerade die Kombination aus dem Vorhandensein einer magnetisch toten Oberflächenschicht, der Menge und der Anordnung von Leerstellen sowie das Auftreten und die Anzahl von APBs ermittelt werden.

Contents

Chapter 1

Introduction

1.1 Motivation

The small size of iron oxide nanoparticles leads to several properties that are of interest both from a technological as well as from a purely scientific point of view. The most important feature is the phenomenon of superparamagnetism. In mechanical engineering ferrofluids, i.e. concentrated dispersions of superparamagnetic iron oxide nanoparticles are used as seals and adaptive dampers.[1–3] A patent by Apple Inc. illustrates a further use case of ferrofluids. In the patent induction charging is described where ferrofluids are used as an intermediate layer between transceiver and receiver coils.[4] This helps to reduce coil misalignment issues that lead to less efficient charging of devices. Another large area is the field of nanomedicine including imaging diagnostics using magnetic nanoparticles as contrast agents e.g. in magnetic resonance imaging (MRI)[5–11], magnetic drug targeting[10,12,13] or cancer treatment methods based on magnetic hyperthermia[14–20]. The latter method is based on the heat generation in iron oxide nanoparticles in alternating external magnetic fields that leads to the destruction of cancerous tissue or makes it more susceptible for further treatment. Clinically approved treatment is already offered for glioblastoma, an aggressive type of brain cancer.[21,22]

The same principle of local heating albeit different application using superparamgnetic iron oxide nanoparticles has also been used for polymerization[23] and catalysis.[24] Most important in all these processes is the specific absorption rate (SAR) that depends on the magnetic properties as well as the concentration of the particles and is a measure of the heating efficiency. A large saturation magnetization is associated with a larger SAR and thus better heating. Additionally, defects and deviations from the perfect crystal structure have been shown to lead to better heating performance.[25] It is evident that a precise understanding of the microscopic and macroscopic properties of iron oxide nanoparticles is necessary in order to improve existing applications and enable the development of new ones.

This work aims at expanding the knowledge of the physical properties of superparamagnetic iron oxide nanoparticles, specifically with regards to the complex interplay between the crystal structure and the magnetization distribution under consideration of varying particle sizes.

1.2 State of research

Numerous studies have addressed the question of the magnetization distribution within iron oxide nanoparticles and the size dependence of the magnetic properties.[20,25–49] In this section an overview of the most relevant works for this thesis is given.

A natural starting point for size dependent magnetic properties of nanoparticles is the investigation of the surface region, since the surface-to-volume ratio is the most obvious size-dependent parameter. An early investigation dealing with surface effects by Hendriksen et al.[26] showed on the basis of spin-wave calculations that for very small particles the magnetization in the surface layer decreases faster with increasing temperature than that of the particle center. Therefore, the magnetization as a function of the temperature $M(T)$ is different for small particles than the expected bulk behaviour. An early account of size dependent magnetic properties of maghemite particles was given by Berkowitz et al.[27] They found a significant decrease in the measured saturation magnetization with decreasing particle size for particles below 20 nm. Coey et al.[28] later reported for 6 nm maghemite particles a saturation magnetization at 4.2 K of 59 $\mathrm{Am^2/kg_{Magh.}}$ in addition to a reduced iron atom fraction. The reduction in the magnetization was attributed to the presence of spin canting in a surface layer. This conclusion was later supported by Morrish et al.[29] who studied different sizes of spherical maghemite nanoparticles. On the basis of Mössbauer spectroscopy they provided a value of the surface layer thickness of 0.4 nm for particles with diameters of 6.5 nm. Parker et al.[30] acknowledged that surface spins show reduced exchange interactions compared to the spins in the core, however they argue that spin canting is not a surface effect, but is instead a volume effect induced by the finite size of the particles. Morales et al.[31] support this view of spin canting in the particle interior. However they observe a decrease in saturation magnetization from spherical particles of 120(10) nm in diameter to larger elongated particles. They also find superstructure peaks in X-ray powder diffraction patterns relating to vacancy ordering for the smaller particles that are absent for the larger ones. Thus, they conclude that vacancy ordering and structural order both increase the magnetization and propose that these effects are not necessarily dependent on the particle size.

It has to be emphasized that in all the described works the particle sizes, shapes and synthesis methods vary quite strongly. This was also recognized by Li et al.[33] who performed a more systematic study for different particle sizes. They prepared maghemite particles via flame spray pyrolysis with sizes of 6, 8, 10, 11, 13, 18 and 53 nm. Here, they found an increasing vacancy ordering with increasing particle size up to the particle size of 18 nm where full vacancy order in the space group $P4_32_12$ is reached. Magnetometry data on these particles suggest a saturation magnetization of nearly bulk values for particles with diameters $\geq$ 13 nm. They thus conclude that the observed saturation magnetizations are a direct consequence of the vacancy ordering as well as possible crystal facets for the larger particles leading to a reduced surface region of disordered spins. Roca et al.[35] produced iron oxide nanoparticles by the thermal decomposition method with a mean size of 6.4 nm coated with oleic acid. For these particles a room temperature saturation magnetization of 84 $\mathrm{Am^2/kg_{Ferrite}}$ was recorded, showing that even at this particle size almost bulk like magnetization is possible for perfectly crystalline particles. It was also concluded that the oleic acid coating reduces the surface spin canting through inorganic-organic interactions that are not entirely understood yet. Daou et al.[34] propose on the basis of Mössbauer spectroscopy and IR spectroscopy of 39(5) nm particles coated with a carboxylate layer a compositional core-shell structure with the more oxygen rich maghemite phase at the particle surface and the stoichiometric magnetite phase in the core. Further, they suggested the presence of a spin canted surface layer of 0.55(15) nm. This notion of a chemical core-shell structure has very recently been disputed by Andersen et al.[50] Instead a more gradual transition from a more iron rich core to a more iron depleted surface is proposed, while the particles are structurally coherent. Dutta et al.[42] observed an increase in the saturation magnetization from $15\,\mathrm{Am^2/kg_{Ferrite}}$ for particles with diameters of 4 nm to $62\,\mathrm{Am^2/kg_{Ferrite}}$ for 12 nm nanoparticles. This increase was attributed to a non-magnetic surface layer of 0.68 nm thickness for particles with 6, 8, 10 and 12 nm and a slightly larger surface layer of 0.86 nm for the smallest particles. However, these estimates were made only on the basis of the magnetometry measurements.

New support to the theory of a surface layer dominating the magnetic properties of nanoparticles was presented by Krycka et al.[37] With uniaxial polarization analysis in SANS they found a magnetic surface layer of 1.0 to 1.5 nm in 9.0 nm spherical particles, where the net magnetization is oriented perpendicular to the magnetization orientation of the particle core. However, Michels et al.[40] raised serious concerns regarding the methodology of this work and argue that the presented results are not supported by the neutron data analysis. Furthermore, Disch et al.[43] analyzed 9.9 nm spherical particles with SANS with polarized neutrons (SANSPOL) and found a surface layer with spin disorder of only 0.3(1) nm. In this work it is also emphasized that a reduced magnetization in the particle core contributes to a larger extent to the overall magnetization than the surface layer. As mechanisms for the reduced core magnetization spin canting around defects is proposed. Herltischke et al.[51] studied 7.4 nm spherical particles with SANSPOL and nuclear forward scattering and arrived at the conclusion that surface effects are negligible and spin disorder distributed in the particle cores is responsible for the observed reduced magnetization. More recently Zákútna et al.[44] found a magnetically dead surface layer of 0.28(6) nm at an applied field of 1.2 T in cobalt ferrite nanoparticles. Additionally, they report a field dependence of the surface layer thickness such that it increases with decreasing fields.

Apart from canting around vacancies and defects another mechanism has been proposed by Levy et al.[45] and Wetterskog et al.[46], namely the presence of antiphase boundaries (APBs). In the latter work high resolution TEM (HRTEM) provided evidence for the presence of APBs in cubic iron oxide nanoparticles with edge lengths of 23(3) nm. They proposed that these defects, which are commonly found in magnetite thin films, emerge due to the increase of oxygen-to-iron ratio during the thermal decomposition synthesis procedure leading to the transition from wüstite over magnetite to maghemite and that they are mostly responsible for the observed reduced saturation magnetization. In the work of Levy et al. the particles were synthesized by a seeded growth method in the size range of 6 to 18 nm. Here, a trend of decreasing saturation magnetization with increasing particle size was found which was related to primarily the presence of lattice strain and APBs. More experimental evidence for APBs in iron oxide nanoparticles was provided by Nedelkoski et al.[47] via atomically resolved HRTEM. In addition, they performed Monte Carlo simulations of the effect of antiphase boundaries on the spin structure of magnetite nanoparticles, claiming that a magnetic multi-domain state forms that reduces the total magnetization by 26 % as compared to a particle without this defect. Even larger reduction of 34 % is proposed for a non-planar APB.

1.3 Concept

The above overview of the literature over the past few decades shows that while progress has been made the internal properties of iron oxide nanoparticles and their influence on the saturation magnetization are still far from being clear. The goal of this thesis is thus to gain a complete understanding of the magnetization distribution within these particles especially under consideration of the particle size. In this work also an emphasis is placed on the complex interplay of size, composition, structural defects and magnetic properties of iron oxide nanoparticles.

To this end a systematic study of different particle sizes with complementary techniques is performed in combination with computer simulations. The focus of this work lies on particles produced by a thermal decomposition method, as they can be produced in a range of particle sizes with small size distributions and in large quantities.[52] As shown by Wetterskog et al.[46] particles synthesized in this way are prone to develop antiphase boundaries (APBs). Therefore, in the first part of this work Monte Carlo simulations are used to investigate the effect of APBs on the spin structure of maghemite nanoparticles. In the second part X-ray diffraction simulations are carried out to determine the influence of APBs on the diffraction patterns. Finally, experimental studies are performed on particles in the size range of 5 to 20 nm. Here, first the particle sizes and size distributions are determined via small-angle scattering and electron microscopy. Magnetometry, small-angle neutron scattering, X-ray diffraction, Mössbauer spectroscopy and high resolution transmission electron microscopy are used to obtain information on the chemical composition, the structural and the magnetic properties.

Chapter 2

Theoretical background

2.1 Magnetism

2.1.1 Fundamentals

2.1.1.1 Magnetic moment

For a single atomic electron the magnetic moment is a consequence of angular momentum resulting from an orbital and a spin contribution. In analogy to classical electromagnetic theory the angular momentum of a charge leads to a magnetic dipole moment. However, it should be noted that magnetism is an entirely quantum mechanical phenomenon as shown in the Bohr-van Leeuwen theorem.[53] An electron in an orbital around a nucleus possesses angular momentum, denoted by $\boldsymbol{l}$ due to the orbital motion. In this case the magnetic moment relates to the angular momentum according to

$$\boldsymbol{\mu}_l = -\frac{\mu_B}{\hbar}\boldsymbol{l}, \tag{2.1}$$

where $\boldsymbol{l} = \boldsymbol{r} \times \boldsymbol{p}$ is the orbital angular momentum operator defined by the cross product of the position operator $\boldsymbol{r}$ and the momentum operator $\boldsymbol{p}$ and $\mu_B = e\hbar/2m_e$ is the Bohr magneton, with e the elemental charge, $\hbar$ the reduced Planck constant and m_e the mass of the electron. The spin angular momentum is denoted by $\boldsymbol{s}$ and the resulting magnetic dipole moment is

$$\boldsymbol{\mu}_s = -\frac{g_e\mu_B}{\hbar}\boldsymbol{s}, \tag{2.2}$$

where $g_e \approx 2$ is the Landé g-factor. The total magnetic moment of an atomic electron is the sum of both contributions (neglecting spin-orbit coupling), i.e.

$$\boldsymbol{\mu} = \boldsymbol{\mu}_l + \boldsymbol{\mu}_s = -\frac{\mu_B}{\hbar}(\boldsymbol{l} + g_e\boldsymbol{s}). \tag{2.3}$$

2.1.1.2 Magnetic moment in a field

The quantum mechanical Hamiltonian for an atom with atomic number Z and one electron is given by

$$\mathcal{H}_0 = \frac{\boldsymbol{p}^2}{2m} - \frac{Ze^2}{4\pi\epsilon_0|\boldsymbol{r}|}, \tag{2.4}$$

where the first term corresponds to the kinetic energy with the momentum operator $\boldsymbol{p}$ and the last term corresponds to the Coulomb potential with the radial distance between nucleus and electron $|\boldsymbol{r}|$ and the permittivity of free space ϵ_0.[54] Application of an uniform magnetic field leads to a modification of the original

Hamiltonian expressed in an additional term due to the interaction between the field and the spin magnetic moment of the system and a modification of the kinetic energy through the resulting vector potential, $\boldsymbol{A}$.[55] Hence the resulting Hamiltonian is given by

$$\begin{aligned}\mathcal{H} &= \frac{(\boldsymbol{p}+e\boldsymbol{A}(\boldsymbol{r}))^2}{2m} - (\boldsymbol{\mu}_s \cdot \boldsymbol{B}) - \frac{Ze^2}{4\pi\epsilon_0|\boldsymbol{r}|} \\ &= \frac{(\boldsymbol{p}+e\boldsymbol{A}(\boldsymbol{r}))^2}{2m} + \frac{\mu_B g_e}{\hbar}\boldsymbol{s}\cdot\boldsymbol{B} - \frac{Ze^2}{4\pi\epsilon_0|\boldsymbol{r}|}.\end{aligned} \tag{2.5}$$

The magnetic field can be represented as the curl of a vector potential, i.e. $\boldsymbol{B} = \nabla \times \boldsymbol{A}$.[56] The Coulomb gauge, i.e. setting the divergence of the vector field $\boldsymbol{A}$ to zero, is fulfilled by $\boldsymbol{A}(\boldsymbol{r}) = \frac{1}{2}(\boldsymbol{B}\times\boldsymbol{r})$.[55] This allows the momentum operator and the vector potential to commute. The above equation can be rearranged to

$$\mathcal{H} = \frac{\boldsymbol{p}^2}{2m} + \frac{e}{2m}\boldsymbol{p}\cdot(\boldsymbol{B}\times\boldsymbol{r}) + \frac{e^2}{2m}\left(\frac{1}{2}\boldsymbol{B}\times\boldsymbol{r}\right)^2 + \frac{\mu_B g_e}{\hbar}\boldsymbol{s}\cdot\boldsymbol{B} - \frac{Ze^2}{4\pi\epsilon_0|\boldsymbol{r}|}. \tag{2.6}$$

Identifying the first and the last term as the original Hamiltonian of eq. 2.4 yields

$$\mathcal{H} = \mathcal{H}_0 + \frac{e}{2m}\boldsymbol{p}\cdot(\boldsymbol{B}\times\boldsymbol{r}) + \frac{e^2}{2m}\left(\frac{1}{2}\boldsymbol{B}\times\boldsymbol{r}\right)^2 + \frac{\mu_B g_e}{\hbar}\boldsymbol{s}\cdot\boldsymbol{B}. \tag{2.7}$$

Using the property of the triple product in the second term that it is invariant under a circular permutation of the arguments it can be written as $\boldsymbol{p}\cdot(\boldsymbol{B}\times\boldsymbol{r}) = \boldsymbol{B}\cdot(\boldsymbol{r}\times\boldsymbol{p}) = \boldsymbol{B}\cdot\boldsymbol{l}$, where $\boldsymbol{l}$ is the orbital angular momentum operator. This is possible because of the commutation relation between the momentum and position operator $[r_i, p_i] = i\hbar\delta_{ij}$, i.e. the components with different indices commute. Inserting into eq. 2.7 gives

$$\mathcal{H} = \mathcal{H}_0 + \frac{e}{2m}\boldsymbol{B}\cdot\boldsymbol{l} + \frac{e^2}{2m}\left(\frac{1}{2}\boldsymbol{B}\times\boldsymbol{r}\right)^2 + \frac{\mu_B g_e}{\hbar}\boldsymbol{s}\cdot\boldsymbol{B}. \tag{2.8}$$

Finally, with the definition of the Bohr magneton and combining the terms including angular momenta the Hamiltonian reads

$$\mathcal{H} = \mathcal{H}_0 + \frac{\mu_B}{\hbar}(\boldsymbol{l} + g_e\boldsymbol{s})\cdot\boldsymbol{B} + \frac{e^2}{2m}\left(\frac{1}{2}\boldsymbol{B}\times\boldsymbol{r}\right)^2. \tag{2.9}$$

Generalization of eq. 2.9 to a system with n electrons is straightforward if the spin-orbit coupling of the electrons is neglected.[55] Then the individual orbital and the spin angular momenta are added according to $\boldsymbol{L} = \sum_i^n \boldsymbol{l}_i$ and $\boldsymbol{S} = \sum_i^n \boldsymbol{s}_i$. With that the total angular momentum is introduced as $\boldsymbol{J} = (\boldsymbol{L} + g_e\boldsymbol{S})$, giving[53]

$$\mathcal{H} = \mathcal{H}_0 + \frac{\mu_B}{\hbar}g_J\boldsymbol{J}\cdot\boldsymbol{B} - \frac{e^2}{2m}\sum_{i=1}^{n}\left(\frac{1}{2}\boldsymbol{B}\times\boldsymbol{r}_i\right)^2. \tag{2.10}$$

with the Landé factor g_J given by

$$g_J = \left(1 + \frac{J(J+1) - L(L+1) + S(S+1)}{2J(J+1)}\right)\hbar^2, \tag{2.11}$$

where $g_e = 2$ was used and the eigenvalues of the angular momentum operators according to $J(J+1)\hbar^2$ for $\boldsymbol{J}^2$, $L(L+1)\hbar^2$ for $\boldsymbol{L}^2$ and $S(S+1)\hbar^2$ for $\boldsymbol{S}^2$.[53] Minimization of the angle between the total angular momentum and the applied field vector, i.e. alignment of the magnetic moment with the field leads to a decrease of the energy as can be seen from the second term of eq. 2.10, which is called Zeeman term that describes this *paramagnetic* effect.[55] The potential energy of the magnetic moment in a field is given by

$$E_{\text{Zeeman}} = -\frac{\mu_B}{\hbar} g_J \boldsymbol{J} \cdot \boldsymbol{B} = -\boldsymbol{\mu} \cdot \boldsymbol{B}. \tag{2.12}$$

The second term of eq. 2.10 is called *diamagnetic* term as it relates to an increase in energy due to the applied field and can be shown to lead to a magnetization opposite to the applied field.[53]

2.1.1.3 Magnetization and susceptibility

If the paramagnetic contribution to the magnetic moment is not zero, e.g. for the partially filled orbitals in 3d transition metals[55], it can be assumed that it is much larger than the diamagnetic term, which therefore can be neglected in the following. For localized magnetic moments, as can be assumed for the compounds used in this thesis, the volume magnetization corresponds to the sum over the N individual magnetic moments in the volume V

$$\boldsymbol{M} = \frac{1}{V}\sum_i^N \boldsymbol{\mu}_i. \tag{2.13}$$

This is easily transformed into the frequently occurring mass magnetization by dividing by the density of the material. For maximum alignment with a uniform field the magnetic moment is given by

$$m_{max} = \mu_B g_J J\hbar, \tag{2.14}$$

$J\hbar$ being the maximum of the component of the total angular momentum, i.e. the maximum possible magnetic quantum number.[53] Therefore, the volume saturation magnetization is given by

$$M_{sat} = \frac{1}{V}\sum_i^N m_{i,max}. \tag{2.15}$$

The response of the magnetization to an applied field is described by the magnetic susceptibility χ according to

$$\boldsymbol{M} = \chi \boldsymbol{H}, \tag{2.16}$$

where $\boldsymbol{H}$ is the external field. In general χ is negative for diamagnetic materials and positive for paramagnetic materials.[54] For a distribution of magnetic moments as described by $\boldsymbol{M}$ the external field is influenced by the magnetic moments inside this body resulting in an internal magnetic field given by

$$\boldsymbol{B} = \mu_0(\boldsymbol{H} + \boldsymbol{M}), \tag{2.17}$$

where μ_0 is the permeability of free space such that in a vacuum the two vector fields are related by this proportionality constant, i.e. $\boldsymbol{B} = \mu_0 \boldsymbol{H}$.[53]

2.1.1.4 Brillouin and Langevin functions

A description of the temperature and field dependence of the magnetization is obtained by a consideration of the possible states of the system. For a single paramagnetic atom there are $J(J+1)$ possible states with eigenvalues of the J_z component $m_J \hbar$, if spin orbit coupling is neglected. Thus the eigenvalues of the Hamiltonian corresponding to a paramagnetic moment in a field are the energies according to[54]

$$E = \mu_B g_J m_J B. \tag{2.18}$$

With the partition function

$$Z = \sum_{m_J} e^{-E\frac{1}{k_B T}} = \sum_{m_J} e^{-\mu_B g_J m_J B \frac{1}{k_B T}} \tag{2.19}$$

the general behaviour of the magnetic moment can be shown to follow

$$M = M_{sat}\left[\frac{2J+1}{2J}\coth\left(\frac{2J+1}{2J}y\right) - \frac{1}{2J}\coth\frac{y}{2J}\right] = M_{sat}B_J(y), \tag{2.20}$$

where $y = \frac{g_J \mu_B B}{k_B T} J$ and $B_J(y)$ is the Brillouin function.[53–55,57] This function simplifies to the Langevin function for $J \to \infty$ according to[53]

$$M = M_{sat}\left[\coth y - \frac{1}{y}\right], \tag{2.21}$$

where in this limit $g_J \mu_B J$ can be replaced by the much larger macro- or superspin.[58] This is used to describe non-interacting superparamagnetic nanoparticles above the blocking temperature (see section 2.1.3).

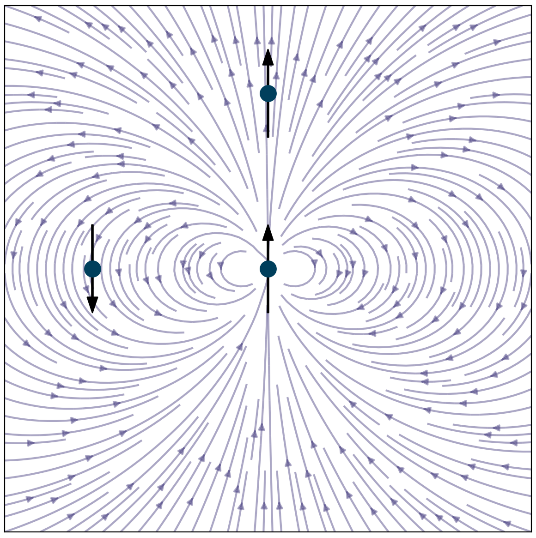


Fig. 2.1 Magnetic dipole field originating from a magnetic moment in the center. The dots symbolize magnetic moments in this dipole field with the resulting orientation (not considering their own dipole field). It should be noted that the field lines do not terminate as is drawn in the figure for better visibility but are continuous.

2.1.2 Collective magnetism

2.1.2.1 Dipole-dipole interactions

From magnetostatic theory it is known that a magnetic dipole generates a field that can be described by

$$\boldsymbol{B}_{\text{dipole}} = \frac{\mu_0}{4\pi}\left(\frac{3(\boldsymbol{\mu}_j \cdot \boldsymbol{R}_{ij})\boldsymbol{R}_{ij} - \boldsymbol{R}_{ij}^2\boldsymbol{\mu}_j}{R_{ij}^5}\right), \tag{2.22}$$

where $\boldsymbol{R}_{ij} = \boldsymbol{R}_i - \boldsymbol{R}_j$ is the distance vector between points $\boldsymbol{R}_i$ and $\boldsymbol{R}_j$. R_{ij} is the corresponding distance.[55,56] Thus using eq. 2.12 the potential energy of a moment $\boldsymbol{\mu}_i$ due to magnetic dipole interactions with all the other moments in the magnetic body is given by

$$E_{dd} = -\boldsymbol{\mu}_i \cdot \boldsymbol{B}_{\text{dipole}}(\boldsymbol{R}_j) = -\frac{\mu_0}{4\pi}\sum_j \frac{1}{R_{ij}^3}\left[3(\boldsymbol{\mu}_i \cdot \hat{\mathbf{R}}_{ij})(\boldsymbol{\mu}_j \cdot \hat{\mathbf{R}}_{ij}) - \boldsymbol{\mu}_i \cdot \boldsymbol{\mu}_j\right], \tag{2.23}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ denote magnetic dipoles separated by a distance R_{ij} with direction given by the unit vector $\hat{\boldsymbol{R}}_{ij}$.[53,55]

The energetically most favourable configuration for two dipoles with the same moment is obtained by parallel alignment if the vector $\hat{\boldsymbol{R}}_{ij}$ is also parallel to the moments. If $\hat{\boldsymbol{R}}_{ij} \perp \boldsymbol{\mu}$ antiparallel alignment of the moments is favoured. This can be understood by considering that one moment is subject to the dipole field of the other. The field lines form a closed loop, thus depending on the relative placement of the moments this dipole field is either parallel to the moment orientation or antiparallel or anything in between (fig. 2.1). Since the dipole field is often opposite or at least at an angle to the alignment of the magnetic moments in an applied field it acts against the magnetization. For two isolated moments this energetic contribution is comparably small (approx. -2.15×10^{-24} J for $\hat{\boldsymbol{R}}_{ij} \parallel \boldsymbol{\mu}$, $\boldsymbol{\mu}_i \parallel \boldsymbol{\mu}_j$, $\mu_i = \mu_j = 1\mu_B$, and $R_{ij} = 0.2\,\text{nm}$) and is not able to explain the magnetic order observed in many materials at room temperature and above.[54] Nevertheless, its long range nature (decay $\propto R_{ij}^{-3}$) and the sum of many individual contributions can lead to non-negligible effects on the collective alignment of magnetic moments. Dipole interactions and the resulting stray fields are responsible for the formation of magnetic domains (section 2.1.2.8) and for shape anisotropy in finite size systems (section 2.1.2.5).[53,55]

2.1.2.2 Direct exchange

Direct exchange results in the alignment of magnetic moments for two atoms with unpaired electrons whose orbitals overlap. Since electrons are Fermions they are indistinguishable, hence their combined wave function must be totally antisymmetric upon exchange of the two. The total wave function is given by the combination of a spatial part and a spin part. The latter is defined by the total spin S and the quantum number m_s giving the state $|S; m_s\rangle$. The possible combinations of the two spin states lead to an antisymmetric singlet state with $S = 0$ and a symmetric triplet state with $S = 1$ given by

$$\begin{aligned} |0;0\rangle &= \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle) \\ |1;1\rangle &= |\uparrow\uparrow\rangle \\ |1;0\rangle &= \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle) \\ |1;-1\rangle &= |\downarrow\downarrow\rangle. \end{aligned} \tag{2.24}$$

To achieve an antisymmetric total wave function the triplet spin configurations that are symmetrical under exchange must be combined with an antisymmetrical spatial wave function. For the singlet configuration the opposite is true, which results in the wave functions

$$\Psi_S = \frac{1}{\sqrt{2}}\left(\psi_a(\boldsymbol{r}_1)\psi_b(\boldsymbol{r}_2) + \psi_a(\boldsymbol{r}_2)\psi_b(\boldsymbol{r}_1)\right) X_S \tag{2.25}$$

$$\Psi_T = \frac{1}{\sqrt{2}}\left(\psi_a(\boldsymbol{r}_1)\psi_b(\boldsymbol{r}_2) - \psi_a(\boldsymbol{r}_2)\psi_b(\boldsymbol{r}_1)\right) X_T, \tag{2.26}$$

where X_S and X_T denote the spin parts of the wave functions for the singlet and the triplet configuration, respectively.[53] The Hamiltonian for a system of interacting particles is given by

$$\mathcal{H} = \sum_{i=1}^{2} \frac{\boldsymbol{p}_i^2}{2m} + V(\boldsymbol{r}_1, \boldsymbol{r}_2), \tag{2.27}$$

which on a first glance seems to be independent of the spin contribution.[55] However, as shown above the spin configuration directly influences the spatial wave function, thus it should be possible to construct an effective Hamiltonian resulting in the same energy eigenvalues but with the explicit dependence on the spin configuration included in the equation. The energies associated with the singlet and the triplet state are obtained by

$$E_T = \int \Psi_T^* \mathcal{H} \Psi_T d\boldsymbol{r}_1 \boldsymbol{r}_2 \tag{2.28}$$

$$E_S = \int \Psi_S^* \mathcal{H} \Psi_S d\boldsymbol{r}_1 \boldsymbol{r}_2, \tag{2.29}$$

where E_S and E_T corresponds to the energy of the singlet and the triplet state respectively. The asterisks denote the complex conjugate. To construct the spin dependent Hamiltonian the scalar product of the two spin operators is useful. The eigenvalue of this quantity can be obtained from the known eigenvalues of the squared total spin operator $\boldsymbol{S}^2 = S(S+1)\hbar^2$, where $\boldsymbol{S} = \boldsymbol{s}_1 + \boldsymbol{s}_2$ with $\boldsymbol{s}_i$ the spin operator of electron i and the eigenvalue of the squared single spin operator $\boldsymbol{s}^2 = s(s+1)\hbar^2$.[53,54] The square of the sum of the two spins can be expanded according to

$$\boldsymbol{S}^2 = (\boldsymbol{s}_1 + \boldsymbol{s}_2)^2 = \boldsymbol{s}_1^2 + \boldsymbol{s}_2^2 + 2\boldsymbol{s}_1 \cdot \boldsymbol{s}_2. \tag{2.30}$$

Rearranging gives

$$\frac{1}{2}(\boldsymbol{S}^2 - \boldsymbol{s}_1^2 - \boldsymbol{s}_2^2) = \boldsymbol{s}_1 \cdot \boldsymbol{s}_2, \tag{2.31}$$

which for the eigenvalues evaluate to

$$\frac{1}{2}\left(S(S+1) - \frac{3}{4} - \frac{3}{4}\right)\hbar^2 = \lambda, \tag{2.32}$$

where λ denotes the eigenvalue of $\boldsymbol{s}_1 \cdot \boldsymbol{s}_2$. Thus,

$$\lambda = \frac{1}{2}S(S+1)\hbar^2 - \frac{3}{4}\hbar^2. \tag{2.33}$$

This results in an eigenvalue of $-\frac{3}{4}\hbar^2$ for $S = 0$, i.e. the singlet state and $\frac{1}{4}\hbar^2$ for $S = 1$, i.e. the triplet state. Therefore, if $\boldsymbol{s}_1 \cdot \boldsymbol{s}_2$ is to be used in the effective Hamiltonian it has to be made sure that the correct energies are obtained for the singlet and the triplet state. A Hamiltonian satisfying this requirement is provided by[53,54]

$$\mathcal{H} = \frac{1}{4}(E_S + 3E_T) - (E_S - E_T)\boldsymbol{s}_1 \cdot \boldsymbol{s}_2, \tag{2.34}$$

which upon insertion of the eigenvalue for the triplet state, $\frac{1}{4}\hbar$, returns the correct energy eigenvalue E_T. The same is true for the singlet state. Another useful quantity to define is the exchange integral, J, according to

$$J = \frac{1}{2}(E_S - E_T) = \int \psi_a^*(\boldsymbol{r}_1)\psi_b^*(\boldsymbol{r}_2)\mathcal{H}\psi_a(\boldsymbol{r}_1)\psi_b(\boldsymbol{r}_2)d\boldsymbol{r}_1 d\boldsymbol{r}_2, \tag{2.35}$$

where eqs. 2.25 and 2.26 were used as well as the assumption that the spin components are normalized. This finally results in an effective spin Hamiltonian obtained from the second term in eq. 2.34 and by insertion of eq. 2.35

$$\mathcal{H}_{\text{spin}} = -2J\boldsymbol{s}_1 \cdot \boldsymbol{s}_2. \tag{2.36}$$

For the generalization of many such interactions in a solid it is postulated that the above expression holds for any neighbouring pair of atoms resulting in the Heisenberg model, where the spin Hamiltonian is given by[53,55]

$$\mathcal{H}_{\text{Heisenberg}} = -2\sum_{i>j} J_{ij}\boldsymbol{s}_i \cdot \boldsymbol{s}_j. \tag{2.37}$$

In the semi-classical approach the spin operators are replaced by classical vectors. Direct exchange is rare in solids as the atomic orbitals are too far from each other, however magnetic interactions can also be mediated by hopping of electrons. This is the basis of indirect exchange interactions discussed in the following. Two important mechanisms for this work are superexchange and double exchange. Other indirect exchange interactions that are not treated here are the Rudermann-Kittel-Kasuya-Yosida (RKKY) and the Dzyaloshinski-Moriya interactions.[55]

2.1.2.3 Superexchange

This exchange mechanism allows the establishment of magnetic order despite a distance between the magnetic ions that would be too large for direct exchange. The exchange interaction is instead mediated by an ion such as oxygen.[55] This type of interaction can again be described by an effective Heisenberg Hamiltonian as given in eq. 2.37, however the interpretation of J is different.[55] Here the exchange constant is related to the kinetic energy obtained by delocalization of the electron and the Coulomb energy.[53] Cation-anion-cation bonds with bond angles of 180° or 90° represent two limiting cases which lead to antiferromagnetic and ferromagnetic alignment of the cation magnetic moments, respectively. Intermediate angles of 125° that are important in the sublattice coupling of the maghemite and magnetite structures also lead to antiferromagnetic alignment. These qualitative descriptions of the magnetic ordering can be inferred from the Goodenough-

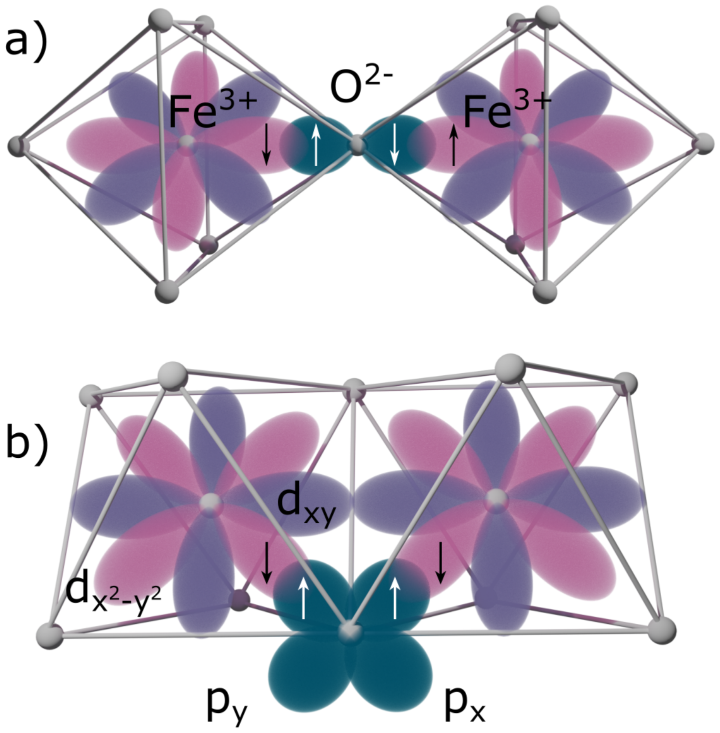


Fig. 2.2 In a) the 180° $Fe^{3+} - O - Fe^{3+}$ bond is shown, where the iron cations are placed in the center of oxygen octahedra being found in APBs. The partial covalent bond between the $d_{x^2-y^2}$ (orange) and the p_x-orbital (blue) leads to antiferromagnetic alignment of the spins. b) The 90° bond is found between two edge sharing octahedra. The partial covalent bonds are formed between $d_{x^2-y^2}$ and p_x for one cation and $d_{x^2-y^2}$ and p_y for the other. This time ferromagnetic alignment is favoured.

Kanamori-Anderson (GKA) rules, which are based on the interactions of atomic orbitals.[59]

The limiting cases are realized in the maghemite structure. The 180° cation-anion-cation bond occurs at antiphase boundaries where two oxygen octahedra with Fe^{3+}-ions in the center that are connected on a corner (fig. 2.2a). In the octahedral environment the degeneracy of the five d-orbitals is lifted leading to t_{2g}-orbitals (d_{xy}, d_{yz}, and d_{zx}, dark purple in fig. 2.2) with lower and e_g-orbitals ($d_{x^2-y^2}$, light purple in fig. 2.2, and d_{z^2}) with higher energy. The $d_{x^2-y^2}$-orbital points towards the oxygen ligands along the x-axis, the same orientation as the p_x orbital of the oxygen ion. This leads to partial covalent bonds at the cations. Due to the Pauli exclusion principle the two oxygen electrons in the p_x-orbital have to occupy different spin states thus leading to antiparallel alignment of the $d_{x^2-y^2}$-electrons of the cations. This results in the antiferromagnetic configuration.

The 90° bond can be found between neighbouring Fe^{3+}-ions in octahedral edge-connected chains (fig. 2.2b). One contribution to the resulting spin configuration in this geometry is again the formation of covalent bonds at the cations. However, due to the 90° bond the bonds are formed with both p-orbitals of the oxygen ion leading to ferromagnetic alignment of the cation spins. A competing interaction is due to the proximity of d_{xy}-orbitals of the cations which favour antiparallel spin orientations via the delocalization of electrons. Thus in general the 90° cation-anion-cation bond leads to ferromagnetic alignment, however in some cases antiferromagnetic configurations may be energetically more favourable.[59] Determination of the exact exchange constants for the interactions is in general a complex problem and they are often found empirically and validated by comparing the calculated Curie temperature resulting from these constants with experimental values or via ab-initio calculations.[60,61]

2.1.2.4 Double exchange

Double Exchange is a form of indirect exchange interaction between two metal ions of different valence state separated by a non-magnetic anion.[53] In case of a $Fe^{2+} - O^{2-} - Fe^{3+}$ bond the electron of the Fe^{2+}-ion hops via the oxygen ion to the Fe^{3+} ion. To be more precise actually two hopping processes take place simultaneously from the Fe^{2+}-ion to O^{2-} and from O^{2-} to Fe^{3+}, hence the name double exchange.[55] This exchange favours ferromagnetic alignment as the electron moves from an e_g orbital to another e_g orbital which is only energetically favourable if the t_{2g} electrons have the same spin direction. This kind of exchange is found in the magnetite structure between edge sharing octahedra similar to the configuration depicted in fig. 2.2b) where one of the octahedral centres is occupied by an Fe^{2+} ion.[53] For total spin $S = 1/2$ this interaction can also be described by an Heisenberg type Hamiltonian (eq. 2.37) with an appropriate effective exchange constant, for larger S this is only an approximation.[55]

2.1.2.5 Magnetic anisotropy

Magnetic anisotropy describes the tendency of magnetic moments to align preferentially along a certain crystallographic direction due to anisotropic crystal fields or the sample shape even without the presence of an external field. It is a direct consequence of the coupling of electron orbits to the crystal field together with the spin-orbit interaction (magnetocrystalline anisotropy) or of the dipole-dipole interactions between individual moments (shape anisotropy), respectively.[55,62]

Magnetocrystalline anisotropy is intrinsic to the material under consideration and related to the crystal symmetry. For crystals with cubic symmetry the easy axis are oriented along the cube edges and the hard axis along the cube diagonals if the anisotropy constant is positive and vice versa for a negative constant (fig. 2.3).[53,63] The anisotropy constants are temperature dependent such that for magnetite the sign of the first order constant switches from negative at room temperature to positive at low temperatures.[64] The anisotropy energy is given by

$$E_{\text{anis}} = K_0 + K_1 V(\alpha^2\beta^2 + \alpha^2\gamma^2 + \beta^2\gamma^2) + K_2 V(\alpha^2\beta^2\gamma^2) + \dots, \tag{2.38}$$

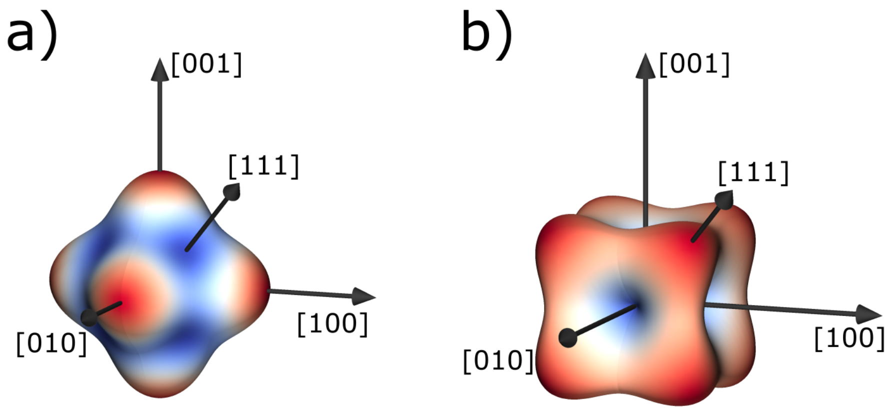


Fig. 2.3 Cubic magnetocrystalline energy surfaces according to eq. 2.38. a) $K_0 = 0.5$ and $K_1 = -1.0$ leading to easy axis along the cube diagonals. b) $K_0 = 1.5$ and $K_2 = 2.5$ giving easy axis along the cube edges.

where $\alpha = M_x/M$, $\beta = M_y/M$, and $\gamma = M_z/M$ are the direction cosines of the magnetization and V is the sample volume.[54,65] The constant K_0 is usually omitted. For iron oxide nanoparticles a common approximation is the use of only the second term of eq. 2.38. An equivalent expression for the anisotropy energy can be derived from the identity relation of the direction cosines

$$\alpha^2 + \beta^2 + \gamma^2 = 1 \tag{2.39}$$

allowing for

$$\alpha^2\beta^2 + \alpha^2\gamma^2 + \beta^2\gamma^2 = -\frac{1}{2}(\alpha^4 + \beta^4 + \gamma^4) \tag{2.40}$$

thus resulting in

$$E_{anis} = -\frac{K_1}{2}(\alpha^4 + \beta^4 + \gamma^4). \tag{2.41}$$

Useful for atomistic simulations is the introduction of an anisotropy constant k_c for each atom obtained by dividing the bulk anisotropy in $\mathrm{J/m^3}$ by the number of atoms per m^3.[47,60,66] Further replacing the direction cosines by the vector components of a semi-classical spin vector gives[65–67]

$$E_{\mathrm{anis}} = -\frac{k_c}{2}\sum_i (S_x^4 + S_y^4 + S_z^4). \tag{2.42}$$

Shape anisotropy is a result of the finite spatial extension of a magnetic body. The divergence of the magnetization at the surface of the object leads to an opposing field $\boldsymbol{H}$ that acts against the magnetization and is thus termed demagnetizing field. For an ellipsoidal magnet the demagnetizing field can be assumed to be uniform inside the magnetic body and can be described by a tensor N according to

$$\boldsymbol{H}_{\mathrm{demag}} = -N\boldsymbol{M}. \tag{2.43}$$

For a perfect sphere the tensor has only diagonal components of 1/3 such that

$$\boldsymbol{H}_{\mathrm{demag}} = -\frac{\boldsymbol{M}}{3}. \tag{2.44}$$

For a magnet of arbitrary shape the demagnetizing tensor is not as straight forward.[53] Although the shape anisotropy can be approximated with the demagnetizing field its origin are the dipole-dipole interactions between atomic magnetic moments. Thus the microscopic dipole field given in eq. 2.22 provides a more exact description.

2.1.2.6 Magnetically ordered structures

The interaction of atomic magnetic moments via the previously discussed mechanisms lead to different kinds of collective arrangements of the moments and thus different macroscopic properties. Important in the context of this thesis are ferro-, antiferro-, and ferrimagnetism.

In a ferromagnet (FM) neighbouring magnetic moments are generally aligned parallel even without an external field through the exchange mechanisms discussed above. This alignment results in a net magnetic moment of the material. However, due to magnetostatic energy domains may be formed which have different orientations of the magnetization vector leading to a macroscopically reduced or even vanishing net magnetization.[54] For ferromagnets the critical temperature is called Curie temperature T_C, below which the moments align in a preferred orientation that evolves into perfect alignment at 0 K.

In an antiferromagnet (AFM) the neighbouring spins are aligned antiparallel thus resulting in two sub-lattices, where each is ferromagnetically arranged. Similar to the ferromagnets a critical temperature exists where the magnetic arrangement vanishes at zero applied field, the Néel temperature T_N. However, even below this temperature no significant magnetization is measured as the magnetization of the two sub-lattices cancel each other.

Similar to antiferromagnetism in a ferrimagnet (FiM) the lattice is build up of two sub-lattices with opposing magnetizations but in this case with different magnitude thus leading to a net magnetization. This type of magnetic order is observed in the maghemite and magnetite structures, which are discussed in more detail in section 2.6.2. The critical temperature at which this long range order of magnetic moments is destroyed is in this case often again called Curie temperature.

Important in distinguishing the different kinds of magnetic order is the Curie-Weiss law applied to susceptibility data at temperatures above the critical temperature, i.e. in the paramagnetic state. It is given by

$$\chi = \frac{C}{T-\theta}, \tag{2.45}$$

where T is the temperature, C is the Curie constant and θ denotes the Weiss temperature. A paramagnet has a Weiss temperature $\theta = 0$, whereas for ferro- and ferrimagnetic systems it is equal to the Curie temperature. For antiferromagnetic structures θ is usually negative although in principle not equal to $-T_N$.[53]

2.1.2.7 Exchange bias

Exchange bias is a phenomenon usually attributed to the presence of exchange anisotropy at a ferromagnetic(FM) - antiferromagnetic(AFM) interface.[53,68–71] In iron oxide nanoparticles a possibility is the presence of a wüstite phase with antiferromagnetic spin structure next to ferrimagnetic maghemite or magnetite.[69] The effect of exchange bias can be observed if the sample containing such an interface is cooled from above the Néel temperature of the antiferromagnetic component.[71] For the mentioned substances this is achieved by cooling from room temperature. Usually then a shift of the recorded hysteresis loop is observed towards smaller fields, i.e. opposite to the cooling field (if the cooling field was positive). Additionally an increased coercive field is observed for these shifted loops.[69]

The microscopic origin of the exchange bias (EB) effect is still not entirely clear.[72] One approach is that of Malozemoff,[71,73] where the EB effect originates from the formation of domain walls (see next section) in the AFM perpendicular to the FM/AFM boundary due to interface roughness. The domains result in a net magnetization at the interface yielding a shift of the hysteresis loop. However, domains due to interface roughness alone are energetically not favourable, therefore the domain formation due to non-magnetic defects throughout the AFM volume was proposed.[71,72] This mechanism of EB origin was experimentally studied and confirmed by introducing varying amounts of defects in the AFM CoO layer of a Co/CoO bilayer, where a strong dependence of the EB on the amount of defects was found.[72] The exchange bias field is defined as the offset of the hysteresis loop center according to

$$H_{\mathrm{EB}} = \frac{H_{c1} + H_{c2}}{2}, \tag{2.46}$$

where H_{c1} and H_{c2} are the coercive fields which are related to the exchange constant at the interface and the anisotropy.[54]

In some cases exchange bias effects have been observed for iron oxide structures without the presence of a wüstite phase, i.e. an FM/AFM interface. Possible explanations are the formation of a spin glass-like structure or spin disorder at nanoparticle surfaces[74,75], antiphase boundaries[70] and the ferrimagnet/ferrimagnet interface of maghemite and magnetite[76].

2.1.2.8 Domains

Magnetic domains are a consequence of the minimization of magnetostatic energy in a finite magnetic material originating from the dipole field as given in eq. 2.22.[53] As shown in fig. 2.4 the formation of a domain wall reduces the stray

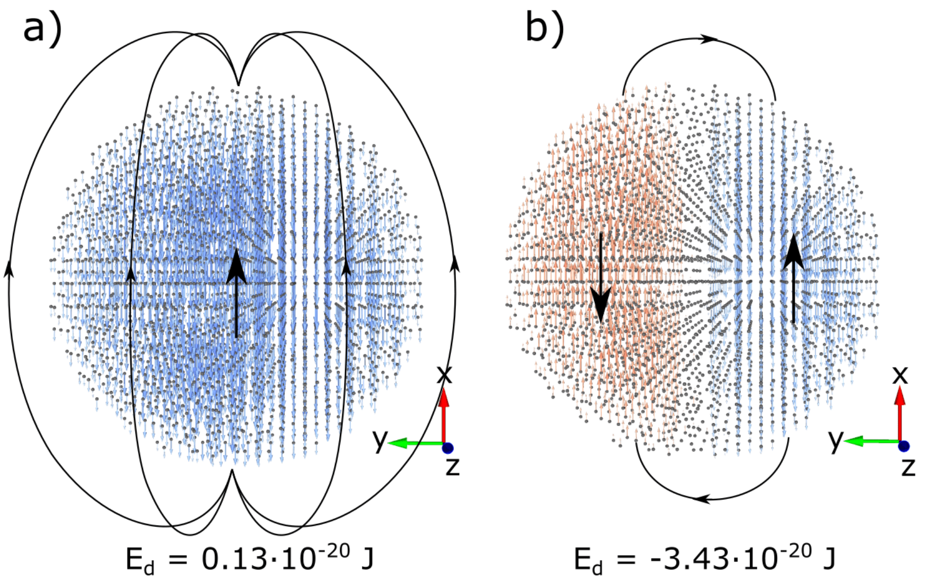


Fig. 2.4 Dipole fields calculated at the lattice points of a spherical maghemite nanoparticle with ideal alignment of the sublattices along x. a) In the single-domain state the dipolar field inside the particle is antiparallel to the particle magnetization orientation that is indicated with the big black arrow in the center. The closure of the field lines leads to larger stray fields outside the particle (black lines) than would be observed for a particle containing a 180° domain wall through the center (b). It should be noted that the plotted lines belong to the magnetic field $\boldsymbol{H}$, while the field lines of the magnetic induction $\boldsymbol{B}$ are continuous. Blue arrows point in $-x$-direction and red ones along x. Below the dipole energy of the system is given, calculated as the sum of the dipole energies at the lattice sites. The depicted particles have a diameter of approximately $9.2\,\mathrm{nm}$.

field outside the particle. The different domains in the material are separated by domain walls, where the direction of magnetization changes by a rotation of the atomic moments. In a Bloch wall, the magnetization rotates about the normal of the domain wall plane, while in thin films the rotation is often parallel to the domain wall plane leading to a Néel wall. The domain wall width depends on the exchange interactions and the magnetocrystalline anisotropy as a rotation of a spin at the domain wall requires to overcome these energies. Strong exchange interactions and small anisotropy constants favour large domain wall widths.[54] Without external field the magnetization within the domains is oriented along the easy axis giving rise to different possible domain wall orientations. E.g. in a material with uniaxial anisotropy 180° domain walls form where the magnetization is aligned along the anisotropy axis in opposite directions on either side of the domain wall. This type of domain wall is shown in fig. 2.4. For cubic magnetocrystalline anisotropy more easy axes are present. Easy axis along the cube edges gives rise to both 90° and 180° domain walls, whereas for easy axis along the cube diagonal 71°, 110°, and 180° walls are possible.[77]

The generation of domains depends on the balance of energy gain due to dipolar energy reduction and the energy cost due to domain wall formation. There exists thus a critical radius for a spherical particle below which the cost of domain wall formation is higher than the dipolar energy that could be saved and the single-domain state becomes energetically more favourable. The consequences of this are discussed in the next section dealing with Nanomagnetism.

2.1.3 Nanomagnetism

The size of the magnetic specimen plays a crucial role for the magnetic properties. In bulk materials the dominating features in ferro- and ferrimagnets are magnetic domains that lead to observable effects like hysteresis behaviour in measurements of the magnetization as a function of the applied magnetic field. As stated previously these domains serve to reduce the total energy of the system by minimizing the magnetostatic energy. Upon reaching a certain size introduction of domain walls may increase the energy as compared to a single-domain particle.[58,78] The magnetostatic energy of the single-domain spherical particle without applied field can be approximated via

$$E_{\text{dip}} = -\frac{\mu_0}{2} \int\limits_V \boldsymbol{M} \cdot \boldsymbol{H}_{\text{demag}} d\boldsymbol{r} = \frac{1}{6}\mu_0 M_s^2 V = \frac{2}{9}\mu_0 M_s^2 \pi R^3, \tag{2.47}$$

where M_s is the volume saturation magnetization and $V = 4/3\pi R^3$ is the sphere volume.[53] Here eq. 2.43 was used for $\boldsymbol{H}_{\text{demag}}$ and the demagnetization tensor was assumed to have only diagonal elements with entries equal to 1/3. The energy gain due to domain formation can be estimated to be half this energy. The competing contribution is the domain wall energy given by

$$E_{\text{dw}} = \beta\sqrt{AK}\pi R^2, \tag{2.48}$$

where the cross section of the sphere was assumed as the domain wall area, A and K are the exchange stiffness and the anisotropy constant, respectively and β is a coefficient related to elastic properties and magnetostriction.[77,79,80] For a 180° domain wall on a (110) plane $\beta = 2.69$.[81] Equating both energies and solving for the particle radius R then gives the critical radius R_c according to

$$R_c \approx 9\beta \frac{\sqrt{AK}}{\mu_0 M_s^2}. \tag{2.49}$$

For magnetite particles this results in a critical particle diameter of 18 to 43 nm depending on the assumed magnetocrystalline anisotropy constant. However, it should be noted that eq. 2.49 only holds for particles with strong anisotropy ($K_u \geq \mu_0 M_s^2/6$). For small anisotropies this equation underestimates the critical radius.[63] For spherical particles composed of γ-Fe_2O_3 an estimate for the critical diameter of 128 nm and 166 nm for Fe_3O_4 is given by Leslie et al.[82] Moskowitz et al.[81] report a diameter of 80(20) nm for magnetite. In Butler et al.[83] a critical diameter of 76 nm is given. Although those are only rough estimates and the critical radii depend on the specific sample this means that most of the considered particles in this thesis theoretically fall well below the limit so that a single-domain state is favourable. An important consequence of the single-domain state is superparamagnetism that is discussed in the following.

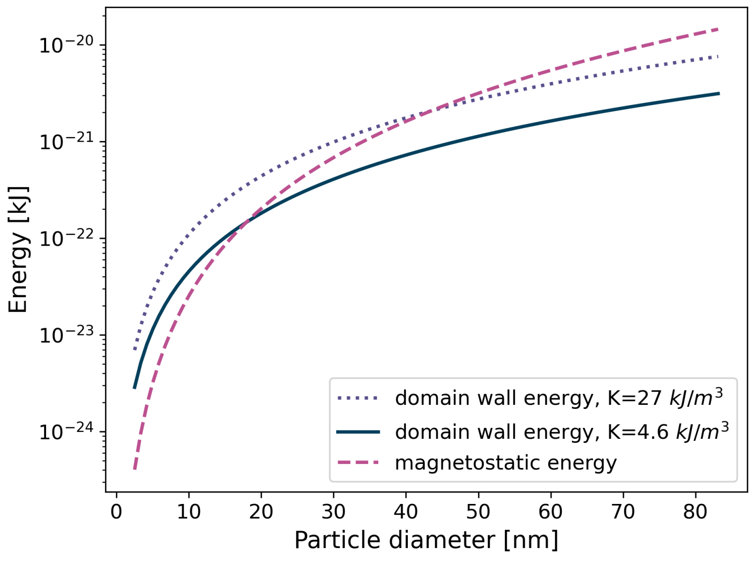


Fig. 2.5 Calculation of the theoretical single-domain diameter. The critical particle diameter below which the single-domain state is energetically more favourable is found by the intersection of the curves for magnetostatic energy and domain wall energy as given in eqs. 2.47 and 2.48 using $\beta = 2.69$, $A = 1 \times 10^{-11}$ J/m, and $M_s = 480\,\mathrm{kA/m^3}$ as well as two estimates for the magnetocrystalline anisotropy constant corresponding to experimental values of the bulk and $15\,\mathrm{nm}$ particles.[84]

2.1.3.1 Superparamagnetism

Superparamagnetism is a property exhibited by single-domain particles that act like paramagnetic atoms although with a much larger magnetic moment in the range of several thousand Bohr magnetons, also called macro- or superspins.[58] In absence of an external magnetic field the magnetization direction of these particles is determined by magnetic anisotropy (see section 2.1.2.5).

Different special cases of superparamagnetism may be distinguished, namely unblocked (isotropic), blocked and interacting superparamagnetism. The unblocked state can be described by the Langevin model, whereas for the blocked state a model was developed by Stoner and Wohlfarth[85].

The defining properties of a superparamagnetic material are a lack of hysteresis above the blocking temperature in a magnetization vs. applied field measurement and a temperature dependence of magnetization vs. field curves in a way that measurements at different temperatures superimpose in a plot of magnetization against H/T.[58] The blocking temperature T_B is defined by the effective anisotropy K of the particle, the particle volume V, as well as the time scale of the measurement τ_m and is given by

$$T_B \approx \frac{KV}{k_B \ln(\tau_m/\tau_0)}, \tag{2.50}$$

where τ_0 is the inverse attempt frequency, a material property that is often taken as $\approx 10^{-12} - 10^{-9}\,\mathrm{s}$.[86] At this temperature the measurement time scale matches the relaxation time scale of the particle. Experimentally this parameter can be determined from the peak in a Zero-Field Cooled (ZFC) magnetization curve, that is obtained by cooling a sample from above the blocking temperature without external field and then recording the magnetization upon heating in a small applied field. In such an experiment the recorded magnetization at first increases due to the alignment of the superspins until at the blocking temperature the thermal fluctuations lead to a decrease of the total recorded magnetization. If during the measurement of the ZFC curve a larger field is applied the blocking temperature shifts to smaller values due to a reduction of the energy barrier needed to overcome. Usually after the ZFC curve also a Field Cooled (FC) curve is recorded by cooling the sample again with the same temperature sweep rate and the same applied field. A splitting of both curves can be observed at the irreversibility point T_irr. For samples with a size distribution generally $T_\mathrm{irr} > T_B$ and the difference depends on the variation in the particle sizes.[86] For the FC curve a plateau is observed for temperatures well below T_B. A decrease of the magnetization in this temperature range indicates the presence of a superspin glass, while an increase is attributed to the presence of paramagnetic clusters or atoms dispersed between the particles. The combination of both effects can lead to a dip in the FC curve for small temperatures.[87]

In some cases[16,82] the presence of an open loop, i.e. hysteresis, at room-temperature (RT) is taken to be a sign of lacking superparamagnetic behaviour. However, for a single-domain particle this is merely a consequence of a blocking temperature larger than RT and thus it should still be considered superparamagnetic.

2.1.3.2 Unblocked superparamagnetism

In this case the thermal energy is much larger than the anisotropy energy of the particle. As mentioned before the magnetic moment of the particle is very large thus allowing the assumption $J \to \infty$ which in turn justifies the usage of the Langevin function (eq. 2.21) to describe the field- and temperature-dependent magnetization. To recall eq. 2.21 is given by

$$L(y) = \coth y - \frac{1}{y}, \tag{2.51}$$

where $y = \mu_p \mu_0 H / k_B T$, with the superspin magnetic moment μ_p.[86] The latter is related to the saturation magnetization in $\mathrm{Am^2/kg_{material}}$ via $M_{sat} = \mu_p / V_t \rho$, where V_t is the total particle volume and ρ the density of the material. In a real sample of nanoparticles a size distribution is present, which has to be considered in fits to experimental data. Accordingly, the magnetization is described by

$$M(B,T) = \int_0^\infty M_{sat} L\left(\frac{\mu_p B}{k_B T}\right) P(v) dv, \tag{2.52}$$

where $P(v)$ is the particle size distribution, as given in eq. 2.98. B is the magnetic field, k_B and T are the Boltzmann constant and the temperature, respectively.[88] It should be noted that this is only valid for negligible interparticle interactions. The change from unblocked to blocked superparamagnetism is not abrupt but a transition region termed anisotropic superparamagnetism can be observed, where deviations from the Langevin behaviour occur.[88]

2.1.3.3 Blocked superparamagnetism - Stoner-Wohlfarth model

Blocked superparamagnetism of nanoparticles is described by the theory of Stoner and Wohlfarth.[85] This model assumes negligible inter-particle interactions and coherent rotation of the magnetization within single particles. Central to this model are the angle θ of the easy axis to the applied field H and the angle ϕ between the magnetization vector of the particle and the applied field (fig. 2.6c). The energy of the system is then described by

$$E = K \sin^2(\theta - \phi) - \mu_0 H M_s \cos(\phi), \tag{2.53}$$

where the first term corresponds to the uniaxial anisotropy energy in polar representation and the second to the Zeeman energy. If no external field is applied the Zeeman term does not contribute to the energy, therefore minimization of the system energy is achieved if the magnetization lies parallel or antiparallel to the anisotropy axis (fig. 2.6e)). Upon application of a field the minimization of the energy is more complex. Energy surfaces can be constructed for differing values of ϕ and H, while fixing the angle between the anisotropy axis and the field (fig. 2.6a) and b)). The paths of minimum energy along these surfaces thus correspond to angles between the magnetization vector and the applied field that minimize the energy. These angles relate to the magnetization via $\cos\phi = M/M_{sat}$. Examples of the resulting hysteresis loops are shown in fig. 2.6d). For a single particle the hysteresis loops can take various shapes, i.e. between a square and a linear slope depending on the angle θ. For a large number of particles an average is observed, as the easy axis directions are randomly distributed.

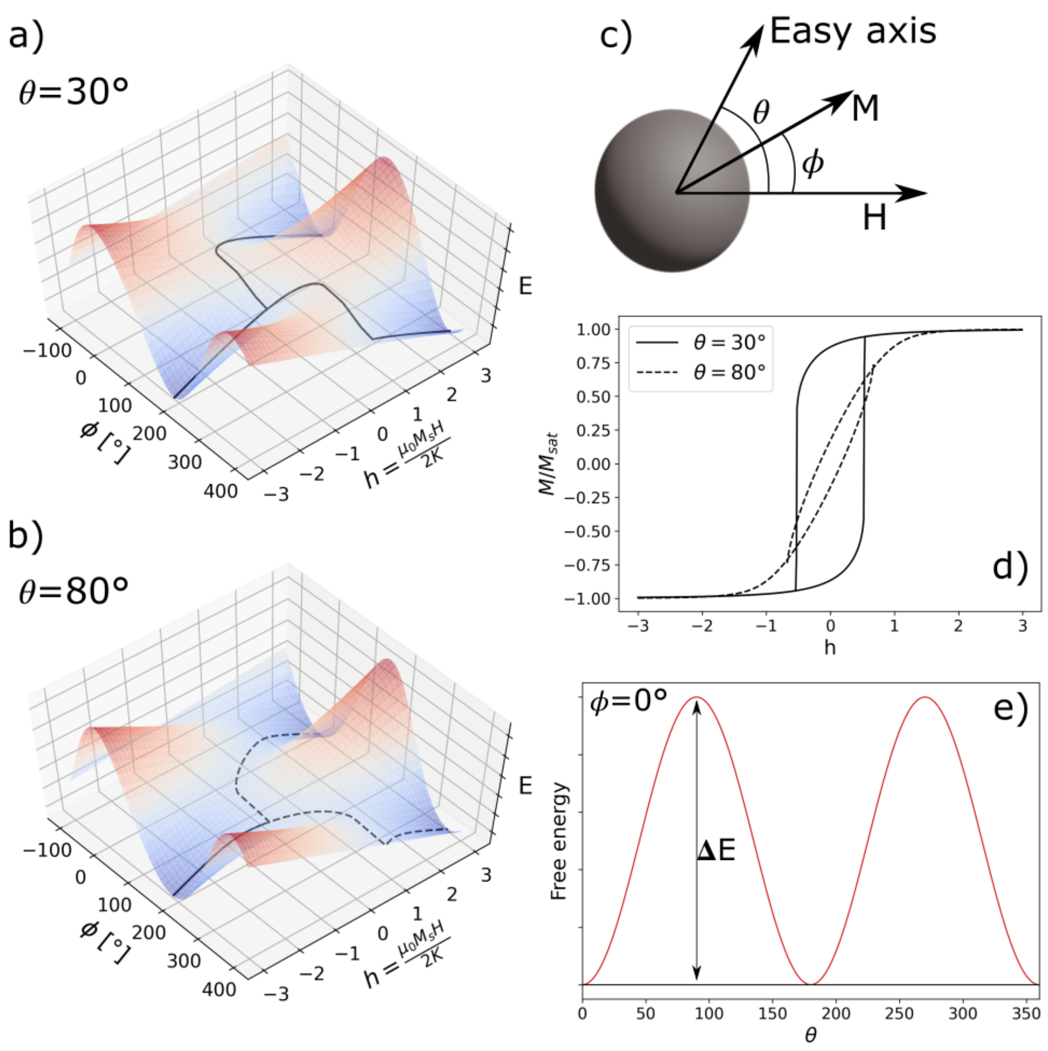


Fig. 2.6 a) and b) show energy surfaces calculated according to eq. 2.53 with the minimum energy paths marked with black lines. c) Angle geometry of the Stoner-Wohlfarth model with ϕ denoting the angle between the magnetization and the applied field and θ the angle between the easy axis and the field. d) Calculated hysteresis loops corresponding to the easy axis angles as used for the energy surfaces. I.e. the solid line corresponds to the energy surface shown in a) and the dashed line to the one depicted in b). e) system energy in dependence of the easy axis orientation without external field ($\phi = 0°$), minimum energy is achieved for $0°$ and $180°$ between easy axis and the magnetization, i.e. parallel or antiparallel alignment.

2.2 Monte Carlo simulations

Computer simulations are well established as a complementary tool to theory and experiment. They provide a method to deal with problems in physics that are very difficult or impossible to solve analytically or where experiments are not possible.[89] Thus simulations can be used to evaluate whether an analytical model describes a physical system well enough. An important advantage is the possibility to calculate properties of a model system exactly without additional approximations apart from the ones used to set up the model.[90] In addition, simulations can serve as a tool to verify approximations used in analytical models.[91] In this thesis Monte Carlo simulations were primarily used to study the influence of antiphase boundaries on the spin structure of iron oxide nanoparticles. Controlling the number and orientation of these defects in real samples is extremely difficult whereas in the simulations model systems can be set up relatively easily to estimate the effects on the macroscopic properties as well as the local spin structure in a defined manner.

2.2.1 General Considerations

The expected value $V(x)$ of some continuous probability distribution $p(x)$ is given by

$$V_p(x) = \int x p(x) dx, \tag{2.54}$$

where the subscript p on $V(x)$ indicates that the random variable x has been drawn from the distribution $p(x)$. The variable x can also be a vector with arbitrary dimension. In the simple sampling Monte Carlo approach the expected value can be approximated by randomly selecting values of x from the distribution $p(x)$ and computing the average according to

$$V_p(x) \approx \frac{1}{N} \sum_{i=1}^{N} x_i, \tag{2.55}$$

thus by increasing N, the number of samples, the resulting value gets closer and closer to the expected value $V_p(x)$. However, there are two drawbacks to this method. First, it may not be possible to sample $p(x)$ directly and secondly, the variance of the obtained expected value might be very large if $p(x)$ has a small distribution and a large fraction of the samples lie in regions that do not significantly contribute to the mean. These problems can be circumvented by using the importance sampling Monte Carlo method. The idea here is that instead of drawing from the distribution $p(x)$ samples are drawn from a different distribution $q(x)$ that provides more samples of x in the region where $p(x)$ is large and thus the contribution of x to the mean is large. The trick is to expand eq. 2.54 by one, i.e.

$$V_p(x) = \int x p(x) \frac{q(x)}{q(x)} dx = \int x \frac{p(x)}{q(x)} q(x) dx. \tag{2.56}$$

Comparing to the equation above now the expected value for samples drawn from $q(x)$ can be approximated by

$$V_p(x) = V_q\left(x \frac{p(x)}{q(x)} \right) \approx \frac{1}{N} \sum_{i=1}^{N} x_i \frac{p(x_i)}{q(x_i)}. \tag{2.57}$$

This approach is particularly useful in statistical physics, where any observable in thermal average is defined in the canonical ensemble

$$\langle A(\boldsymbol{x})\rangle = \frac{1}{Z}\int p(\boldsymbol{x})A(\boldsymbol{x})d\boldsymbol{x}, \tag{2.58}$$

where $A(\boldsymbol{x})$ represents the observable, $p(\boldsymbol{x})$ are the Boltzmann factors according to

$$p(\boldsymbol{x}) = e^{-E(\boldsymbol{x})/k_BT} \tag{2.59}$$

with the total energy E. The partition function Z is given as

$$Z = \int e^{-E(\boldsymbol{x})/k_BT}. \tag{2.60}$$

Thus the expected value of the observable can be approximated in the importance sampling approach as

$$\bar{A}_q(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} A(\boldsymbol{x_i})\frac{p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)}. \tag{2.61}$$

A convenient choice of $q(\boldsymbol{x})$ would be a distribution that is proportional to the Boltzmann factor distribution $p(\boldsymbol{x})$ such that the sum reduces to an arithmetic average of the expected values associated with the selected states $\boldsymbol{x}_i$. A procedure that provides states generated from such a distribution is the Metropolis algorithm, discussed in the following.

2.2.2 Metropolis algorithm

The algorithm developed by Metropolis et al.[92] produces a Markov chain of phase space states thus effectively giving samples for a distribution that is proportional to the Boltzmann distribution. In a Markov chain the selected states are not independent from each other but they are constructed from the previous one in the chain with a transition probability $W(\boldsymbol{x}_i \longrightarrow \boldsymbol{x}_{i+1})$. With an appropriate transition probability it can be ensured that the obtained states approximate the equilibrium distribution for a large number of samples. With the principle of detailed balance according to

$$p(\boldsymbol{x}_i)W(\boldsymbol{x}_i \longrightarrow \boldsymbol{x}_{i'}) = p(\boldsymbol{x}_{i'})W(\boldsymbol{x}_{i'} \longrightarrow \boldsymbol{x}_i), \tag{2.62}$$

where $p(\boldsymbol{x})$ is the Boltzmann distribution, it is made sure that the transition from $\boldsymbol{x}$ to $\boldsymbol{x'}$ and the reverse transition depends on the energy difference between both states.[90] Rearranging gives

$$\frac{W(\boldsymbol{x}_i \longrightarrow \boldsymbol{x}_{i'})}{W(\boldsymbol{x}_{i'} \longrightarrow \boldsymbol{x}_i)} = \frac{p(\boldsymbol{x}_{i'})}{p(\boldsymbol{x}_i)} = \frac{e^{-E(\boldsymbol{x}_{i'})/k_BT}}{e^{-E(\boldsymbol{x}_i)/k_BT}} = \exp\left(-\frac{E(\boldsymbol{x}_{i'}) - E(\boldsymbol{x}_i)}{k_BT}\right). \tag{2.63}$$

Thus one possibility for the transition probability is

$$W(\boldsymbol{x}_i \longrightarrow \boldsymbol{x}_{i'}) = \exp\left(-\delta E/k_B T\right) \tag{2.64}$$

with $\delta E = E(\boldsymbol{x}_{i'}) - E(\boldsymbol{x}_i)$, if δE is positive and $W = 1$ otherwise.[90] The Metropolis algorithm in the context of atomistic Monte Carlo simulations proceeds as follows. First the system is set up, i.e. the crystal structure is generated with e.g. random orientation and random placement of vacancies. After that a spin vector is selected randomly and its classical energy is calculated according to

$$E = E_{exchange} + E_{anisotropy} + E_{Zeeman} + E_{dipole} \tag{2.65}$$

It should be noted here that instead of the Hamiltonian valid for quantum mechanical systems only the classical energy associated with classical spin vectors is calculated. The equations for the energy contributions are given in section 2.1. After the energy is calculated a trial move is performed leading to a new orientation of the spin with a subsequent calculation of the new energy. If the spin move leads to an energy reduction the new orientation is accepted, if the energy is increased the new orientation may still be accepted but with a probability of $\exp\left(-\delta E/k_B T\right)$ (eq. 2.64). This process is repeated a number of times until equilibrium is reached. A Monte Carlo Step (MCS) is defined as the number of trial moves after which every spin in the system has statistically been moved once. Two aspects are important for valid results and reasonable computation times, namely the generation of (pseudo-)random numbers and the generation of trial moves.

2.2.3 Trial moves

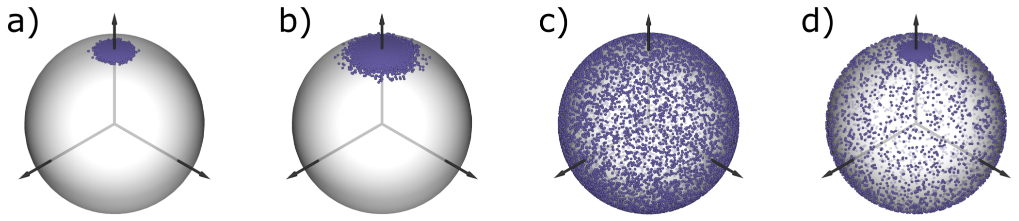


Fig. 2.7 Spin orientations for $10\,000$ spins after one trial move. Each time the starting orientation was along the z-axis. a) Gaussian trial move with $\sigma = 0.14$. b) Gaussian trial move with $\sigma = 0.27$. c) Uniform trial move. d) Hinzke-Nowak trial move by randomly selecting a spin-flip, Gaussian ($\sigma = 0.14$) or uniform trial move.

There are three main ways to perform trial moves, the first and the simplest one being a reflection move, where all vector components are multiplied by -1. In the Heisenberg model used in this work this trial move violates the principle of ergodicity, since not all orientations theoretically possible are achievable. However, it is possible to use this move together with other types that sample the whole phase space.[93] The second possibility is the completely random, i.e. uniform selection of a new spin orientation. While being in accordance with ergodicity with this approach it may take some time until the system reaches equilibrium, as a large number of trial orientations are rejected. The third method is the application of a weight to the possible outcome vectors. It has been recognized by Hinzke and Nowak that a combination of all three trial moves leads to a fast convergence.[66,93] In the implementation of this work for every trial move one of the three possibilities is selected randomly, leading to the distribution of outcome vector orientations depicted in fig. 2.7d). Figs. 2.7a) and b) show the distribution of trial moves of only the third kind, i.e. weighted moves. The weighting is achieved by adding vectors with orientations given by a Gaussian distribution to the initial spin vectors and normalizing the obtained vector. Random numbers with a Gaussian distribution needed for the Gaussian trial move are generated by the fast Ziggurat-algorithm.[94] The opening angle of the cone described by the possible new vector orientations can be adjusted by a parameter σ. For the uniform distribution of trial vectors the Marsaglia method is used, where two random numbers r_1 and r_2 are picked from the uniform distributions $(-1, 1)$ to generate a unit vector. If $S = r_1^2 + r_2^2 < 1$ the Cartesian vector components are calculated according to[95]

$$\begin{aligned}
x &= 2r_1\sqrt{1-S} = 2r_1\sqrt{1-r_1^2-r_2^2} \\
y &= 2r_2\sqrt{1-S} = 2r_2\sqrt{1-r_1^2-r_2^2} \\
z &= 1-2S = 1-2(r_1^2+r_2^2).
\end{aligned} \tag{2.66}$$

2.2.4 Random number generation

Monte Carlo methods rely heavily on good random numbers to generate the sample states chosen from the distribution. The most efficient way to generate these random numbers is by utilization of an algorithm. As such these numbers are in fact not random, but completely deterministic and are thus called pseudo-random numbers. The random numbers are generated in a defined sequence that can be reproduced by using the same seed value. The algorithm used in this thesis was implemented by Ralf Meyer and Fred Hucht on the basis of a process described by Kirkpatrick and Stoll (R250).[96] This random number generator uses a shift-register method, where bitmasks are used to guarantee linear independence between individual bits. The advantage of this approach is the possibility to generate a large number of non-repeating pseudo-random numbers. Non-repeating in the sense that no repeating sequences of random numbers are generated. Random numbers are first generated by some random number generator and stored in a table. From this table new random numbers are obtained via

$$X_n = X_{n-p} \bigoplus X_{n-q}, \tag{2.67}$$

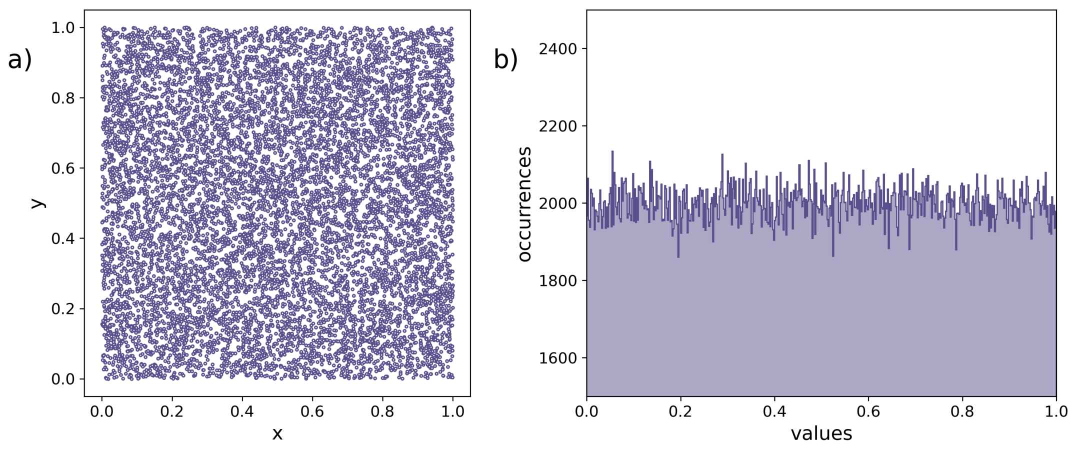


Fig. 2.8 a) So-called parking lot test, where random numbers are taken as coordinate pairs that produce a dot on a $x - y$ plot. No features indicating non-randomness are visible in the $10\,000$ coordinate pairs that were used. b) Uniformity test with the numbers of entries recorded in bins of 0.002 for $1\,000\,000$ numbers.

where $\bigoplus$ denotes a bitwise exclusive-OR operator and $p = 250$ and $q = 103$ leading to a period of about 2^{250}. To test the randomness of the generated sequence of numbers different tests have been developed. A simple visual test is the so-called parking lot test, where sequential random numbers are taken as (x, y) coordinates and plotted as points. Any geometric features such as stripes would indicate non-random behaviour (fig. 2.8a). Another visual test is the uniformity test, where the generated numbers in the interval $[0, 1]$ are binned and the number of entries in each bin is recorded. In a histogram of this data again any obvious features would indicate non-randomness (fig. 2.8b). For both tests no such features are observed for the random number generator used in this work.

2.3 Scattering theory

2.3.1 Basic properties and sources of neutrons and X-rays

One of the fundamental results of quantum physics is the particle-wave duality, e.g. neutrons and X-rays can be treated as both waves and particles depending on the context.[97] The wave properties of neutrons are expressed by the de Broglie equation which relates the mass, m_n, and velocity, v of a neutron to its wavelength, λ according to

$$\lambda = \frac{h}{m_n v}, \tag{2.68}$$

where h is the Planck constant.[98] In the non-relativistic limit the connection to the kinetic energy according to $E = m_n \boldsymbol{v}^2/2$ is then given by

$$\lambda = \frac{h}{\sqrt{2 m_n E}}. \tag{2.69}$$

Thus, in general a large kinetic energy associated with a high temperature via $E = k_B T$, where k_B is the Boltzmann constant, relate to a short wavelength. For neutrons with a wavelength in the range of e.g. nanoparticle sizes (several Å) cold neutrons are needed with temperatures around 100 K. All neutron scattering experiments for this thesis were performed at the FRM II research reactor in Garching, where neutrons are produced in a nuclear fission reaction of ^{235}U isotopes. Other neutron sources that are not further discussed here include spallation sources and nuclear fusion reactions. To achieve the desired wavelengths of a few Å the neutrons generated by the fission reactor have to be moderated, that is their kinetic energy and thus their temperature is lowered by inelastic scattering in a moderator material. At the FRM II liquid deuterium is used as the moderator for cold neutrons. An additional very useful property of neutrons is their magnetic moment that allows the investigation of magnetic properties of materials.[99]

X-rays are electromagnetic radiation where the relation between wavelength λ and the energy is given as

$$E = \frac{hc}{\lambda}, \tag{2.70}$$

where c is the speed of light.[100] Their wavelengths lie between those of γ-rays and ultraviolet light and are therefore again in a useful range to study the properties of materials.[101] X-rays interact with the atomic electrons and are therefore more sensitive to heavier elements. They can be generated in X-ray tubes or in a synchrotron facility. In a X-ray tube electrons are emitted from a cathode, are accelerated in the electric field and hit an anode. The electrons are decelerated by the electric fields of the target metal anions which leads to the *Bremsstrahlung* with a continuous energy distribution. Additionally, characteristic X-rays are emitted with an energy that is specific to the anode material as it originates from electron transitions in the ionized anode atoms.[102] In synchrotron sources the X-ray generation is based on the acceleration of charged particles. The constant acceleration is achieved by forcing the charged particle beam into a circular path by the use of magnets. Special magnet arrangements such as wigglers and undulators are used in modern synchrotrons. By passing through a wiggler or an undulator consisting of alternating magnetic fields the electron beam is forced on an oscillating path. The difference between both lies in the period of the alternating magnetic field in cm, λ_u, and the field strength in tesla, B, expressed in

$$K = \frac{eB\lambda_u}{2\pi m_0 c}, \tag{2.71}$$

where e is the elementary charge, m_0 is the mass of the electron and c is the speed of light. For $K > 1$ the device is called wiggler and otherwise undulator. While with a wiggler a broad X-ray spectrum is generated the undulator produces a sharp X-ray beam spectrum and a higher brilliance.[100] The wavelength of the generated X-ray beam can either be selected by a monochromator from the continuous spectrum of the wiggler or by adjusting the distance between the magnets in the undulator.[103]

2.3.2 Scattering cross sections

In a rough approximation the scattering of neutrons and photons from isolated scattering centres can be described as a deflection of the particles from their original path due to an interaction potential. In the following only elastic scattering is considered, where the kinetic energy of the incident particle is conserved. An intuitively accessible example of scattering of a projectile from a target is the classical hard sphere scattering.[104] The potential in that case is infinite inside the sphere and equal to 0 outside, meaning the incoming projectile cannot pass through the sphere and is scattered, but if the sphere is missed no deflection occurs. The region where scattering is possible is thus the cross-sectional area of the sphere

$$\sigma = \pi R^2, \tag{2.72}$$

also known as the *scattering cross section.*[104] This simple connection between the scattering cross section and the cross section of the target object is generally not given[99,105] and the scattering cross section in the case of X-rays and neutrons should be thought of as a measurement of interaction strength between projectile and target depending on the trajectory.

In other words a larger scattering cross section relates to a larger probability of scattering. In an experiment where an incident beam of particles hits a target, some fraction of the incoming particles is scattered into a solid angle $d\Omega$ (fig. 2.9). The proportionality constant between the number of incoming particles and the number of scattered particles is called the *differential scattering cross section* and is denoted by $d\sigma/d\Omega$. It describes the number of particles scattered into the solid angle $d\Omega$ after passing through an infinitesimal area $d\sigma$ of the scattering cross section. Integration of the differential scattering cross section over all solid angles thus returns the scattering cross section

$$\sigma = \int \frac{d\sigma}{d\Omega} d\Omega, \tag{2.73}$$

where $d\sigma/d\Omega$ should be regarded as a symbol rather than a mathematical expression.[99,105,106] The differential cross section is experimentally accessible through the counting of the number of particles incident on a detector.[99,106] Therefore, the main task in structure analysis is the reconstruction of this quantity with an adequate fitting model.

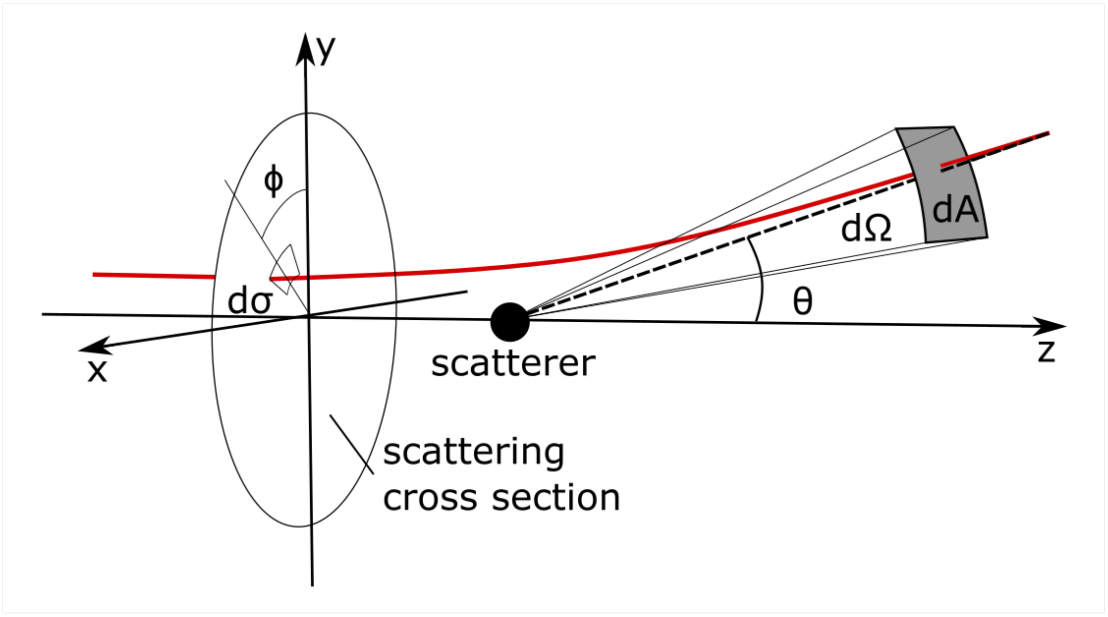


Fig. 2.9 The particle trajectory is indicated with the solid red line passing through the infinitesimal element $d\sigma$ of the scattering cross section. Due to the interaction with the potential located at the scattering center the particle is deflected into the infinitesimal solid angle $d\Omega$ and passes through the surface element dA. The azimuthal angle is denoted with θ, the polar angle is ϕ.

The particle picture of scattering is intuitive, however interference phenomena can only be explained using the wave properties of X-rays and neutrons.[97,99] Therefore a description of the differential cross section, i.e. the measurable quantity, based on the scattering of waves is needed. The basic tool is the quantum mechanical wave function Ψ, defined by a wavelength λ through the wave vector $\mathbf{k}$ pointing in the direction of propagation with length $2\pi/\lambda$.

The asymptotic solution to the time independent Schrödinger equation for a wave scattered from a potential is a superposition of an incoming plane wave

$$\Psi_{in} = \Psi_0 e^{(i\boldsymbol{k}\cdot\boldsymbol{r})}, \tag{2.74}$$

with the amplitude Ψ_0 and a scattered spherical wave with scattering amplitude $A(\theta, \phi)$

$$\Psi_{sc} = \Psi_0 \frac{A(\theta, \phi)}{r} e^{(ikr)}, \tag{2.75}$$

where r is the distance from the origin and accordingly the intensity per unit area, $|\Psi(\boldsymbol{r})|^2$, decreases proportional to the square of the distance.[106,107] The superposition of both waves is then

$$\Psi = \Psi_0 \left[e^{(i\boldsymbol{k}\cdot\boldsymbol{r})} + A(\theta, \phi) \frac{e^{(ikr)}}{r} \right], \tag{2.76}$$

where $\Psi_0 = 1/\sqrt{V}$ is the normalization factor of the wave function with the volume V, ensuring that the volume integral of the wave function is equal to 1, i.e. the particle is present somewhere in the volume.[97,105] The scattering amplitude $A(\theta, \phi)$ in the spherical wave contribution corresponds to the interaction strength between the incoming wave and the target. As it has dimensions of length it is known as the *scattering length.*

X-rays interact with the electron cloud around the atom, thus $A(\theta, \phi)$ is generally dependent on the scattering angle θ but isotropic with respect to ϕ and is called *atomic form factor* usually denoted by $f(\theta)$. For non-magnetic scattering of neutrons the interaction takes place at the atomic nuclei, which can be assumed as points in space with constant scattering length. In this case the usual symbol is b and is simply called *nuclear scattering length.* In magnetic scattering neutrons interact with unpaired electrons of the atoms, where the scattering length depends on the scattering angle similar to the atomic form factor of X-rays and is denoted by $F_d(\theta)$, called the *magnetic form factor.*[99] The differential cross section can be described as the ratio of the current densities of the scattered beam through a solid angle and the incoming beam. Using eq. 2.74 and eq. 2.75 to calculate the incoming (J_{in}) and the scattered beam (J_{sc}) current densities, respectively, it can be shown that the absolute square of the scattering amplitude constitutes the differential cross section[108]

$$\frac{d\sigma}{d\Omega} = \frac{J_{sc}}{J_{in}} r^2 = |A(\theta, \phi)|^2. \tag{2.77}$$

Finally, normalizing the differential cross section by the sample volume yields the scattering intensity

$$I(\theta, \phi) = \frac{1}{V} \frac{d\sigma}{d\Omega} = \frac{1}{V} |A(\theta, \phi)|^2. \tag{2.78}$$

The scattering intensity has units of $\mathrm{cm}^{-1}\mathrm{sr}^{-1}$, where the unit of the solid angle is usually omitted.[108]

It should be noted that in reality the measured intensity is usually slightly different due to factors such as the pixel size, the detector efficiency etc. In most cases $I(\theta, \phi)$ is isotropic with respect to ϕ, however in magnetic scattering of neutrons this is not the case (see section 2.3.4.6). To simplify the notation in the following $A(Q)$, with $Q = 4\pi \sin(\theta/2)/\lambda$ is used where the dependence on ϕ is not needed and $A(\boldsymbol{Q})$ otherwise.

To find expressions for the scattering amplitude it is useful to describe it in the first Born approximation, i.e. in a single scattering approximation, as the interaction of a plane wave with a potential according to

$$A(\boldsymbol{Q}) = \int\limits_V \rho(\boldsymbol{r}) e^{i\boldsymbol{Q}\boldsymbol{r}} dV, \tag{2.79}$$

where $\boldsymbol{Q} = \boldsymbol{k} - \boldsymbol{k}_0$ is the scattering vector, with the wave vectors of the incoming and outgoing wave k_0 and k, respectively. The spatial vector is denoted with $\boldsymbol{r}$.[106] The potential $\rho(\boldsymbol{r})$ can take different forms depending on the length scale under consideration. A point-like potential is usually used to describe the interaction of waves with free electrons or atomic nuclei. For neutron scattering from atomic nuclei the potential is the *Fermi pseudopotential.*[99] A spherically symmetric potential is used for the charge distribution around atoms to derive atomic form factors for X-ray scattering.[109] A periodic potential represents the arrangement of atoms in a crystal[101,106] and an arbitrary shaped potential could be attributed to a larger particle.[106] The integral in eq. 2.79 is the Fourier transform of the respective potential field, therefore, by reconstructing the differential scattering cross section information on the structure of the sample can be obtained on the length scale of the incident wavelength.

2.3.3 Debye scattering equation

The Debye Scattering Equation (DSE), established already in 1915,[110] represents a fundamental result of scattering theory of isotropic samples that shows nicely the connection between the different areas of scattering that are discussed in the following sections. The derivation is based on very few assumptions and the result can be applied to scattering from ordered crystal structures, disordered or amorphous structures, to derive the expressions used in Pair Distribution Function (PDF) analysis and even for small-angle scattering.

The starting point for the derivation of the DSE is the amplitude of a scattered wave from a structure, i.e. the sum of the scattered waves from the individual scattering sources. As mentioned, this structure is not required to be ordered and the scattering sources could be atoms in a crystal structure, molecules, volume elements in a particle or entire particles. The differential cross section is the absolute square of the amplitude, i.e. the product of $A(\boldsymbol{Q})$ with its complex conjugate $A(\boldsymbol{Q})^*$, giving

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = |A(\boldsymbol{Q})|^2 = A(\boldsymbol{Q})A^*(\boldsymbol{Q}) = \int\limits_{V_m}\int\limits_{V_n} \rho(\boldsymbol{r}_m)\rho(\boldsymbol{r}_n) e^{i\boldsymbol{Q}\boldsymbol{r}_{mn}} dV_n dV_m, \tag{2.80}$$

where $\boldsymbol{r}_{mn} = \boldsymbol{r}_m - \boldsymbol{r}_n$ is the distance vector between two scattering sources n and m (fig. 2.10). The scattered intensity only depends on the relative positions of these

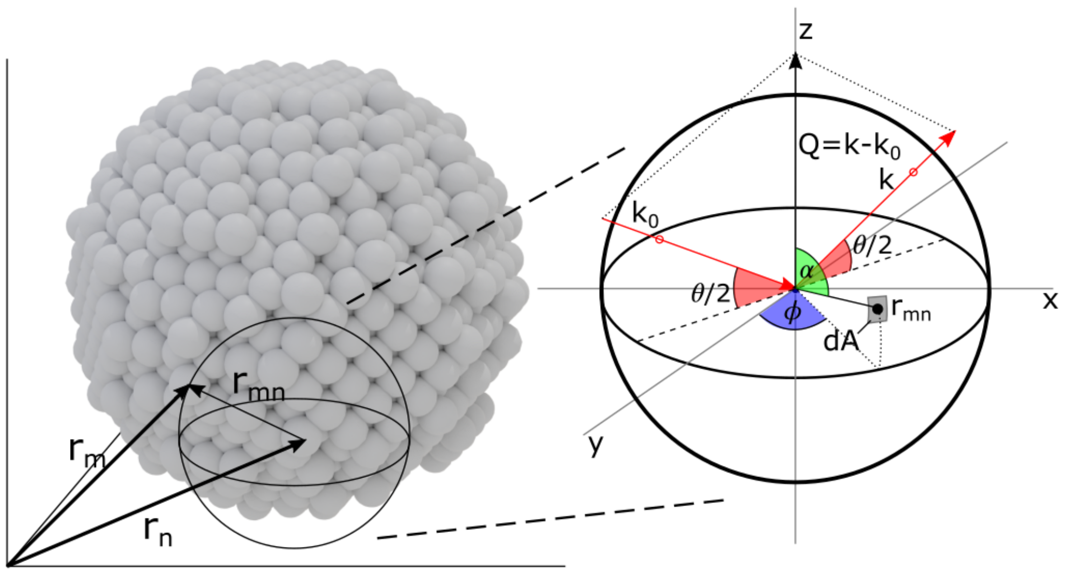


Fig. 2.10 Sketch of the real space and vector geometry used for the Debye scattering equation. The center of the sphere is the atom with position vector r_n, the radius is the distance between r_n and r_m. The angle α lies between the scattering vector Q and the distance vector. The scattering angle is designated with θ and the azimuthal angle is ϕ. The infinitesimal surface element dA is shown as a grey rectangle.

sources and not the absolute locations. The next assumption is that the structure under consideration can take any orientation in space with equal probability, which would for example be the case for a powder sample of crystallites or particles dispersed in a solvent. For each pair of scatterers in the integral of eq. 2.80 one of the scattering sources can be placed in the origin, thus for any orientation of the distance vector in space the endpoint lies on the surface of a sphere with radius r_{mn}, the distance between the two scatterers. The angle α is formed between the scattering vector $\boldsymbol{Q}$ and the distance vector. Therefore the dot product of the vectors in eq. 2.80 can be replaced by $Qr_{mn}\cos\alpha$ and the spherical average for all angles α, i.e. the surface integral, can be performed. With the infinitesimal surface element $dA = rd\alpha \cdot r\sin\alpha d\phi = r^2 \sin\alpha d\phi d\alpha$ the surface average of the exponential term is[111]

$$\langle e^{(iQr_{mn}\cos\alpha)}\rangle = \frac{1}{4\pi r_{mn}^2}\int_0^{\pi}\int_0^{2\pi} e^{(iQr_{mn}\cos\alpha)} r_{mn}^2 \sin(\alpha)d\phi d\alpha. \tag{2.81}$$

The integral over the azimuthal angle ϕ gives a factor of 2π that can be pulled out of the integral together with r_{mn}^2, leaving only a factor 1/2 in front. The integral with respect to α is

$$\int_0^\pi e^{(iQr_{mn}\cos\alpha)}\sin\alpha d\alpha = \left[\frac{e^{(iQr_{mn}\cos\alpha)}}{iQr_{mn}}\right]_\pi^0 = \frac{e^{(iQr_{mn})} - e^{(-iQr_{mn})}}{iQr_{mn}}, \tag{2.82}$$

where the numerator on the right can be written as $2i\sin(Qr_{mn})$ thus leaving

$$\langle e^{i\boldsymbol{Q}\boldsymbol{r}_{mn}}\rangle = \frac{\sin(Qr_{mn})}{Qr_{mn}}. \tag{2.83}$$

This general result is also important for derivation of the form factors of isotropic bodies (section 2.3.4.8). Inserting this into eq. 2.80 gives

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = |A(\boldsymbol{Q})|^2 = \int_{V_m}\int_{V_n} \rho(\boldsymbol{r}_m)\rho(\boldsymbol{r}_n)\frac{\sin(Qr_{mn})}{Qr_{mn}}d\boldsymbol{r}_n d\boldsymbol{r}_m \tag{2.84}$$

If the scattering centres are taken to be atoms at positions $\boldsymbol{r}_i$ the integrals can be replaced by sums and the scattering potentials by the atomic form factors f_n. Thus the differential cross section is the Debye equation in the original form

$$\frac{d\sigma}{d\Omega}(Q) = \sum_m\sum_n f_m f_n \frac{\sin(Qr_{mn})}{Qr_{mn}} = N\langle f^2\rangle + \sum_{m\neq n} f_m f_n \frac{\sin(Qr_{mn})}{Qr_{mn}}, \tag{2.85}$$

where angle brackets denote the compositional average and N is the number of atoms in the structure. Note that the scattered intensity is isotropic with respect to the azimuthal angle ϕ (see fig. 2.9) and thus only depends on Q. The first contribution in the equation on the right corresponds to the so-called self-scattering from individual atoms, which produces a continuous background. If instead the scatterers are particles in a small-angle scattering experiment the scattering potentials correspond to the particle form factors and the first term is the scattering from an individual particle, while the interference term on the right corresponds to particle interactions and ultimately can be shown to relate to the structure factor.

If applied to a crystalline particle the calculated pattern is similar to that of a powder sample due to the spherical average used during the derivation (fig. 2.11). However, only one particle is used to calculate the entire pattern, which means that size distribution effects are not considered. Another drawback is the computational burden imposed by the sum over all atomic pairs. The advantage of this approach is that all features of the calculated pattern are directly related to the input structure without further assumptions and possible diffuse scattering arising from disorder is included in the calculated pattern. As such the use of the Debye scattering equation is a total scattering approach similar to PDF analysis.[111,112] The particle shape and finite size are also implicitly considered, leading to realistic

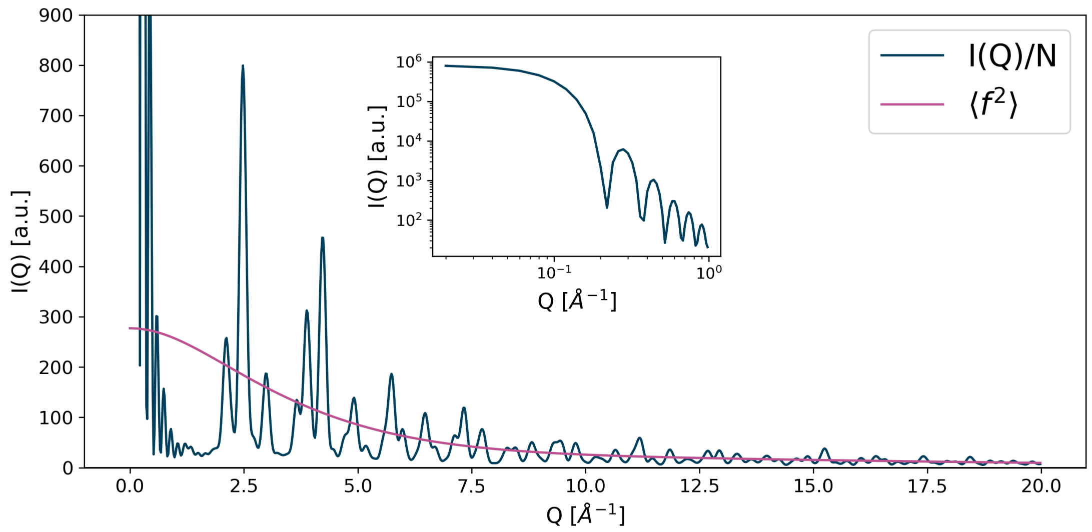


Fig. 2.11 Example of a powder diffraction pattern calculated with the Debye scattering equation (eq. 2.85) for a spherical Fe_3O_4 particle with diameter of $4.2\,\mathrm{nm}$ (blue line). The Bragg peaks are clearly visible as well as the broadening of peaks originating from the particle size. The inset shows the log-log plot of the small-Q region where the typical oscillations expected for a spherical particle are visible (see section 2.3.4). The self-scattering contribution, $\langle f^2 \rangle$, is also drawn (red line). Here f is the atomic form factor not to be confused with the particle form factor.

peak broadening that would need to be phenomenologically included in traditional approaches to calculation of powder patterns.

In the low-Q region the small-angle scattering contribution is observed. As Q gets smaller the atomic details are not resolved any more and the sum in eq. 2.85 can be replaced by an integral over volume elements, which means that for small Q this term is equal to the self-scattering contribution of a single particle with a scattering length density distribution. This is explored in more detail in the following section.

2.3.4 Small-angle scattering

Small-Angle Scattering (SAS) is the technique that allows investigation of the Q-range from approximately 1×10^{-3} to $1\,\mathrm{Å}^{-1}$. In this thesis it is used to obtain exact information on the particle size, the size distribution, the organic shell thickness, interparticle interactions as well as the magnetization distribution within a single particle. In the following, first the concept of scattering length densities is discussed, then general theoretical considerations regarding the scattering from isolated and dispersed particles are explored. The influence of polydispersity and instrumental resolution on the scattering intensities are shown. The theoretical background of SAS with polarized neutrons, which allows investigation of the magnetization distribution, is given. Moreover, contrast variation is discussed that can be used to determine the particle composition. And finally the analytical expressions for the particle form and structure factors are presented.

2.3.4.1 Scattering length densities

For small-angle scattering from particles the scattered wave results from a linear superposition of waves scattered from the individual scatterers, i.e. atomic nuclei, atomic electrons or spins in the structure. However, the atomic details can not be resolved in the small-angle regime as the probed length scales are much larger. Using the Bragg equation[113] according to

$$n\lambda = 2d \sin \frac{\theta}{2}, \tag{2.86}$$

where n is the diffraction order, λ is the wavelength, d is the distance between scatterers and θ is the scattering angle and the magnitude of the scattering vector $Q = 4\pi/\lambda \sin \theta/2$ the following relation can be obtained

$$Q = \frac{2\pi}{d}. \tag{2.87}$$

Thus both large wavelengths and small angles result in information on larger length scales.[114]

Therefore, in the small-angle scattering regime the discrete scattering from individual atoms can be approximated by a continuous distribution of scattering lengths, the *Scattering Length Density* (SLD), which is defined as

$$\rho = \frac{1}{V_m} \sum_{j=1}^{N} f_j, \tag{2.88}$$

where V_m is the molecular volume obtained from the molecular weight M and the bulk density of the material ρ according to

$$V_m = \frac{M}{\rho N_a}, \tag{2.89}$$

with the Avogadro constant N_a. The sum in eq. 2.88 is over the atoms present in the molecular volume and f_j is the form factor or scattering length associated with atom j. For X-rays the atomic form factor generally depends on the scattering angle, however for small angles the approximation $f_j(Q)$ Z_j can be made, where Z_j is the electron number of the atom j.[115] To obtain units of m^{-2} the scattering length density has to be multiplied by the classical electron radius $r_e = 2.81 \times 10^{-15}$ m. Since the neutron nuclear scattering lengths are independent of the scattering angle the respective values can be used directly. For magnetic neutron scattering lengths a similar small-angle approximation can be made as for X-rays and the magnetic scattering lengths are assumed to be constants. The scattering length densities of X-rays and neutrons are generally quite different

Tab. 2.1 Calculated scattering length densities for selected compounds according to eq. 2.88 with tabulated values for the nuclear and X-ray (at $9299\,\mathrm{eV}$) scattering lengths. Values are given in $1 \times 10^{-6}\,\mathrm{\AA}^{-2}$.

Substance	Fe_3O_4	γ-Fe_2O_3	Oleic acid	toluene	D-toluene
X-ray SLD	41.457	39.054	8.514	7.982	7.982
neutron SLD	6.934	6.655	0.078	0.939	5.662

thus both methods yield complementary information on the studied sample. In tab. 2.1 SLDs are given for typical iron oxide materials and solvents for both types of radiation.

2.3.4.2 Small-angle scattering from a particle

Recalling eq. 2.79 the scattering amplitude is given as the Fourier transform of the scattering potential

$$A(\boldsymbol{Q}) = \int\limits_V \rho(\boldsymbol{r}) e^{i\boldsymbol{Q}\boldsymbol{r}} dV. \tag{2.90}$$

For a single particle this potential can be identified as the scattering length density. With that $A(\boldsymbol{Q})$ is called particle form factor amplitude denoted by $F(\boldsymbol{Q})$, where the particle form factor $P(\boldsymbol{Q})$ is the absolute square of this quantity.

For particles in a solvent the form factor amplitude is scaled by the difference between the solvent and the particle SLD. This quantity is called *contrast* and is defined as $\Delta\rho = \langle\rho_{particle}\rangle - \rho_{solvent}$, where the angle brackets denote the volume average. Since the observed intensity is proportional to the squared amplitude only the absolute value of the contrast matters, a concept described by Babinet's principle.[106] For scattering from a spherical particle in a solvent the scattering amplitude is the sum of the scattering amplitudes from the particle and the solvent according to

$$\begin{aligned}
F(\boldsymbol{Q}) &= \int\limits_{V_p} \rho_p e^{i\boldsymbol{Q}\boldsymbol{r}} dV + \int\limits_{V-V_p} \rho_{solv.} e^{i\boldsymbol{Q}\boldsymbol{r}} dV \\
&= \int\limits_{V_p} \rho_p e^{i\boldsymbol{Q}\boldsymbol{r}} dV + \int\limits_{V} \rho_{solv.} e^{i\boldsymbol{Q}\boldsymbol{r}} dV - \int\limits_{V_p} \rho_{solv.} e^{i\boldsymbol{Q}\boldsymbol{r}} dV \\
&= \int\limits_{V_p} (\rho_p - \rho_{solv.}) e^{i\boldsymbol{Q}\boldsymbol{r}} dV + \int\limits_{V} \rho_{solv.} e^{i\boldsymbol{Q}\boldsymbol{r}} dV \\
&\approx \int\limits_{V_p} \Delta\rho e^{i\boldsymbol{Q}\boldsymbol{r}} dV,
\end{aligned} \tag{2.91}$$

where V_p and V are the particle volume and the total volume respectively. This is illustrated in fig. 2.12. The last term in the third line corresponds to a scattering volume that is typically large compared to the investigated length scales (i.e. in the range of the sample volume) and thus leads only to scattering at very low angles that are experimentally not observable. Therefore, this term can be ignored and the scattering amplitude depends only on the contrast.[115] This dependence of the scattering intensity on the contrast is exploited in the experimental technique of contrast variation (see section 2.3.4.7). As can be seen from the tabulated neutron SLDs in tab. 2.1 the contrast between solvent and particle changes drastically by a simple deuteration of the solvent.

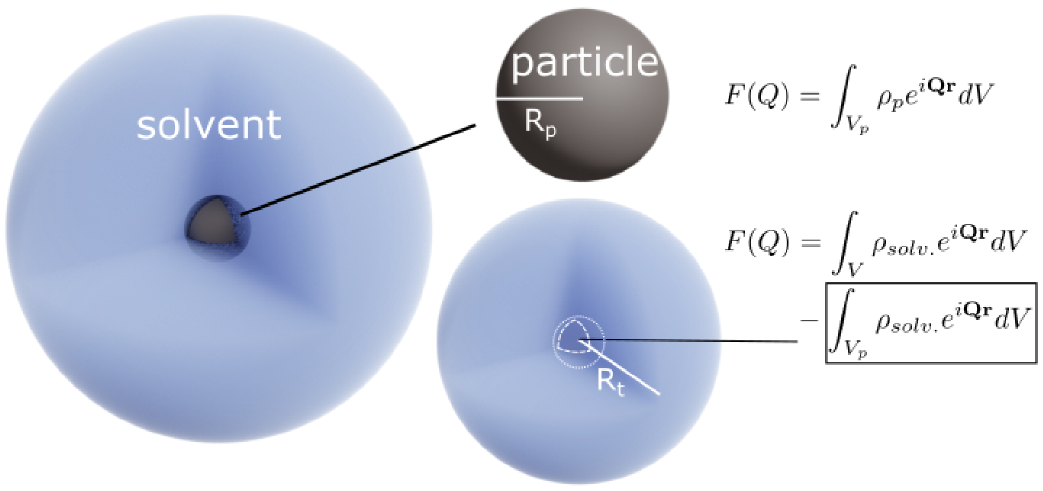

solvent
particle
R_p
$F(Q) = \int_{V_p} \rho_p e^{i\mathbf{Qr}} dV$
$F(Q) = \int_V \rho_{solv.} e^{i\mathbf{Qr}} dV$
$- \boxed{\int_{V_p} \rho_{solv.} e^{i\mathbf{Qr}} dV}$
R_t

Fig. 2.12 Illustration of the concept of contrast and the resulting form factor amplitudes of a particle in a solvent (eq. 2.91).

2.3.4.3 Scattering from dispersed particles

Considering a sample of particles dispersed in a solvent the scattering potential of the sample can be described by an isotropic distribution of particle potentials at positions $\boldsymbol{r}_n$ within the sample. In this case the scattering amplitude of the sample is obtained by replacing the integral of eq. 2.79 by a sum over the particle scattering amplitudes in the sample, giving

$$A(\boldsymbol{Q})_{sample} = \sum_n F(\boldsymbol{Q})_n e^{i\boldsymbol{Q}\boldsymbol{r}_n}. \tag{2.92}$$

The differential cross section is then

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = |A(\boldsymbol{Q})_{sample}|^2 = \sum_n \sum_m F(\boldsymbol{Q})_n F(\boldsymbol{Q})_m e^{i\boldsymbol{Q}\boldsymbol{r}_{mn}}, \tag{2.93}$$

where $\boldsymbol{r}_{mn}$ is the vector between particle m and n. This can be rewritten in a similar form to the Debye scattering equation[116] (eq. 2.85)

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = \sum_{n}^{N} |F_n(\boldsymbol{Q})|^2 + \sum_{m \neq n} F_n(\boldsymbol{Q}) F_m(\boldsymbol{Q}) e^{i\boldsymbol{Q}\boldsymbol{r}_{mn}}, \tag{2.94}$$

where N is the number of particles in the sample, the first expression corresponds to the self-scattering from one particle described by the absolute square of the particle amplitude and the second expression relates to the interference of scattered waves originating from different particles, i.e. it corresponds to the spatial arrangement of particles in the sample. For a sufficiently large number of particles, as is usually the case for dispersed particles in a solvent, the differential cross section contain all possible orientations of particles thus the observed intensity is proportional to the orientational average. The first term can then be interpreted as the particle form factor $P(Q)$ times the number of particles and for the second term the expression for the orientational average of the exponential term derived in eq. 2.83 can be substituted. Further simplification of eq. 2.94 is achieved by introduction of the structure factor $S(Q)$. This allows the transformation to[115]

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = NP(\boldsymbol{Q})S(\boldsymbol{Q}), \tag{2.95}$$

with

$$S(\boldsymbol{Q}) = 1 + \frac{\sum\limits_{m \neq n} F_n(\boldsymbol{Q}) F_m(\boldsymbol{Q}) \frac{\sin(Qr_{mn})}{Qr_{mn}}}{\sum_n^N |F_n(\boldsymbol{Q})|^2}. \tag{2.96}$$

For widely separated particles in the sample, i.e. low concentration of particles, the structure factor approaches 1 and can be omitted.[117] If this is not the case expressions have to be found to properly describe the particle interactions and the resulting relative placements in the sample encoded in the particle-particle distances r_{mn}.

2.3.4.4 Polydispersity

Thus far only monodisperse particle systems were considered, if a size distribution is present the particles have different form factors and the collective interactions may also be different. A useful approximation valid for diluted samples is the local monodisperse approximation, where it is assumed that a particle is always surrounded by particles with the same size.[117] With that the size distribution can be introduced by simply performing the integral over the particle sizes according to

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = N \int D(R) P(\boldsymbol{Q}, R) S(\boldsymbol{Q}, R) dR, \tag{2.97}$$

where $D(R)$ is the function describing the size distribution and R is the particle radius. Usually a lognormal size distribution is used, since particle sizes cannot be negative and experimentally observed particle size spread generally obeys this

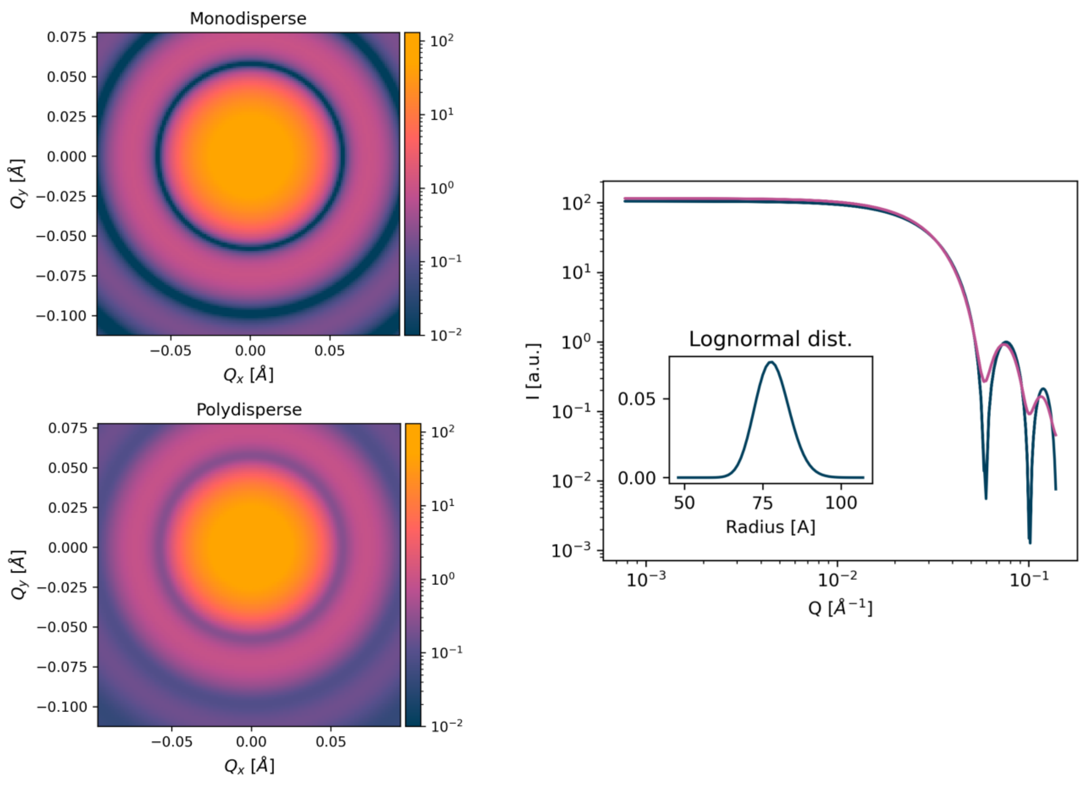


Fig. 2.13 Simulated detector image for scattering of core-shell nanoparticles without (top) and with (bottom) a lognormal size distribution. The plot on the right hand side shows the radially averaged data, where the blue line corresponds to the monodisperse case and the pink line to the polydisperse particles. The inset depicts the probability density function used in the simulation with a parameter $\sigma = 0.07$ and a median particle size of $7.8\,\mathrm{nm}$.

distribution law. The lognormal probability density function used in this thesis is defined as

$$D(R, R_0, \sigma) = \frac{1}{\sqrt{2\pi}\sigma R} \exp\left[-\frac{\ln^2(R/R_0)}{2\sigma^2}\right], \tag{2.98}$$

where σ is the shape parameter defined as the standard deviation on a logarithmic scale, R and R_0 are the particle radius and the median particle radius, respectively, on the natural scale.[118] The mean of the distribution is obtained via $R_m = R_0 \exp(1/2\sigma^2)$. The effect of polydispersity on the scattering intensity, namely the smearing out of the minima, is illustrated in fig. 2.13.

2.3.4.5 Instrumental resolution

Finally resolution effects have to be considered which are introduced as a second integral over the scattering vector $\boldsymbol{Q}$, with the resolution function Q_{res}. With that the differential cross section of a small-angle scattering experiment reads

$$\frac{d\sigma}{d\Omega}(\boldsymbol{Q}) = \iint D(R)P(\boldsymbol{Q}, R)S(\boldsymbol{Q}, R)Q_{res}(\boldsymbol{Q}, \boldsymbol{Q}_0, \sigma_Q)dRd\boldsymbol{Q}. \tag{2.99}$$

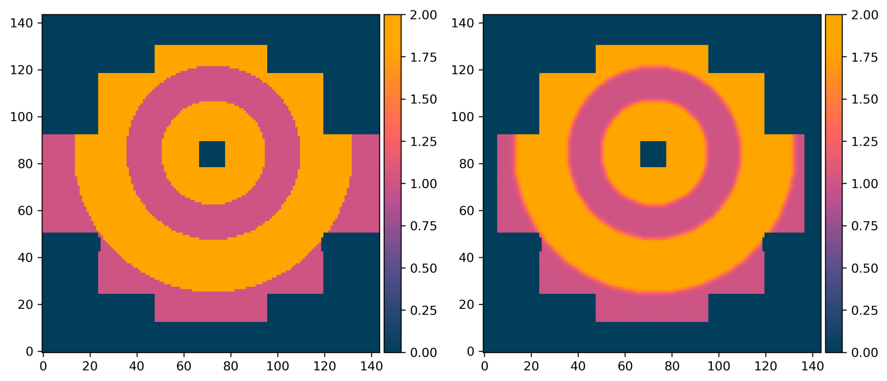


Fig. 2.14 Simulated Q-resolution smearing by applying a 2D-Gaussian resolution function to each data point. The left image shows a test detector image with two rings of intensity 2.0 (yellow) and all other values set to 1.0. The detector mask as used for KWS-1 is superimposed. On the right the resolution function is applied leading to a smearing of the edges of the sample image.

where $\boldsymbol{Q}_0$ denotes the average scattering vector at which the radiation is detected. This expression is valid for both X-ray and neutron small-angle scattering. The instrumental resolution leads to smearing of the experimentally recorded scattering intensity (fig. 2.14) thus it has to be accounted for to obtain reliable parameters of the particle size distribution that also produces smearing (fig. 2.13). Three contributions to the resolution smearing have to be considered, namely the wavelength spread, the angular uncertainty and detector pixel size, all of which result in an uncertainty in Q.[119,120]

Using the expression $Q = 4\pi/\lambda \sin\theta/2$ for the magnitude of the scattering vector the partial derivatives with respect to the wavelength λ and the scattering angle θ are

$$\frac{\delta Q}{\delta\lambda} = \frac{-4\pi}{\lambda^2}\sin\frac{\theta}{2} = -\frac{1}{\lambda}Q \tag{2.100}$$

$$\frac{\delta Q}{\delta\theta} = \frac{2\pi}{\lambda}\cos\frac{\theta}{2}. \tag{2.101}$$

These express the uncertainties in the wavelength arising from the monochromator or velocity selector and in the scattering angle originating from the collimation and the detector pixel sizes. The full width at half maximum (FWHM) of the resolution function is given by the sum of the FWHM of the components resulting from angular and wavelength uncertainties according to

$$\sigma^2 = \sigma_\theta^2 + \sigma_\lambda^2 = \left(\frac{1}{2\sqrt{2\ln 2}}\frac{\delta Q}{\delta\theta}d\theta\right)^2 + \left(\frac{1}{2\sqrt{2\ln 2}}\frac{\delta Q}{\delta\lambda}d\lambda\right)^2, \tag{2.102}$$

thus yielding

$$\sigma = \frac{1}{2\sqrt{2\ln 2}}\sqrt{\left(\frac{2\pi}{\lambda}\cos\frac{\theta}{2}d\theta\right)^2 + \left(\frac{d\lambda}{\lambda}Q\right)^2}. \tag{2.103}$$

With this the combined resolution function for radially averaged data is given as[119]

$$Q_{res}(Q, Q_0, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{(Q - Q_0)^2}{\sigma^2}\right)\right]. \tag{2.104}$$

2.3.4.6 Small-angle scattering of polarized neutrons (SANSPOL)

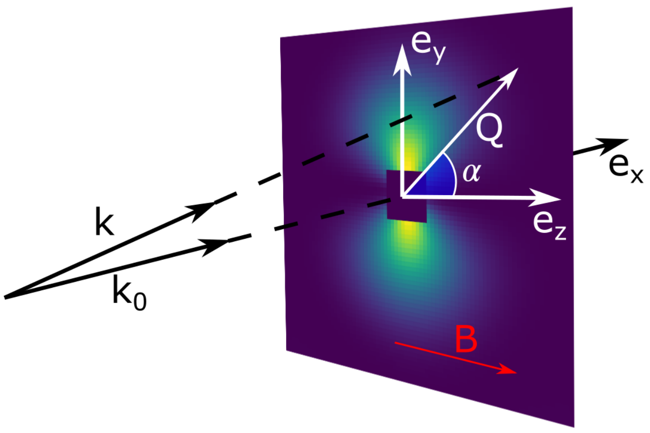


Fig. 2.15 Scattering geometry of the SANSPOL experiment. The wave vectors of the incident and scattered beam are denoted by $\boldsymbol{k}_0$ and $\boldsymbol{k}$, respectively. The scattering vector $\boldsymbol{Q} = \boldsymbol{k}_0 - \boldsymbol{k}$ forms an angle α with the applied field vector $\boldsymbol{B}$. The Cartesian coordinate system is chosen such that the unit vector $\boldsymbol{e}_x$ points along the primary beam and the vector $\boldsymbol{e}_z$ lies parallel to the applied field direction. The image shows the simulated intensity according to eq. 2.113 with the $\sin^2\alpha$ dependence resulting in the characteristic shape.

Small-Angle Neutron Scattering with POLarized neutrons (SANSPOL) is used to study the magnetic properties of nanoparticles. Neutron polarization, i.e. the alignment of the neutron spin in a certain direction, can be achieved by multiple methods, e.g. by absorption of the undesired polarization direction in ^3He filters. The scattering geometry is shown in fig. 2.15, where the applied field is perpendicular to the neutron beam direction. Due to the large sample-to-detector distance the x-component of the scattering vector $\boldsymbol{Q}$ can be ignored in a first approximation and the intensity as a function of the y and z components is recorded on the detector (fig. 2.15). The angle between applied field and $\boldsymbol{Q}$ is denoted with α. As mentioned briefly in section 2.3.2 the neutron magnetic dipole moment interacts with the field produced by unpaired electrons of ions resulting in magnetic scattering. In the case of small-angle neutron scattering on magnetic iron oxide nanoparticles it is the dipole moment of the particles with which the neutrons interact. The interaction potential between a neutron and an electron is composed of a contribution due to the spin of the electron, $\boldsymbol{B}_S(\boldsymbol{r})$, and due to its orbital momentum, $\boldsymbol{B}_L(\boldsymbol{r})$

$$V_M(\boldsymbol{r}) = -\boldsymbol{\mu}_n \cdot (\boldsymbol{B}_L(\boldsymbol{r}) + \boldsymbol{B}_S(\boldsymbol{r})), \tag{2.105}$$

where μ_n is the magnetic moment of the neutron. It can be shown that this potential field is related to the magnetization of the sample. In particular, only the component $\boldsymbol{M}_\perp$ of the magnetization that is perpendicular to the scattering vector enters the potential and thus leads to scattering of neutrons[121]

$$V_M(\boldsymbol{Q}) = -\boldsymbol{\mu}_n \cdot \boldsymbol{B}(\boldsymbol{Q}) = -\mu_0 \boldsymbol{\mu_n} \boldsymbol{M}_\perp. \tag{2.106}$$

The magnetic interaction between a neutron and an atom can be described by a magnetic atomic form factor and similar to the nuclear small-angle scattering a magnetic scattering length density, SLD_m, is introduced. The magnetic form factor is given by the Fourier transform of the spin density of the ion and is therefore dependent of the scattering angle, similar to X-ray scattering form factors.[121] In small-angle scattering the approximation is made that this $\boldsymbol{Q}$-dependence is not significant in the angle region of consideration and the normalized magnetic form factor is assumed to be unity and thus the magnetic scattering length depends only on the magnetic moment of the respective atom. Therefore, the magnetic scattering length density can be calculated from the magnetic moment components $\boldsymbol{M}_{i,\perp}$ of atom i in units of the Bohr magneton μ_B, that are in the plane perpendicular to the scattering vector, according to

$$\rho_m = \mathrm{SLD}_m = C \frac{1}{V_m} \sum_{j=1}^{N} \boldsymbol{M}_{i,\perp}, \tag{2.107}$$

where the sum is over the atoms in the volume V_m and the constant C is given by $C = \frac{1}{2} g_n r_0 = 2.7 \times 10^{-5}$ Å, with the gyromagnetic factor of the neutron g_n and the classical electron radius r_0. This scattering length density is experimentally accessible and, if recorded at saturating fields, is related to the saturation magnetization of a magnetic nanoparticle via

$$M_{sat} = \frac{\mathrm{SLD}_m \mu_B}{C\rho} \frac{V_{eff}}{V_t}, \tag{2.108}$$

where μ_B is the Bohr magneton. The effective magnetic volume is given by V_{eff} and the total particle volume is V_t. The density of the material is denoted by ρ. This results in saturation magnetization values in units of Am^2/kg. Similar to the case of nuclear neutron scattering a contrast $\Delta\rho = \langle \rho_{particle} \rangle - \rho_{surrounding}$ is used, where angle brackets denote the compositional average. Since usually organic solvents are used the magnetic scattering length density of the solvent can be assumed to be zero. Completely analogous to the nuclear scattering case a magnetic particle form factor can be defined as the Fourier transform of the magnetic scattering length density and is in the following symbolized by $F_M(\boldsymbol{Q})$. In SANSPOL the spins of the neutrons are aligned either parallel or anti-parallel to the applied field by the use of a combination of neutron polarizer and neutron spin flipper. After the scattering process the neutron spin can be either unchanged (non spin flip scattering, nsf) or flipped (spin flip scattering, sf) and the resulting scattered intensity is given by the sum of nsf and sf.[122] For non-interacting polydisperse particles in a saturating magnetic field the intensity is given as[123,124]

$$I^{+}(\boldsymbol{Q}) = |A^{++}|^2 + |A^{+-}|^2 \tag{2.109}$$

$$I^{-}(\boldsymbol{Q}) = |A^{--}|^2 + |A^{-+}|^2, \tag{2.110}$$

where A denotes the scattering amplitude. If the neutron polarization is along the scattering vector only coherent nuclear scattering gives rise to nsf and magnetic scattering leads to spin flip scattering. Then the following relations can be set up[123]

$$|A^{\pm\pm}|^2 = |F_N(\boldsymbol{Q})|^2 \tag{2.111}$$

$$|A^{\pm\mp}|^2 = \left[|F_M(\boldsymbol{Q})|^2 - 2P(1-2\epsilon^{\pm})F_N(\boldsymbol{Q})F_M(\boldsymbol{Q})\right]\sin^2\alpha, \tag{2.112}$$

with α as the angle between $\boldsymbol{Q}$ and the applied field (see fig. 2.15). The flipper efficiency is ϵ and is a value close to 1 for spin up and 0 for spin down, because the polarizer produces a down polarized neutron beam that has to be flipped for the spin up channel. The polarization is denoted by P and is defined as $P = (n^{+} - n^{-})/(n^{+} + n^{-})$, where n^{+} and n^{-} are the numbers of neutrons with spins parallel and anti-parallel to the applied field.[123] The $\sin^2\alpha$ dependence originates from the $\sin\alpha$ dependence of the magnetic scattering amplitude, which is strongest for perpendicular arrangement of the magnetic moment to the field vector and decreases to zero for parallel alignment. Subtracting $I^{+}(\boldsymbol{Q})$ from $I^{-}(\boldsymbol{Q})$ yields the nuclear magnetic interference term according to

$$I^{-}(\boldsymbol{Q}) - I^{+}(\boldsymbol{Q}) = 4P(1+\epsilon^{-})F_N(\boldsymbol{Q})F_M(\boldsymbol{Q})\sin^2\alpha. \tag{2.113}$$

2.3.4.7 Contrast Variation

In general the treatment of contrast in small-angle scattering is simplified by the use of basic functions according to

$$I(Q) = I_s(Q) + \Delta\rho I_{cs}(Q) + (\Delta\rho)^2 I_c(Q), \tag{2.114}$$

where $I(Q)$ is the total scattering intensity, $I_s(Q)$ corresponds to the scattering from density fluctuations within the particles and $I_c(Q)$ designates the scattering from the particle shape.[125] The contrast is defined as the difference between the average particle scattering length density (SLD) and the solvent SLD, $\Delta\rho = \bar{\rho} - \rho_{solv.}$. At $Q = 0$ only the term from the particle shape contributes and is given by the square of the particle volume V_p (the volume inaccessible by the solvent, i.e. core plus shell volume) times the number density n. The above equation becomes

$$I(0) = (\Delta\rho)^2 n V_p^2, \tag{2.115}$$

thus showing the square dependence of $I(0)$ on the contrast. $I(0)$ can be determined by application of the Guinier approximation to the small Q region (sec-

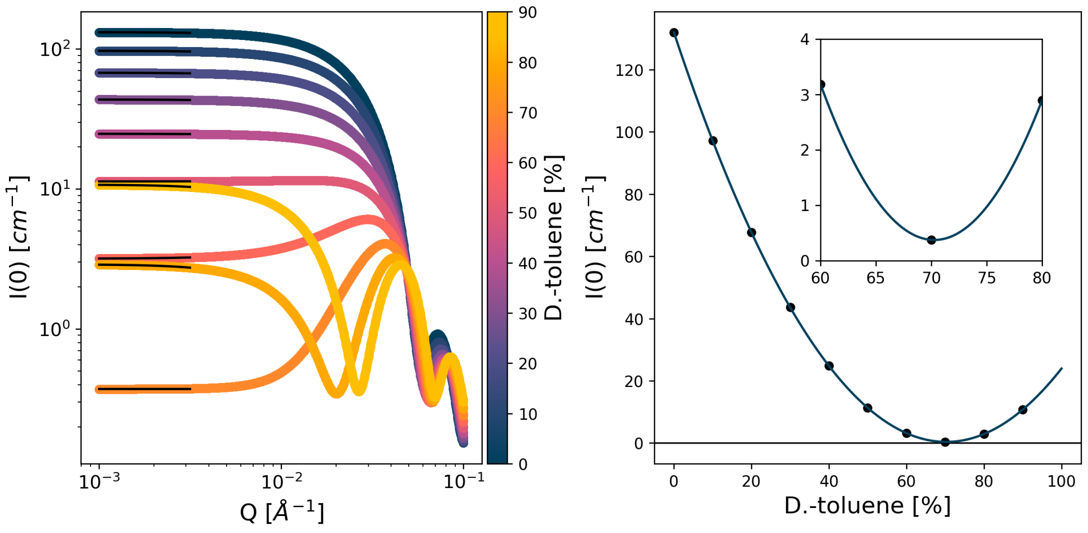


Fig. 2.16 Simulated contrast variation for core-shell nanoparticles in a solvent with varying SLDs. The core radius was taken as $78\,\text{Å}$, the shell thickness as $14\,\text{Å}$. A lognormal size distribution was used with $\sigma = 0.1$. The core SLD and shell SLD values of 6.8×10^{-6} and $0.078 \times 10^{-6}\,\text{Å}^{-2}$ were used. The solvent SLDs correspond to deuterated toluene in the solvent from 0 to $90\,\%$, using the SLDs for toluene and deuterated toluene as given in tab. 2.1. On the right the $I(Q=0)$ determined from Guinier fits to the low-Q region (black lines on the left) are shown as a function of the solvent deuteration, i.e. the contrast. From the matchpoint obtained as the minimum of a parabolic fit to the $I(Q=0)$ the input core SLD was calculated according to eq. 2.121. The determined value matches well with the input parameter. The inset on the right shows the non-zero value of $I(0)$ at the match point due to polydispersity.

tion 2.3.4.8). A plot of $I(0)$ against the scattering length density contrast for different solvent SLDs shows a minimum at the so-called match point where the average SLD of the particles is equal to the solvent SLD at that point. For polydispersity in the studied particles the size distribution has to be considered, i.e. following Avdeev[125]

$$I(Q) = \langle I_s(Q) \rangle + \langle \Delta\rho I_{cs}(Q) \rangle + \langle (\Delta\rho)^2 I_c(Q) \rangle, \tag{2.116}$$

where $\langle \dots \rangle$ denotes $\int D(R, R_0, \sigma) \dots dR$. A modified contrast is defined as $\Delta\rho_{mod.} = \bar{\rho}_e - \rho_{solv.}$, where $\bar{\rho}_e$ is an effective SLD of the particles. The modified scattering length density can be calculated via

$$\bar{\rho}_e = \frac{\langle \bar{\rho} V_p^2 \rangle}{\langle V_p^2 \rangle}. \tag{2.117}$$

Since V_p is given by $V_p = V_c + V_s$, where V_c and V_s are the inorganic core and the organic shell volumes respectively eq. 2.117 is written as

$$\bar{\rho}_e = \frac{\langle \bar{\rho}(V_c + V_s)^2 \rangle}{\langle (V_c + V_s)^2 \rangle}. \tag{2.118}$$

$\bar{\rho}$ is given by $(V_c\rho_c + V_s\rho_s)/(V_c + V_s)$ thus eq. 2.118 gives

$$\bar{\rho}_e = \frac{\langle (V_c\rho_c + V_s\rho_s)(V_p) \rangle}{\langle V_p^2 \rangle}, \tag{2.119}$$

making use of the addition rule for integration the angle brackets in the numerator on the write can be separated yielding

$$\bar{\rho}_e \langle V_p^2 \rangle = \langle V_c V_p \rho_c \rangle + \langle V_s V_p \rho_s \rangle. \tag{2.120}$$

Now it can be assumed that ρ_c and ρ_s are constant with respect to the particle size and thus can be pulled out of the integrals. Finally solving for the core SLD and with the definition of $\bar{\rho}_e = \rho_{mp}$, i.e. at the match point (mp) the effective SLD is equal to the solvent SLD, gives

$$\rho_c = \frac{\rho_{mp} \langle V_p^2 \rangle - \rho_s \langle V_s V_p \rangle}{\langle V_c V_p \rangle}. \tag{2.121}$$

Thus by experimentally determining the match point scattering length density the SLD of the particle core can be obtained, provided that the organic shell thickness as well as the shell SLD is known. It should be noted that in the presence of polydispersity for core-shell nanoparticles full contrast matching cannot be achieved as the volume averaged SLD for particles with different inorganic core sizes is slightly different due to a different volume ratio between the organic shell and the inorganic core. A simulated contrast variation experiment of polydisperse core-shell nanoparticles in a solvent is shown in fig. 2.16, where this effect can be seen.

2.3.4.8 Particle form and structure factors

In this section analytical expressions for particle form and structure factors are discussed. Including the solid sphere form factor and the Guinier approximation, as well as the spherical core-shell model. The sticky hard sphere structure factor is discussed as it provides a good description of the interactions observed for iron oxide nanoparticles induced by a magnetic field.

Solid sphere form factor For a solid sphere with radius R the form factor is obtained by the Fourier transform of the density distribution $\rho(\boldsymbol{r})$, which in this case is equal to the density of the sphere inside the particle ($r < R$) and zero outside ($r > R$). As stated before

$$F(\boldsymbol{Q}) = \int\limits_V \Delta\rho(\boldsymbol{r}) e^{i\boldsymbol{Q}\cdot\boldsymbol{r}} dV. \tag{2.122}$$

Using the definition of the contrast and assuming the scattering length density to be constant for small-angle scattering the integral can be carried out in spherical coordinates according to

$$F(\boldsymbol{Q}) = \Delta\rho \int\limits_0^R \int\limits_0^\pi \int\limits_0^{2\pi} e^{i\boldsymbol{Q}\cdot\boldsymbol{r}} r^2 dr \sin\theta d\theta d\phi. \tag{2.123}$$

This integral with respect to θ and ϕ has been shown in the context of the derivation of the Debye scattering equation (section 2.3.3). With that the above equation reads

$$F(Q) = \Delta\rho(2\pi) \int\limits_0^R \frac{2\sin Qr}{Qr} r^2 dr. \tag{2.124}$$

The integral over the particle radius yields[114,117]

$$\begin{aligned} F(Q) &= \Delta\rho \frac{4\pi}{Q} \left[\frac{\sin(QR) - QR\cos(QR)}{Q^2} \right] \\ &= \Delta\rho 4\pi R^3 \left[\frac{\sin(QR) - QR\cos(QR)}{(QR)^3} \right] \\ &= 3\Delta\rho V_{particle} \left[\frac{\sin(QR) - QR\cos(QR)}{(QR)^3} \right]. \end{aligned} \tag{2.125}$$

Guinier approximation In the low-Q region the small-angle scattering curve can be approximated by the so-called Guinier law

$$I(Q) = I(0) e^{-\frac{R_g^2 Q^2}{3}}, \tag{2.126}$$

where $I(0)$ is the scattering intensity in forward direction, given by the square of the total particle scattering length, i.e. $I(0) = n(\Delta\rho V)^2$, with the particle volume V, the scattering contrast $\Delta\rho$ and the particle number density n.[106] The radius of gyration of the particles is R_g and can be interpreted as the average distance of parts of a body to its center of mass, thus it is generally not equal to the physical particle radius. For homogeneous spherical particles the relation between radius of gyration and physical radius is $R_g = \sqrt{\frac{3}{5}} R$. In the so-called Guinier plot of $\ln I(Q)$ vs. Q^2 the Guinier law is linear and R_g can be determined from the slope.[126] The

Guinier law is valid only in the region $qR_g < 1.3$ and deviations of the data from the linear behaviour in this representation indicate particle aggregation.

Spherical Core Shell Form Factor The core shell model is used to describe nuclear neutron scattering from particles with a coating of different scattering length density than the particle core or in magnetic scattering the difference between a magnetic core and a magnetically different shell. The spherical core shell form factor is constructed in a similar way as the solid sphere in a solvent (fig. 2.12) except that the particle contribution is split into a contribution from the particle core and one from the shell according to

$$\begin{aligned} F(Q)_{core-shell} &= F(Q)_{core} + F(Q)_{shell} + F(Q)_{solvent} \\ &= \int\limits_{V_{core}} \rho_c e^{i\boldsymbol{Qr}} dV + \int\limits_{V_{shell}} \rho_{shell} e^{i\boldsymbol{Qr}} dV + \int\limits_{V-V_{particle}} \rho_{solv.} e^{i\boldsymbol{Qr}} dV \\ &\approx \int\limits_{V_{core}} \rho_c e^{i\boldsymbol{Qr}} dV + \int\limits_{V_{shell}} \rho_{shell} e^{i\boldsymbol{Qr}} dV - \int\limits_{V_{particle}} \rho_{solv.} e^{i\boldsymbol{Qr}} dV, \end{aligned} \tag{2.127}$$

where in the last step the same assumption as in section 2.3.4.2 was made that the solvent scattering contribution is negligible in the observable angular range. The general equation of the form factor for spherically symmetric body is given as

$$F(Q) = 4\pi \int \rho(r) \frac{\sin(Qr)}{Qr} r^2 dr. \tag{2.128}$$

The contribution from the particle core is the solid sphere form factor derived above for a particle in vacuum, i.e.

$$F(Q)_{core} = 3\rho_{core} V_{core} \left[\frac{\sin(QR_c) - QR_c \cos(QR_c)}{(QR_c)^3} \right], \tag{2.129}$$

where R_c is the particle core radius. The solvent contribution is

$$\int\limits_{V_{particle}} \rho_{solv.} e^{i\boldsymbol{Qr}} dV = 3\rho_{solv.} V_{particle} \left[\frac{\sin(QR_t) - QR_t \cos(QR_t)}{(QR_t)^3} \right], \tag{2.130}$$

where the total particle radius R_t is the sum of the particle core radius and the shell thickness t, i.e. $R_t = R_c + t$. The shell contribution is given as

$$F(Q)_{shell} = 4\pi \int\limits_{R_c}^{R_t} \rho(r)_{shell} \frac{\sin(Qr)}{Qr} r^2 dr, \tag{2.131}$$

which under the assumption of a constant ρ_{shell} yields

$$\begin{aligned} F(Q)_{shell} = {} & 3\rho_{shell}V_{particle}\frac{\sin(QR_t) - QR_t\cos(QR_t)}{(QR_t)^3} \\ & - 3\rho_{shell}V_{core}\frac{\sin(QR_c) - QR_c\cos(QR_c)}{(QR_c)^3}. \end{aligned} \tag{2.132}$$

Together with the expressions for the solvent and the core contribution the core shell form factor is[114]

$$\begin{aligned} F(Q)_{core-shell} = {} & 3(\rho_{core} - \rho_{shell})V_{core}\frac{\sin(QR_c) - QR_c\cos(QR_c)}{(QR_c)^3} \\ & + 3(\rho_{shell} - \rho_{solvent})V_{particle}\frac{\sin(QR_t) - QR_t\cos(QR_t)}{(QR_t)^3}. \end{aligned} \tag{2.133}$$

Sticky hard sphere structure factor The sticky hard sphere model can be derived as a perturbative solution of the factorized Ornstein-Zernike equation or from the Percus-Yevick approximation for a square well potential. It is suitable to describe the interactions of sterically coated particles, i.e. short ranged attractive potentials. The parameters that enter this model are the stickiness, the volume fraction of dispersed particles, the perturbation parameter and the effective particle radius R_S. The latter is defined as the sum of the particle radius R_t and an effective additional radius. The perturbation parameter is usually set to zero. The structure factor is given as

$$S(Q) = \frac{1}{A(Q)^2 + B(Q)^2}, \tag{2.134}$$

where A(Q) and B(Q) are lengthy expressions that can be found in Menon et al.[127]. A python implementation based on the code from SasView is given in appendix D. A simulation is shown in fig. 2.17.

2.3.5 Wide-angle scattering

This area of scattering theory deals with the atomic arrangements of structures and in this thesis is used to obtain information on the crystal structure of the particle core, including structural defects and features such as vacancy ordering. The first part of this section is concerned with general considerations regarding the coherent scattering from ordered crystalline structures, with a focus on isotropic powder samples. The second part deals with the real space analysis of the local atomic structure with the pair distribution function method. Since in this thesis only X-rays were used for both techniques the theory is developed for this kind of radiation. However, in principle with some minor modifications it holds for scattering of neutrons as well.

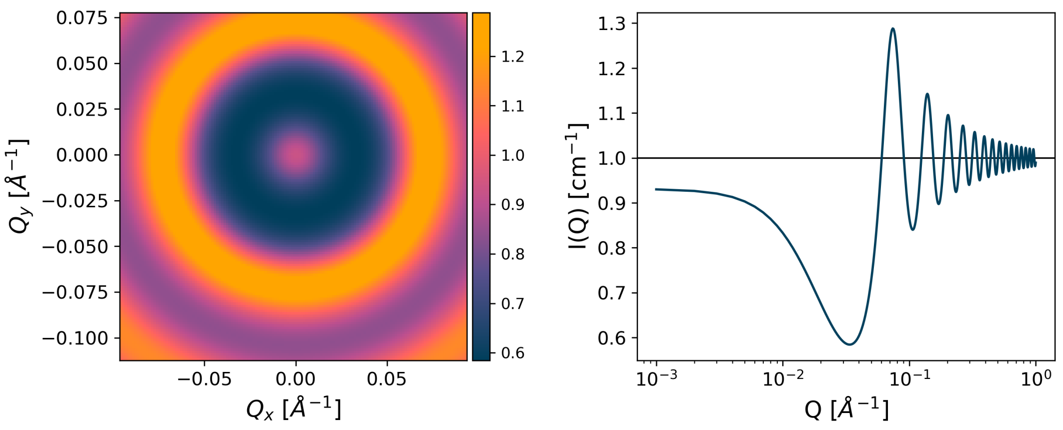


Fig. 2.17 Simulated sticky hard sphere structure factor in 2D and 1D. The structure factor is isotropic with respect to the scattering vector orientation and oscillates around 1 for large Q. An effective radius $R_S = 50\,\text{Å}$ was used together with a stickiness of 0.2 and a volume fraction of 0.2. The perturbation parameter was set to zero.

2.3.5.1 Bragg's law and Miller-indices

A condition for constructive interference of waves was found by W.H. and W.L. Bragg. From simple geometric considerations they arrived at the expression given in eq. 2.86 as

$$2d \sin\theta = n\lambda,$$

where d is the distance between crystal lattice planes, θ is the scattering half-angle, λ is the wavelength and n is the order of the reflection.[113]

For every set of parallel planes in a crystal a vector can be constructed that is perpendicular to these planes. To describe the orientation of the planes in the crystal so-called *Miller-indices* are used. These are obtained from the intersections of the planes with the coordinate system axis. E.g. in the cubic system the intersection with the crystallographic a-axis gives $1/h$, where h is the Miller-index. Similar for the other axis such that a triple of integers hkl can be found that describes the orientation of the set of planes. A Miller-index of 0 indicates that the planes are parallel to the respective axis, i.e. the planes with indices (100), (010) and (001) only intersect the a, b and c axis, respectively. In a cubic crystal system with lattice parameter a the lattice plane distance d is connected to the Miller-indices via

$$d_{hkl} = \frac{a}{\sqrt{h^2 + k^2 + l^2}}. \tag{2.135}$$

A reciprocal crystal lattice can be constructed by setting up reciprocal lattice vectors $\boldsymbol{b}_i$ that are perpendicular to two real space lattice vectors. Points in this lattice can be described by a vector $\boldsymbol{H} = h\boldsymbol{b}_1 + k\boldsymbol{b}_2 + l\boldsymbol{b}_3$, where h, k, l are the Miller-indices. Thus each point in the reciprocal lattice corresponds to a set of parallel planes in the real lattice. The length of a reciprocal lattice vector is $|\boldsymbol{H}| = 2\pi/d_{hkl}$, i.e. inversely proportional to the lattice plane distance (fig. 2.18b).[109]

2.3.5.2 Scattering from crystals

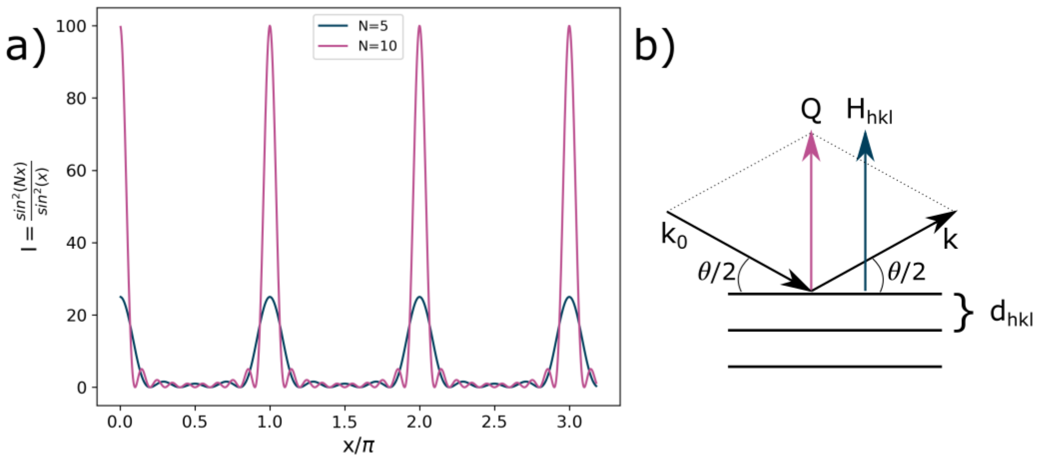


Fig. 2.18 a) Plot of the function $I = \sin^2(Nx)/\sin^2(x)$ appearing in eq. 2.139 for two values of N. It is evident that intensity maxima occur at $x = n\pi$ and their intensity and sharpness scales with N. b) Sketch of the scattering geometry illustrating the relationship between the scattering vector $\boldsymbol{Q}$ and the reciprocal lattice vector $\boldsymbol{H}_{hkl}$.

The basis of this discussion is again the definition of the scattering amplitude given in eq. 2.79 according to

$$A(\boldsymbol{Q}) = \int\limits_V \rho(\boldsymbol{r}) e^{i\boldsymbol{Qr}} dV, \tag{2.136}$$

where in diffraction by crystals the potential field $\rho(\boldsymbol{r})$ can be represented as localized potentials at the atomic sites and the scattering amplitude is given as the superposition of the scattered waves from the individual potentials. The observed interference pattern is due to the phase difference between waves scattered from different points in space, i.e. atomic positions.

The following discussion is based mainly on the work from B.E. Warren.[109] The scattering vector is $\boldsymbol{Q} = \boldsymbol{k} - \boldsymbol{k}_0$, where $\boldsymbol{k}_0$ and $\boldsymbol{k}$ are the wave vectors of the incident and the outgoing waves, respectively. The integral of eq. 2.136 can be replaced by a sum as the scattering potentials are assumed to be localized on the discrete atomic positions, therefore

$$A(\boldsymbol{Q})_{crystal} = \sum_n f_n(Q) e^{i\boldsymbol{QR}_n}, \tag{2.137}$$

where the scattering potentials were replaced by the atomic form factors for X-ray scattering and $\boldsymbol{R}_n$ are the atomic positions in the crystal structure. The long range order of the crystal structure allows a subdivision into smaller units that are periodically arranged to return the total structure. These units are called unit cells, which are defined by lattice parameters $\boldsymbol{a}$, symmetry operations and atomic positions. The position of a unit cell in the crystal structure is described by the vector $\boldsymbol{ma} = m_1a_1 + m_2a_2 + m_3a_3$, where m_1, m_2 and m_3 refer to the numbers of unit cells along a_1, a_2 and a_3, respectively. Thus the position of an atom relative to the crystal origin is given by $\boldsymbol{R}_n = \boldsymbol{ma} + \boldsymbol{r}_n$, where $\boldsymbol{r}_n$ denotes the position of the atom n inside the unit cell. Thus the scattering amplitude of the wave scattered by a periodic crystal structure is

$$\begin{aligned} A(\boldsymbol{Q})_{crystal} &= \sum_n f_n(Q)e^{i\boldsymbol{Q}(\boldsymbol{ma}+\boldsymbol{r}_n)} \\ &= \sum_n f_n(Q)e^{i\boldsymbol{Q}\boldsymbol{r}_n} \sum_{m_1=0}^{N_1-1} e^{i\boldsymbol{Q}(m_1a_1)} \sum_{m_2=0}^{N_2-1} e^{i\boldsymbol{Q}(m_2a_2)} \sum_{m_3=0}^{N_3-1} e^{i\boldsymbol{Q}(m_3a_3)}. \end{aligned} \tag{2.138}$$

The first sum is called *structure factor*, since it contains the structural information, i.e. the atomic positions in the unit cells. To complete the confusion of terms and symbols from small-angle scattering and wide-angle scattering it is commonly denoted by $F(Q)$. The other sums relate to the phase shifts originating from the placement of the unit cells. The scattered intensity is proportional to the absolute square of the scattering amplitude giving

$$\frac{d\sigma}{d\Omega} \propto |A(\boldsymbol{Q})_{crystal}|^2 = |F(\boldsymbol{Q})|^2 \frac{\sin^2(\boldsymbol{Q}\cdot N_1\boldsymbol{a}_1)}{\sin^2(\boldsymbol{Q}\boldsymbol{a}_1)} \frac{\sin^2(\boldsymbol{Q}\cdot N_2\boldsymbol{a}_2)}{\sin^2(\boldsymbol{Q}\boldsymbol{a}_2)} \frac{\sin^2(\boldsymbol{Q}\cdot N_3\boldsymbol{a}_3)}{\sin^2(\boldsymbol{Q}\boldsymbol{a}_3)}, \tag{2.139}$$

where the fact was used that the sums in eq. 2.138 can be represented as geometric series. N_i is the number of unit cells in direction $\hat{e}_i$. There are two major observations to be made:

1. Eq. 2.139 has maxima where $\boldsymbol{Q}\cdot\boldsymbol{a}_i$ is equal to integer multiples of π. It follows that a maximum in the differential cross section is observed only when this is satisfied for all three fractions in eq. 2.139 simultaneously which leads to the three *Laue equations*

$$\boldsymbol{Q}\cdot\boldsymbol{a}_1 = h\pi \tag{2.140}$$
$$\boldsymbol{Q}\cdot\boldsymbol{a}_2 = k\pi \tag{2.141}$$
$$\boldsymbol{Q}\cdot\boldsymbol{a}_3 = l\pi, \tag{2.142}$$

   where the h, k and l are integers. It can be shown that this is equivalent to the vector representation of the Bragg condition for scattering that states that for constructive interference to occur the scattering vector $\boldsymbol{Q}$ has to be parallel and equal in length to a reciprocal lattice vector $\boldsymbol{H}_{hkl}$, defined by the reciprocal coordinates h, k and l and with length $|\boldsymbol{H}_{hkl}| = 2\pi/d_{hkl}$, where d_{hkl} is the spacing of hkl-planes (see section 2.3.5.1 and fig. 2.18b). Thus

$$\boldsymbol{Q} = \boldsymbol{H}_{hkl}, \tag{2.143}$$

from which follows

$$|\boldsymbol{Q}| = |\boldsymbol{H}_{hkl}| \rightarrow \frac{4\pi}{\lambda}\sin\frac{\theta}{2} = \frac{2\pi}{d_{hkl}}, \longleftrightarrow \lambda = 2d_{hkl}\sin\frac{\theta}{2} \tag{2.144}$$

where the right equation is Bragg's law as given in eq. 2.86.

2. The sharpness of the maxima of eq. 2.139 is influenced by the crystal dimensions N_i. If the crystal dimensions are large, eq. 2.139 is essentially 0 everywhere, except for the reciprocal lattice points, defined by $\mathbf{H}_{hkl}$. The width of a peak produced by a function $\sin^2(N\pi x)/\sin^2(\pi x)$ gets larger as N decreases, thus for smaller crystals broader peaks are observed (fig. 2.18a). This fact can be used to determine crystallite sizes from powder diffraction patterns.

2.3.5.3 Powder diffraction

A powder sample consists of a large number of crystallites that are typically randomly oriented. The consequence of this is that the Bragg condition (eq. 2.144) is satisfied simultaneously for all sets of lattice planes. Additionally, Bragg peaks are not observed as points in reciprocal space as is the case for single crystal diffraction but produce rings at the scattering angle θ for the respective reflection due to the presence of crystals in different rotations with respect to the incident wave. Integration of these rings yields the powder diffraction pattern, where the scattered intensity is depicted as function of the diffraction angle θ or as function of the length of the scattering vector $\boldsymbol{Q}$, respectively. A number of parameters can be extracted from this pattern. From the position of peaks the lattice parameters can be obtained (through eqs. 2.135 and 2.144, if the structure is already known), the width of peaks contains information on the particle size, lattice strain and lattice defects, such as antiphase boundaries. The relative intensity of peaks is directly related to the structure factor (eq. 2.138), which can be used to perform structure refinement with e.g. the Rietveld method.[128] Another approach to study the structural properties through powder diffraction data is the method of whole powder pattern modelling (WPPM) introduced by Scardi et al.[129] Here the diffraction peaks are modeled using the Fourier coefficients of the different contributions to the peak profile, such as an instrumental component and the peak profile resulting from the finite size. Other contributions can be included as well if the respective Fourier coefficients are known. In general a peak is described by

$$I_{\mathrm{hkl}}(Q, Q_{\mathrm{hkl}}) = k\int_0^\infty C_{\mathrm{hkl}} e^{\pi i(Q-Q_{\mathrm{hkl}})} dL, \tag{2.145}$$

where C_{hkl} are the aforementioned Fourier coefficients, L is a distance in real space along $[hkl]$, Q_{hkl} is the peak position and k stands for constant terms such as the square of the structure factor.

Peak shapes In the context of nanoparticle research the dependence of the peak shape and width on the particle size deserves special attention, as it provides another method to determine the particle size in addition to the more commonly used methods of SAXS and TEM. The sizes obtained from powder diffraction patterns relate to the coherence length in a direction perpendicular to the set of lattice planes of the corresponding diffraction peak. Therefore, it is generally possible to obtain different sizes for particles from different peaks if the coherence lengths in a crystal change for different crystallographic directions. To obtain sizes

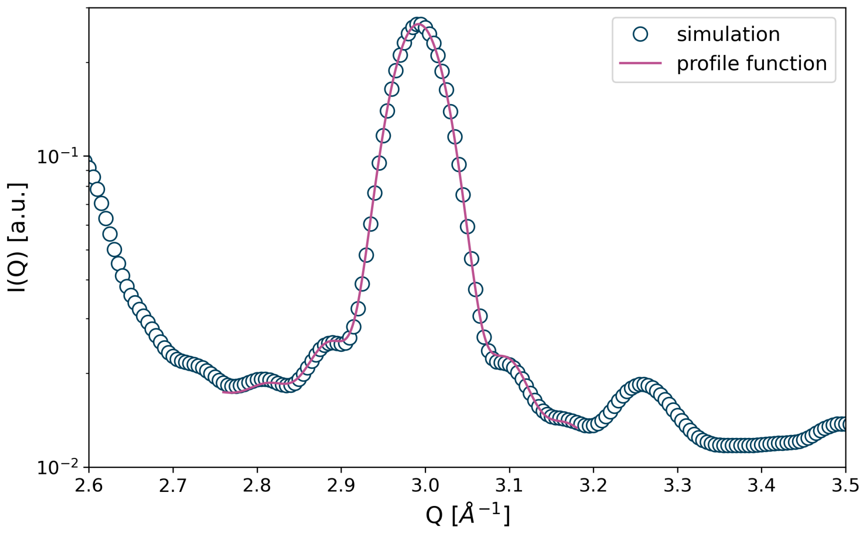


Fig. 2.19 Comparison of the peak shape obtained from a powder pattern simulation using the Debye scattering equation for a spherical particle to a theoretical peak shape based on eq. 2.149. The small satellite peaks due to the sphere shape as well as the peak broadening due to the finite size of the particle are well described by the profile function.

from the peak width of diffraction patterns the most often employed approach is the Scherrer equation[130], that was first derived for cubic crystallites and relates the full width at half maximum (FWHM) in radians to the cube edge length L according to

$$B(\theta) = \frac{K\lambda}{L\cos\theta/2}, \tag{2.146}$$

where λ is the wavelength and θ is the scattering angle. K is a constant related to the crystallite shape. Different values can be found depending on the approximations made during the derivation.[131] It can be shown that the correct value of K for spherical particles is 1.107.[131] However, in practice often values of 0.94 or 1.0 are used for spherical particles with cubic unit cells.[132] This shows that there is a degree of arbitrariness involved in determining sizes with the Scherrer equation. Although efforts are made to modify the Scherrer equation for use on nanomaterials[132] the application of the equation as given in eq. 2.146 is discouraged for use in determining absolute values of nanoparticle sizes and should only be used to obtain relative sizes from peak widths to gain insight into possible systematic changes of coherent sizes.[133,134] As pointed out by Scardi et al.[135] a more adequate description of the peak shape produced by small crystallites is achieved by an expression derived by Patterson[131]. He showed that the peak shape can be directly obtained from a Fourier transform of the particle profile function or common volume function. This function appears again as the autocorrelation function of the shape function in pair distribution function analysis. The profile function is given as the intersection volume, V_i between the particle with diameter D and an identical copy shifted by a distance x along the scattering vector normalized to the particle volume V

$$A_P(x,D) = \frac{V_i(x,D)}{V(D)} = \frac{\frac{\pi}{12}(D-x)^2(2D+x)}{\frac{4}{3}\pi(D/2)^3} = 1 - \frac{3}{2}\frac{x}{D} + \frac{1}{2}\left(\frac{x}{D}\right)^3. \tag{2.147}$$

This profile function can be directly used in eq. 2.145. Alternatively for this peak shape there exists also an analytical expression obtained from the Fourier transform of eq. 2.147 according to

$$I(Q,D) = \int_0^D A_P(x,D)e^{ixQ}dx, \tag{2.148}$$

which leads to

$$\begin{aligned} I(Q,D) = &\frac{(\sin\Delta - \Delta\cos\Delta)^2 + \Delta^2\sin^2\Delta}{\Delta^4} \\ &- \frac{(\sin\Xi - \Xi\cos\Xi)^2 + \Xi^2\sin^2\Xi}{\Xi^4}, \end{aligned} \tag{2.149}$$

where

$$\Delta = \frac{D}{2}(Q - Q_B) \tag{2.150}$$

$$\Xi = \frac{D}{2}(Q + Q_B), \tag{2.151}$$

with the peak position of the Bragg peak, Q_B.[131,135] The use of this peak profile for the determination of particle sizes removes the arbitrariness introduced by the choice of peak profiles used to derive the FWHM and the value for K in eq. 2.146. Comparison of a theoretical curve calculated with eq. 2.149 to a simulated powder diffraction peak using the Debye scattering equation (fig. 2.19) confirms the theoretical considerations and shows that this peak shape is well suited to estimate sizes from nanoparticles. Peak profiles for different particle shapes are shown in Leonardi et al.[136]

Instrumental Resolution The instrumental resolution of a powder diffractometer can be determined by the use of a standard sample, e.g. LaB_6 or CeO_2, that would theoretically produce infinitely sharp diffraction peaks, which get smeared out due to a convolution of the real peak shape with the instrumental resolution profile. This leads to peak broadening in the measured diffraction patterns, which can be significant for laboratory X-ray sources and need to be properly considered. However, for synchrotron sources this peak broadening contribution is usually orders of magnitude smaller than the peak broadening observed from the finite size of nanoparticles and can in most cases be neglected. For the symmetric peaks from standard samples a description by a Voigt function is adequate, which is given by the convolution of a Gaussian (G) and a Lorentzian (L) peak profile function

$$V(x,\sigma,\gamma) = \int G(\tau)L(x-\tau)d\tau \tag{2.152}$$

$$G(x,\sigma) = \frac{e^{-x^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \tag{2.153}$$

$$L(x,\gamma) = \frac{\gamma}{\pi(x^2+\gamma^2)}, \tag{2.154}$$

where σ is the standard deviation of the Gaussian contribution and γ corresponds to the width of the Lorentzian function. The obtained peak shape from the standard sample is then convoluted with the size broadening component.[133] For inclusion in the whole powder pattern modeling approach (eq. 2.145) the normalized Fourier transform is useful, given by[137]

$$A_{\mathrm{IP}}(\sigma,\eta,L) = (1-k)\exp\left(\frac{-\pi^2\sigma^2L^2}{\ln 2}\right) + k\exp(-2\pi\sigma L), \tag{2.155}$$

where

$$k = \frac{1}{1+[1/(\pi\ln 2)^{1/2}(1-\eta)/\eta]}. \tag{2.156}$$

The parameter η describes the relative contributions of the Gaussian and the Lorentzian functions to the peak shape and σ relates to the peak broadening due to the instrumental resolution.

2.3.5.4 Pair distribution function

The powder Pair Distribution Function (PDF) can be calculated from an experimental powder diffraction pattern and basically shows the frequency of interatomic distances as a function of the distances. It provides access to the local structure and is especially useful in amorphous, disordered crystalline or nanosized materials.[138–140]

To calculate the PDF, first the experimental pattern has to be normalized to the average atomic form factor and corrected for a possibly present background. It should be noted that the background is strictly only the contribution of e.g. the sample holder or other instrumentation related factors, which should also be experimentally determined. Other background contributions from e.g. diffuse scattering due to structural disorder must not be subtracted as the entire point of PDF analysis is the inclusion of these effects in the total scattering approach.

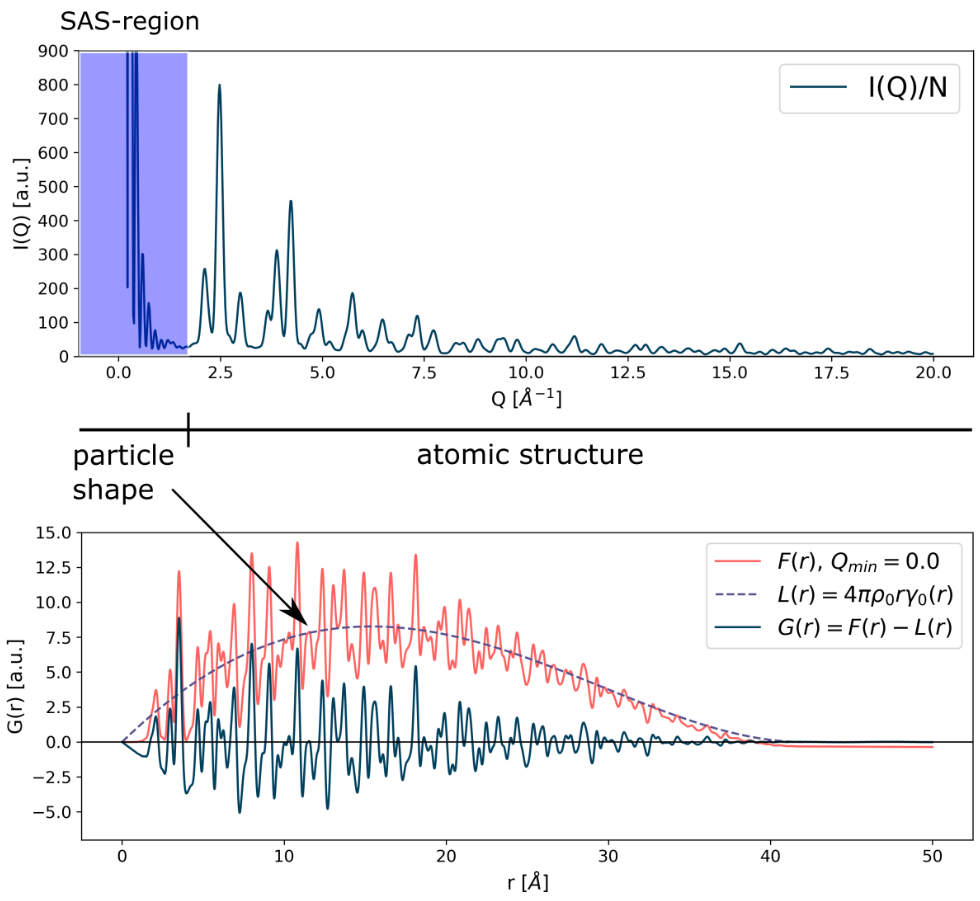


Fig. 2.20 At the top a calculated powder diffraction pattern is shown from a spherical iron oxide nanoparticle with diameter of 42 Å. The small-angle scattering is highlighted in blue. Below the calculated reduced pair distribution function $G(r)$ is shown. If the SAS region is included in the calculation a background due to the particle shape function is visible. If Q_{min} is larger than the SAS region this contribution is not visible. At the particle diameter the reduced PDF converges to zero.

The equations used for PDF analysis can be derived from the Debye scattering equation as given in eq. 2.85 as it represents the scattering from a not necessarily periodic structure and therefore inherently also considers diffuse scattering and contributions originating from disorder. The basis of pair distribution function analysis is the *total scattering function* $S(Q)$, which is defined through the coherent intensity, I_c, obtained from the Debye scattering equation or from an experiment after appropriate corrections, as

$$S(Q) = 1 + \frac{I_c(Q) - N\langle f^2\rangle}{N\langle f\rangle^2} = 1 + \frac{1}{N\langle f\rangle^2}\sum_{m\neq n} f_n f_m \frac{\sin(Qr_{mn})}{Qr_{mn}}, \tag{2.157}$$

where angle brackets denote the compositional averages of the atomic form factors and N is the number of scatterers. The function $S(Q)$ represents a normalized function of the scattered intensity in units of scattering per atom, which oscillates around and asymptotically approaches 1 for large Q. The corrections needed for experimental data have to be either known or determined by an ad hoc fitting procedure, as done in the program PDFgetX3.[141] A derived function $F(Q)$, called reduced structure function, is introduced that emphasizes the intensity in the high Q region and oscillates around zero according to

$$F(Q) = Q\,[S(Q) - 1] = \frac{1}{N\langle f\rangle^2} \sum_{m\neq n} f_n f_m \frac{\sin(Qr_{mn})}{r_{mn}}. \tag{2.158}$$

The reduced structure function , $F(Q)$, is Fourier transformed to yield a function of the interatomic distances, $F(r)$. Due to the sin function in the numerator $F(Q)$ is an odd function and therefore the Fourier transform is given by a sine transform

$$F(r) = \frac{2}{\pi} \int\limits_0^\infty F(Q)\sin(Qr)dQ = \frac{R(r)}{r} = 4\pi r\rho(r), \tag{2.159}$$

where $R(r)$ is the radial distribution function, that is defined as

$$\int\limits_a^b R(r)dr = N_{ab}, \tag{2.160}$$

i.e. the integral over a certain interatomic distance interval returns the number of atomic pairs that are separated by this distance. The function $\rho(r)$ is the atom-pair density function.[111]

If a PDF is calculated from simulated data that extends to $Q_{min} = 0$ a background due to the small-angle scattering (SAS) is visible (fig. 2.20). In real measurements Q_{min} is usually larger than the Q-range of SAS, thus this contribution is usually not found and is introduced as a correction factor in the definition of the reduced pair distribution function $G(r)$. For $G(r)$ calculated from experimental data Q_{max} is also some finite value thus

$$G(r) = \frac{2}{\pi} \int\limits_{Q_{min}}^{Q_{max}} F(Q)\sin(Qr)dQ = 4\pi r\rho(r) - L(r), \tag{2.161}$$

where the function $L(r)$ describes the small-angle scattering contribution. For an infinite sample with uniform number density, ρ_0, the contribution $L(r)$ is equal to $4\pi r\rho_0$. Nanoparticles can of course not be regarded as infinite and the small-angle contribution depends on the autocorrelation function of the shape function, $\gamma_0(r)$, which is given in eq. 2.147 for a sphere. In this case

$$G(r) = 4\pi r\rho(r) - 4\pi r\rho_0\gamma_0(r) = 4\pi r[\rho(r) - \rho_0\gamma_0(r)]. \tag{2.162}$$

This leads to the behaviour observed in fig. 2.20, where for small r the initial slope is observed according to $-4\pi r\rho_0$ and for large r the function approaches zero where the maximum size of the particle is reached. Applying the approximation $\rho(r) = \gamma_0(r)\rho_{bulk}(r)$, i.e. assuming that the particle is a piece cut from a bulk structure, which is the approach utilized in PDFgui,[142] $G(r)$ takes the form

$$G(r) = 4\pi r\gamma_0(r)[\rho_{bulk}(r) - \rho_0] = 4\pi r\gamma_0(r)\rho_0[g(r) - 1], \tag{2.163}$$

where $g(r) = \rho(r)_{bulk}/\rho_0$ is the pair distribution function, which approaches the average number density at large r and is zero for r smaller than the smallest interatomic distance in the structure. As such $g(r)$ is closely related to the sample structure, however, $G(r)$ is more commonly used as it can be directly computed from the corrected experimental data. Additionally, uncertainties in the measured data are constant in r for $G(r)$, which allows for more straight forward fitting procedures.[111] It should be noted that $G(r)$ is also frequently called pair distribution function.

Finally, artefacts originating from finite limits for Q have to be considered. A finite value for the upper Q limit results in termination ripples of the Fourier transform. This effect can be reduced by application of a damping function. However, it is generally advised to measure up to large enough values of Q where the intensity naturally approaches zero and the resulting termination ripples are not as strong. Furthermore, the high Q peaks contain important information about the local atomic arrangement. Termination ripples occur also due to low statistics in the high Q-range. This results in the need for synchrotron radiation to obtain reliable PDFs.[111]

2.4 Transmission electron microscopy

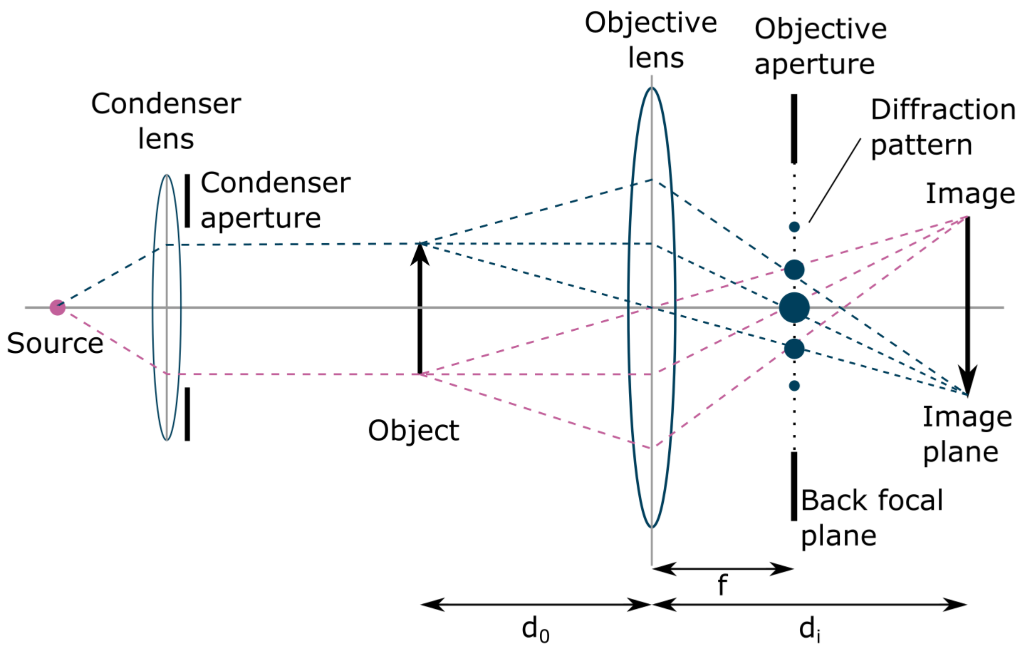


Fig. 2.21 Schematic ray diagram illustrating the image formation in a TEM. The rays from the sample are focused by the lens such that they converge to points on the image plane, thus forming the image. All parallel rays are focused in the back focal plane, leading to a diffraction pattern. The arrows at the bottom refer to the distance between the object and the lens (d_o), the distance between the lense and the image (d_i) and the distance between the lens and the back focal plane (f). Modified after Williams et al.[143] and Hawkes[144].

In this section the basic principles of transmission electron microscopy (TEM) and high resolution TEM (HRTEM) are presented. This discussion is based on the work of D.B. Williams and C.B. Carter[143] unless otherwise indicated.

The basic image formation process in transmission electron microscopy (TEM) is sketched in fig. 2.21. An electron beam is generated by the electron gun, where electrons are accelerated. In the condenser system comprised of usually two or more condenser lenses the convergence of the electron beam can be adjusted. For lower magnification typically a parallel beam is used to illuminate a large area of the sample, while for higher magnification a larger convergence is used thus illuminating a smaller area. Rays emerging from the sample are focused by the lens to points on the image plane, thereby forming the image. Magnetic lenses are used in TEM to focus the electrons. In addition, all parallel rays, i.e. rays with the same scattering angle are focused in points on the back focal plane (BFP), which creates a diffraction pattern. The object to lens (d_o), lens to BFP (f) and lens to image (d_i) distances are combined in the thin-lens equation

$$\frac{1}{f} = \frac{1}{d_o} + \frac{1}{d_i}. \tag{2.164}$$

The magnification of a convex lens is given as[145]

$$M = \frac{d_i}{d_o} = \frac{f}{d_o - f} = \frac{d_i - f}{f}, \tag{2.165}$$

thus higher magnification can be reached by either changing the focal length f of the lens or by changing the distance between the object and the lens d_o. Usually additional lenses are used after the objective lens. By adjusting the strength of these lenses either the image is projected on the screen, which is called *imaging mode* or the diffraction pattern is projected on the screen, which is called *diffraction mode*. In imaging mode the objective aperture can be used to select the beams that contribute to the image formation. If only the direct beam is used a so-called *bright field* (BF) image is formed. If instead only diffracted beams are used a *dark field* (DF) image is produced. In TEM and high resolution TEM (HRTEM) of crystalline samples usually no objective aperture is used thus allowing both the direct and the diffracted beams to form the image.

In transmission electron microscopy the observed image contrast is due to the differences in the scattering of the electron beam by different parts of the sample. Two types of contrast may be distinguished, namely mass-thickness contrast and diffraction contrast. If more than one diffracted electron beam is allowed to contribute to the image formation via opening of the objective aperture also phase contrast can be observed, that is due to the interference of the diffracted electron waves. Phase contrast can be viewed as a type of diffraction contrast. In the experimental work for this thesis only the multiple beam mode was used thus mass-thickness, diffraction and phase contrast contribute to the image.

For non-crystalline materials generally mass-thickness contrast is most important as the periodic arrangement of atoms is lacking that produces significant diffraction or phase contrast.[145] This type of contrast can be understood by incoherent elastic scattering of electrons from the nuclei in the sample. Elements with a higher atomic number Z have larger cross section for elastic scattering and thus will scatter the electrons more strongly. With the use of an objective aperture centered around the primary beam this leads to the appearance of darker areas, where more high-Z elements are present. Similar for thicker areas of the sample more elastic scattering will take place and therefore again these regions appear darker. The two effects combined are called mass-thickness contrast. In the samples used in this thesis mass-thickness contrast results in the visibility of the inorganic particle cores against the amorphous carbon background. However, the organic shell is invisible as it also mainly contains carbon and is thin compared to the particle core thus no contrast contribution from the shell can be seen. For crystalline nanoparticle samples mass-thickness contrast would suggest that all particles show similar contrast to the surrounding amorphous substrate. However, in experimental images it can clearly be seen that some particles appear darker than others. This can be explained by the additional diffraction contrast due to a certain orientation of particles i.e. certain orientation of crystal lattice planes, where Bragg diffraction of the electron beam is stronger and thus the particle appears darker.[145] Due to the mass-thickness and diffraction contrast in low magnification TEM the inorganic particle size can be determined by measuring the diameter of the dark areas of the recorded image.

Despite the common association of diffraction or phase contrast with high resolution TEM (HRTEM) it can also be observed in lower magnification TEM images if a large enough objective aperture is used and the image is recorded in defocus. Phase contrast is the origin of so-called lattice fringes, which can only occur via the interference of electron beams. If the Bragg condition for a set of planes is fulfilled a peak will form in the back focal plane. If the condition is satisfied for multiple lattice planes a diffraction pattern forms with several spots (fig. 2.21). The interference of a diffracted beam with the direct beam leads to lattice fringes, wich are periodically varying intensities in the detected image. The distance between intensity maxima of the lattice fringes is equal to the lattice plane spacing of the lattice planes that resulted in the diffracted beam. However, defocus as well as the sample thickness affect the position of the fringes thus they generally do not coincide with the real lattice planes.[146] The periodicity and the orientation can still provide information on the crystal structure of the specimen. Multiple lattice fringes originating from the simultaneous fulfillment of the Bragg condition for several sets of lattice planes can also cross each other leading to bright and dark spots in the recorded TEM image that can in some case correspond to individual atomic columns. However, the real position of the atoms can usually only be inferred from image simulations.[147]

2.5 Mössbauer spectroscopy

Mössbauer spectroscopy is based on a recoilless nuclear resonant absorption of γ-quanta by isotopes.[148] The Mössbauer effect, named after the discoverer R. L. Mössbauer, can be used to analyse the chemical, structural, magnetic and thermodynamic properties of materials. In this work Mössbauer spectroscopy is used to distinguish magnetite from maghemite and quantify the phase fractions via determination of the Fe^{2+} contribution to the spectrum.

An important process for this technique is the radioactive decay of nuclides that produce daughter nuclei. These nuclei are in an excited state that reach a stable ground state by emitting γ-rays. The γ-rays in turn can excite ground-state nuclei of the same isotopes. This effect is called nuclear resonant absorption. However, two processes prevent this mechanism from taking place. The emission of the γ-ray results in a recoil of the emitting nucleus thus lowering the γ-ray energy. Secondly the thermal motion of nuclei produces a Doppler effect via the movement of emitting and absorbing nuclei towards and away from each other. This leads to a broadening of the energy distribution of the emitted γ-rays. A solution to this problem was proposed by R. L. Mössbauer. Here, the emitting nucleus is placed within a crystal, where the chemical bond energy prevents the recoil of this individual nucleus and instead the crystal recoils either as a whole or by excitation of phonons. Since the mass of the crystal is much larger than that of the isolated nucleus both the recoil energy and the Doppler broadening become very small. If no phonons are excited the emission of γ-rays can be viewed as recoil-free. The fraction of γ-ray emission occuring without recoil depends on the temperature and the specific transition energy of the source material. For ^{57}Fe this fraction is 0.91 at low temperatures.[149]

The intensity of γ-rays is measured after transmission through the sample. If nuclei of the same isotope as in the source are present in the sample, resonant absorption occurs and the detected γ-ray intensity is reduced. Introducing also a relative motion of the source and the sample allows a fine-tuning of the γ-ray energy by the Doppler effect. Thus, recording the transmission as a function of the relative velocity yields an absorption spectrum.[148]

To study Fe-containing samples the ^{57}Co source is especially suited as it decays to ^{57}Fe in excited nuclear states. By de-excitation to the ground-state these produce γ-rays with the right energy to excite iron nuclei in the sample. The above described utilization of the Doppler effect can be used to study hyperfine interactions of the iron nuclei. Most important for this work are the isomer shift, the quadrupole splitting and the magnetic splitting of energy levels (fig. 2.22). The isomer shift is a consequence of the Coulomb interactions between electrons and the nuclear charges. Thus the different charge densities at the core of Fe^{2+} and Fe^{3+} cations lead to a difference in the observed isomer shift. An isomer shift is observed for both cations as the excited ^{57}Fe has a slightly larger nucleus and therefore the emitted γ-ray has a slightly different energy than the absorption energies of both cations. Quadrupole splitting is a result of the local crystallographic environment, leading to a doublet in the transmission spectrum caused by

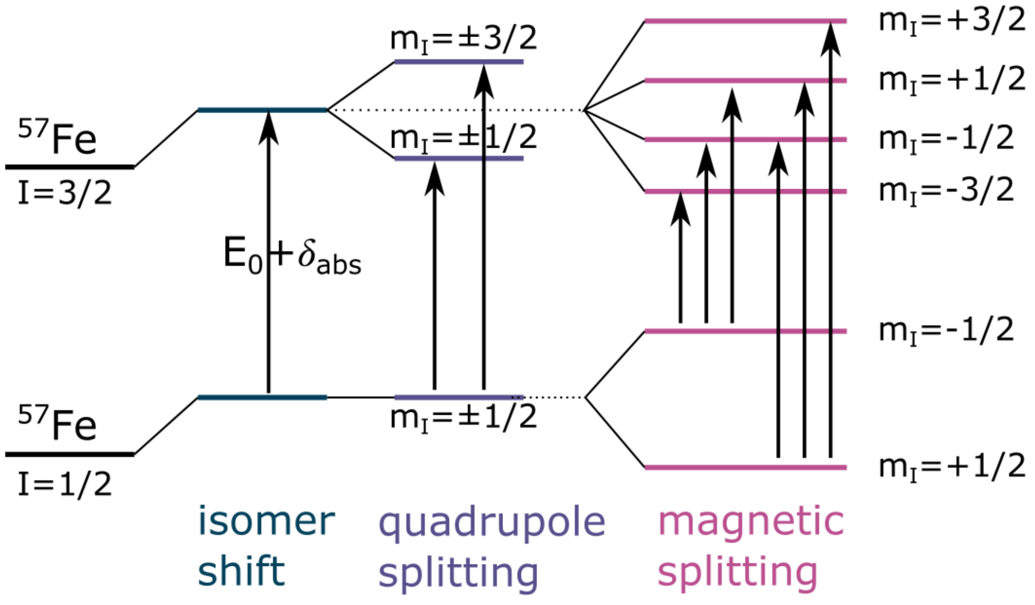


Fig. 2.22 Nuclear energy levels of ^{57}Fe. The effects of isomer shift, quadrupole splitting and magnetic splitting are shown from left to right. The arrows indicate the transition due to absorption of γ-rays. Figure adapted from Blundell.[53]

the splitting of the excited energy level. Magnetic splitting is caused by a local magnetic field where the ground state is split into a doublet and the excited state into a quadruplet thereby resulting in six absorption lines, i.e. a sextet.[53]

In the iron oxide samples of this work these effects are superimposed, i.e. there are sextets corresponding to the magnetic hyperfine splitting of both iron cations. These sextets are isomer shifted due to the different charge densities at the nucleus. Additionally, the hyperfine magnetic fields are lower for Fe^{2+} ions.[150] Different local environments such as octahedral or tetrahedral iron positions lead to a difference in the hyperfine fields as well, however for maghemite and magnetite this difference is very small.[151] Quadrupole splitting leads to a slight asymmetry in the sextets, however again for maghemite and magnetite this effect is small.[151] It should also be noted that the magnetic splitting in nanoparticles is only observed below the blocking temperature, as above this temperature the superparamagnetic relaxation smears out the observed spectrum leading to a doublet spectrum.[152] Considering these effects in a fit to the measured Mössbauer spectra an estimate of the magnetite content can be made on the basis of the relative contribution of the Fe^{2+} sextet to the total spectrum.

2.6 Iron oxides

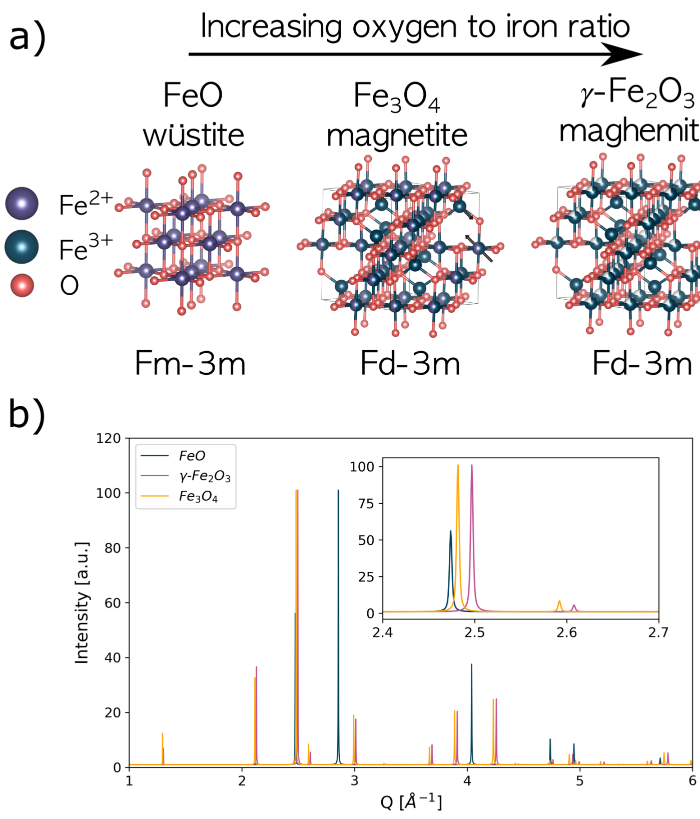


Fig. 2.23 a) Unit cells for the most important iron oxides in the context of iron oxide nanoparticles. With increasing ratio of oxygen-to-iron, the structure changes from FeO (wüstite) over Fe_3O_4 (magnetite) to γ-Fe_2O_3 (maghemite). Through this transition the oxygen lattice is almost not affected but the overall symmetry changes due to different placement of the iron cations. Additionally, the ratio of Fe^{2+} to Fe^{3+} decreases from 0.5 to 0 with increasing oxygen-to-iron ratio. In the maghemite structure no Fe^{2+} cations are left and instead vacancies are formed mainly on the octahedral sites (indicated as white wedges on the atom balls). The symmetry of maghemite can be further reduced by vacancy ordering leading to a tetragonal space group. b) Calculated powder diffraction patterns from the ideal crystal structure models as given in the text. Patterns were calculated using Vesta.[153] The different lattice parameters of the iron oxides lead to a shift in the peak positions. Here $Q = 4\pi/\lambda \sin\theta$ was used.

The three most important iron oxides to be considered for this work are wüstite ($Fe_{1-x}O$), magnetite (Fe_3O_4) and maghemite (γ-Fe_2O_3). This is due to the fact that during the particle synthesis via the thermal decomposition of an iron precursor in a boiling organic solvent, oxidation in the sense of an increasing oxygen-to-iron ratio of particles takes place from wüstite over magnetite to maghemite (fig. 2.23).[46] This is possible, since the crystal structures of all three iron oxides are rather similar and can be transformed into each other by diffusion of Fe cations and adjustments in the oxygen distances and inter-layer spacing.[154] Despite their

Tab. 2.2 Ideal crystal structure data of wüstite in space group $Fm\bar{3}m$. The multiplicity of the sites is given under "mult.", the site occupancy is abbreviated by "occ.".

label	mult.	x/a	y/a	z/a	occ.	ion
Fe(oct)	4	0.0	0.0	0.0	1.0	Fe^{2+}
O	4	0.5	0.5	0.5	1.0	O^{2-}

similarities these iron oxides exhibit significant differences in structural features as well as their magnetic properties, which will be analysed in this chapter.

2.6.1 Wüstite

The iron oxide wüstite (FeO) crystallizes in the cubic space group $Fm\bar{3}m$ with a lattice parameter of 4.28 to 4.31 Å (fig. 2.23 and tab. 2.2).[154] This is slightly larger than half of the lattice parameter of maghemite and magnetite. The structure is constructed of cubic close-packed octahedrally coordinated iron atoms and can be described by two face-centred lattices, one for Fe^{2+} ions and one for O^{2-}. Transformation to magnetite is relatively easy since only minor rearrangements in the iron sub-lattice are necessary, which lead to a slight reduction in the oxygen distances as well as the layer spacing. The magnetic structure of wüstite is that of an antiferromagnet with alternating orientation of magnetic moments between (111) planes that cancel each other. The Néel temperature lies in the range of 203 to 211 K.[154] Above this temperature wüstite is paramagnetic. Due to this low transition temperature it is possible to measure an exchange bias effect when cooling iron oxide particles containing both wüstite and magnetite from room temperature to temperatures below T_N in an applied field.[155] However, the exchange bias effect observed for a nanoparticle sample alone is not enough to prove the presence of wüstite in the structure. Further insights may be provided by X-ray powder diffraction as the Bragg peak positions of this phase are very different compared to the ones obtained from maghemite and magnetite (fig. 2.23b)). Additionally, Mössbauer spectroscopy can provide information on the presence of wüstite (see section 2.5).

2.6.2 Magnetite and maghemite

Magnetite crystallizes in the inverse spinel structure with a cubic unit cell with space group $Fd\bar{3}m$ and lattice constant $a = 0.839\,67(3)$ nm that contains 8 formula units, i.e. 32 oxygen atoms, 8 Fe^{2+} and 16 Fe^{3+} ions.[165] The oxygen anions form a cubic close packing lattice, where iron cations are placed in the interstices. Half of the trivalent iron ions are placed in oxygen tetrahedra (A-sites). The other half and the 8 divalent ions are distributed randomly within oxygen octahedra (B-sites). A-site ions have 4 nearest neighbours on A-positions and 12 on B-positions. B-site ions have 6 nearest neighbours on A-positions and 6 on B-positions. The crystal structure data is given in tab. 2.4.

Tab. 2.3 Material properties of magnetite and maghemite. a is the cubic lattice parameter, T_C the ferrimagnetic Curie temperature, K_1 the magnetocrystalline anisotropy constant. The exchange constants are J_{AA}, J_{AB} and J_{BB}, where the subscript indicates the involved lattice sites. $M_{sat.}$ is the saturation magnetization. The critical diameter below which the single domain state is favourable is given as D_{sd}.

Parameter	magnetite	maghemite
a	$0.839\,67(3)\,\mathrm{nm}$[156]	$0.834\,57\,\mathrm{nm}$[157]
T_C	$856\,\mathrm{K}$[158]	893 to $918\,\mathrm{K}$[159,160]
K_1 (nano)	$-27\,\mathrm{kJ}$[64]	-27 to $-19\,\mathrm{kJ}$ ($15\,\mathrm{nm}$)[84,150]
K_1 (bulk)	$-13.5\,\mathrm{kJ}$ (above T_V)[64]	$-4.7\,\mathrm{kJ}$[161]
J_{AA}/k_B	$-1.3\,\mathrm{K}$[61]; $-18\,\mathrm{K}$[154]	$-21.0\,\mathrm{K}$[60]
J_{BB}/k_B	$7.3\,\mathrm{K}$[61]; $3\,\mathrm{K}$[154]	$-8.6\,\mathrm{K}$[60]
J_{AB}/k_B	$-33.9\,\mathrm{K}$[61]; $-28\,\mathrm{K}$[154]	$-28.0\,\mathrm{K}$[60]
$M_{sat.}$ ($0\,\mathrm{K}$)	$96\,\mathrm{Am^2/kg}$[162]	$87\,\mathrm{Am^2/kg}$[162]
$M_{sat.}$ (RT)	$86\,\mathrm{Am^2/kg}$[163]	$76\,\mathrm{Am^2/kg}$[164]
D_{sd}	50 to $128\,\mathrm{nm}$[82,83]	40 to $166\,\mathrm{nm}$[27,82]
ρ	$5.2\,\mathrm{g/cm^3}$	$4.9\,\mathrm{g/cm^3}$[159]

Tab. 2.4 Ideal crystal structure data of magnetite in space group $Fd\bar{3}m$. The multiplicity of the sites is given under "mult.", the site occupancy is abbreviated by "occ.". Tetrahedral and octahedral iron positions are labelled with $Fe(tet)$ and $Fe(oct)$, respectively.

label	mult.	x/a	y/a	z/a	occ.	ion
Fe(tet)	8	0.125	0.125	0.125	1.0	Fe^{3+}
Fe(oct)	16	0.5	0.5	0.5	1.0	Fe^{2+}/Fe^{3+}
O	32	0.25	0.25	0.25	1.0	O^{2-}

The octahedral iron cations interact via double-exchange (between Fe^{2+} and Fe^{3+}, see section 2.1.2.4) and super-exchange (between Fe^{3+} and Fe^{3+}, see section 2.1.2.3) leading to an effective nearest neighbour exchange constant of $J_{BB}/k_B = 7.3\,\mathrm{K}$, i.e. ferromagnetic alignment. The exchange constant for the tetrahedral sub-lattice is negative with $J_{AA}/k_B = -1.3\,\mathrm{K}$ thus favouring antiparallel alignment of the spins. However, the super-exchange interaction between octahedral and tetrahedral iron is strongly antiferromagnetic as well ($J_{AB}/k_B = -33.9\,\mathrm{K}$) giving rise to antiparallel alignment of the sublattices. This acts against the antiferromagnetic alignment between $A-site$ ions resulting in a ferromagnetic structure in the tetrahedral sub-lattice. Hence, the magnetic structure of magnetite in the ground state can be described by two sub-lattices that are antiferromagnetically aligned. In the ideal structure the magnetic moments of the trivalent iron cations in each sub-lattice cancel each other exactly and the net magnetic moment of the unit cell results only from the Fe^{2+} ions. The ferrimagnetic Curie temperature is given as $\approx 870\,\mathrm{K}$.[166] At 120 K magnetite undergoes a phase transition called Verwey transition, that is in a first approximation due to charge ordering of the delocalized electron from the Fe^{2+} ions.[167,168] Below this transition magnetite becomes electrically insulating and the crystal symmetry changes to a monoclinic one.[168] In a $M(T)$ measurement this transition is also characterized by a sudden drop in magnetization upon cooling the sample. This distinct feature is taken as evidence for the presence of magnetite in a sample, however, the opposite is not necessarily true as it was observed that the Verwey transition may be suppressed or at least shifted towards low temperatures in nanomaterials.[50,64,169] For bulk magnetite above the Verwey transition temperature T_V the magnetocrystalline anisotropy constant is negative thus resulting in an easy and hard axis along $\langle 111 \rangle$ and $\langle 100 \rangle$, respectively. Below T_V a structural change leads to a uniaxial anisotropy with the easy axis along $\langle 001 \rangle$. However, as noted above this case may not be realized even at 5 K for nanoparticles due to their low T_V.[64]

Magnetite forms solid solutions with the isostructural maghemite. The latter differs in the structure in that it contains vacancies on octahedral sites and all other cation sites are occupied only by trivalent iron. The transition between both structures is gradual, which makes a distinction of both phases even harder. Maghemite is usually found to have slightly smaller lattice parameters. On the basis of this observation Cervellino et al.[157] proposed a law to determine the cubic lattice parameter from the particle composition and size according to

$$a = [(1-x)a_{\text{maghemite}} + x a_{\text{magnetite}}]\left(1 - \frac{\Omega}{D}\right), \qquad (2.166)$$

where the term in square brackets is Vegard's law with $a_{\text{maghemite}}$ and $a_{\text{magnetite}}$ the lattice parameters of maghemite and magnetite, respectively, given in tab. 2.3. The correction factor in the round brackets accounts for the influence of the particle size on the lattice parameter. The parameter $\Omega = \frac{4\gamma}{3B} = -2.05(21) \times 10^{-3}\,\mathrm{nm}$ is related to the bulk modulus B and the surface tension γ. However, this law does not account for possible deviations from the cubic symmetry as discussed

Tab. 2.5 Ideal crystal structure data of maghemite in the various observed space groups related to different degrees of vacancy ordering. The multiplicity of the sites is given under "mult.", the site occupancy is abbreviated by "occ.". Tetrahedral and octahedral iron positions are labelled with $Fe(tet)$ and $Fe(oct)$, respectively.

label	mult.	x/a	y/a	z/a	occ.
Fd-3m					
Fe1(tet.)	8	0.125	0.125	0.125	1.0
Fe2(oct.)	16	0.5	0.5	0.5	0.83
O1	32	0.25	0.25	0.25	1
P4$_3$32					
Fe1(tet.)	8	0.25	1.0	0.625	1.0
Fe2(oct.)	4	0.875	0.625	0.25	0.33
Fe3(oct.)	12	0.375	0.375	0.50	1.0
O1	8	0.125	0.875	0.50	1
O2	24	0.375	0.125	0.0	1
P4$_3$2$_1$2					
Fe1(tet.)	8	0.75	1.0	0.125	1.0
Fe2(oct.)	4	0.375	0.625	0.75	0.33
Fe3(oct.)	4	0.125	0.875	0.25	1.0
Fe4(oct.)	8	0.375	0.875	1.0	1.0
O1	8	0.125	0.375	0.5	1
O2	8	0.375	0.125	0.0	1
O3	8	0.125	0.875	0.0	1
O4	8	0.375	0.625	1.0	1
P4$_1$2$_1$2					
Fe1(tet.)	8	0.750	1.000	0.0420	1.0
Fe2(tet.)	8	0.750	1.000	0.7083	1.0
Fe3(tet.)	8	0.750	1.000	0.3750	1.0
Fe4(oct.)	4	0.625	0.625	0.0000	1.0
Fe5(oct.)	8	0.125	0.875	0.0830	1.0
Fe6(oct.)	8	0.375	0.875	1.0000	1.0
Fe7(oct.)	8	0.375	0.875	0.3330	1.0
Fe8(oct.)	8	0.375	0.875	0.6670	0.0
Fe9(oct.)	4	0.375	0.625	0.2500	1.0
Fe10(oct.)	8	0.625	0.375	0.0830	1.0
O1	8	0.125	0.375	0.1667	1
O2	8	0.125	0.375	0.5000	1
O3	8	0.125	0.375	0.8330	1
O4	8	0.375	0.125	0.0000	1
O5	8	0.375	0.125	0.3330	1
O6	8	0.375	0.125	0.6660	1
O7	8	0.125	0.875	0.0000	1
O8	8	0.125	0.875	0.3330	1
O9	8	0.125	0.875	0.6670	1
O10	8	0.375	0.625	0.0000	1
O11	8	0.375	0.625	0.3330	1
O12	8	0.375	0.625	0.6670	1

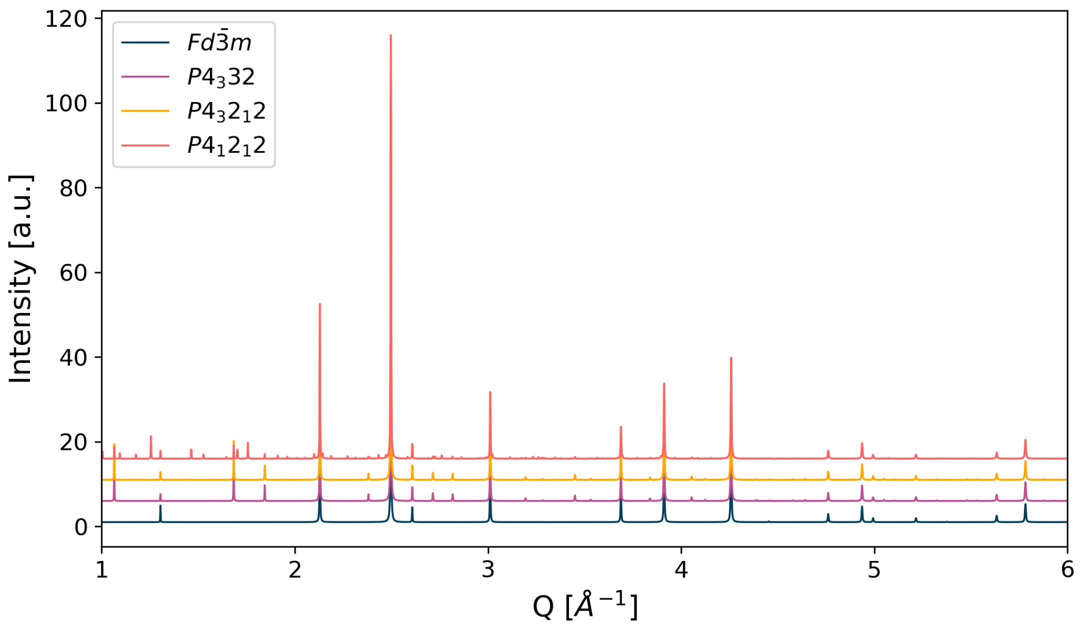


Fig. 2.24 Simulated powder diffraction patterns of the maghemite structures with different degrees of vacancy ordering but same total occupancy of the iron positions. The vacancy ordering leads to a decrease in symmetry from cubic space groups to tetragonal with an increase in small superstructure peaks between the major inverse spinel peaks.

in the following. For iron oxide nanoparticles containing a mixture of maghemite and magnetite it was found that powder diffraction patterns and PDFs are best described by a non-stoichiometric intermediate composition that conforms to the model of a gradual transition from a more oxidized surface region towards a more iron rich core region without a distinct separation.[50]

Different space groups for maghemite are reported in the literature, which can be attributed to different degrees of vacancy ordering. Kelm et al. [170] constructed a continuous group-subgroup relation between disordered cubic maghemite/magnetite with the space group $Fd\bar{3}m$ down to the tetragonal structure with symmetry $\mathrm{P4_12_12}$. Group theoretical considerations leave only the group-subgroup chain of $Fd\bar{3}m$ - $F4_132$ - $P4_332$ ($P4_132$) - $P4_32_12$ ($P4_12_12$) - $P4_12_12$ ($P4_32_12$).[170] The last step involves a tripling of the unit cell along the c direction to describe the periodicity of the vacancy ordering. In parentheses are the enantiomorphic space groups, that differ only in the handedness of the screw axis. The cubic structure with space group $Fd\bar{3}m$ contains only two independent iron positions, for tetrahedral and octahedral iron. The octahedral iron position is then split into an increasing number of independent positions due to the decrease in symmetry. Simulated powder diffraction patterns for the different space groups are shown in fig. 2.24, where the emergence of the vacancy ordering superstructure peaks is clearly visible. Space groups $P4_332$ and $P4_32_12$ do not differ much if the ideal crystal structures are considered, however the latter provides more degrees of freedom in the atomic positions due to the decrease in symmetry from the cubic to the tetragonal system. Generally, vacancies seem to form preferably on the octahedral lattice sites.[157] This may be explained by considering the oxidation of Fe^{2+} ions to Fe^{3+} during the transition from magnetite to maghemite via the migration of mobile electrons away from this site. The Fe^{2+} ions are only found on the octahedral iron sites due to their larger ionic radius. The remaining Fe^{3+} leads to a charge imbalance that can be compensated by removing the excess Fe^{3+}. Finally, this leaves a vacancy on the octahedral site behind.[171] In reality the vacancy formation process is likely more complex as frequently a significant amount of vacancies is also found on tetrahedral lattice sites.[157]

The magnetic structure of maghemite can be described by two antiferromagnetically aligned iron sublattices, in which the magnetic moments are ferromagnetically aligned. Contrary to the magnetite structure only Fe^{3+} ions are present, however still ferrimagnetic ordering can be observed due to the vacancies on the octahedral lattice sites. The bulk saturation magnetization is lower than for magnetite. Moreover, there are no magnetic transitions known for the maghemite structure. The exchange constants between iron ions are given in tab. 2.3. Vacancy ordering has been found to increase the measured saturation magnetization of otherwise defect free maghemite nanoparticles up to the expected bulk values.[33] Additionally, in the same work increasing exchange bias effects with decreasing particle size were attributed to increasing vacancy disorder leading to a higher degree of spin frustration.

2.6.3 Antiphase boundaries

Antiphase boundaries (APBs) occur in many crystal structures and are commonly associated with metallic systems that exhibit disorder-oder transitions, such as the Cu_3Au alloy,[109,172–175] but can exist in all ordered phases.[176] These kinds of defects have also been detected in Fe_3O_4-thin films and were the subject of numerous works.[177–183] More recently it was recognized that APBs might be responsible in large parts for the anomalous magnetic properties of iron oxide nanoparticles.[43,46,47,84,158]

The formation of APBs in iron oxide nanoparticles is intimately linked to the synthesis method. All of the samples considered in this work were produced by the thermal decomposition of an iron oleate complex. In this synthesis the particles start out with wüstite composition, which gets oxidized to magnetite and finally maghemite. Dark-field transmission electron microscopy shows the nucleation of the spinel-phase on several locations at the particle surface. As the oxidation of the particle proceeds these structural and compositional subdomains will eventually meet, where at the interface the iron sublattices might not line up, but are displaced by some fraction of the unit cell parameter.[46] The oxygen sublattice, however, remains rather constant throughout this oxidation process, since all three iron oxide phases (wüstite, magnetite and maghemite) share in principle the same oxygen sublattice configuration differing only slightly in the oxygen-oxygen distances (fig. 2.23). Some configurations of iron sublattice displacements thus formed are especially stable as first principle calculations have found.[184] These are two

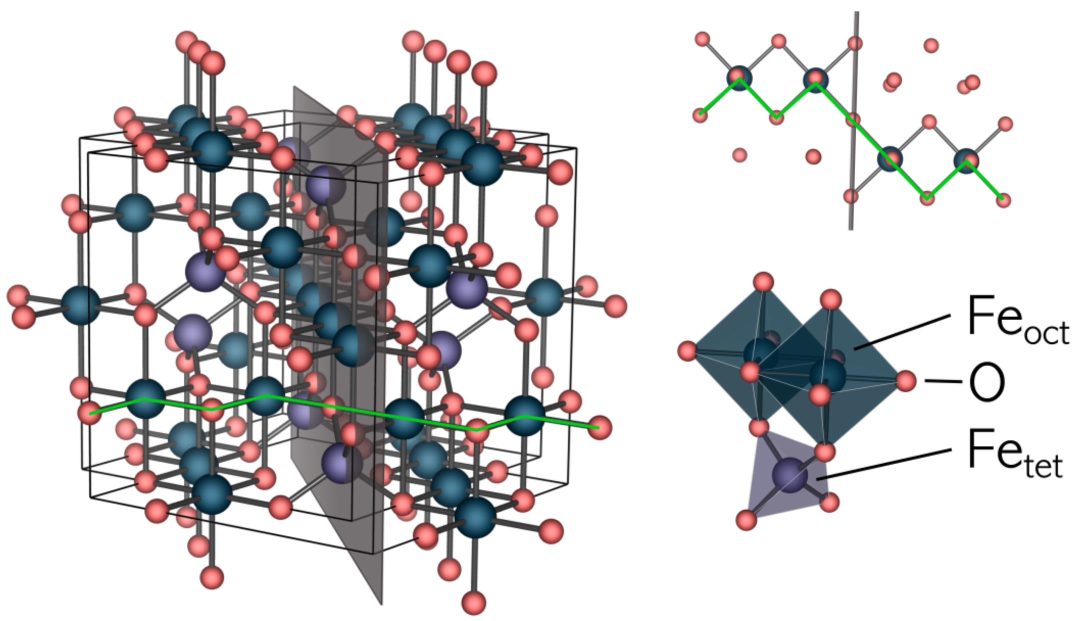


Fig. 2.25 Maghemite/magnetite cubic unit cell containing an $1/4a[110]$ APB indicated with the grey plane. At the APB the octahedral chains that are perpendicular to the APB plane are displaced leading to a $180°$ angle between octahedral iron atoms. This arrangement is also drawn on the top right.

kinds of displacements on $\{110\}$ planes, i.e. the cube diagonals, namely the translations $1/4a[110]$ (APB-I) and $1/4a[110] + 1/4a[1\bar{1}0]$ (APB-II). APB-I, depicted in fig. 2.25 has the lowest calculated formation energy of $102\,\mathrm{mJ/m^2}$. This is due to the small lattice distortions introduced by this shift, which are only present between octahedral and tetrahedral Fe sites at the interface. For APB-II the formation energy is much larger with $954\,\mathrm{mJ/m^2}$, thus the focus of this work will lie on type I APBs as they will be much more likely. Electron microscopy studies of magnetite thin films also seem to indicate the prevalence of type I APBs.[185,186] As can be seen from fig. 2.25 indicated by the green line in the 3D-structure as well as in the top view on the upper right the APB leads to disruption in the edge-sharing octahedral chains giving rise to a corner sharing configuration at the interface that is associated with an increase in the $Fe_B - O - Fe_B$ bond angle from $90°$ to $180°$. As discussed in section 2.1.2.3 according to the Goodenough-Kanamori rules this will lead to a change in the superexchange interaction from ferromagnetic to antiferromagnetic alignment of the magnetic moments. The antiferromagnetic exchange interaction has been experimentally confirmed by magnetotransport measurements across a single antiphase boundary in a magnetite thin film.[187] This abrupt change in the magnetic ordering will have an impact on the spin structure and subsequently the macroscopic magnetic properties of nanoparticles containing these defects. The precise nature of this impact as well as the macroscopic implications are the subject of this work. Regarding the morphology of APBs in materials W.L. Bragg made several observations and considerations already in 1940 in Cu_3Au alloys.[188] Calling the type of order on one side of an APB A and on the other side B he proposed that a region of order A that reaches into region B will become smaller as the atoms in A now have more neighbours in B and thus at the interface they rearrange into B-type order. This process should proceed until the APB is flat, i.e. no more regions of A extend into regions of B but both types of order coexist with a planar boundary between them. Additionally A-type regions surrounded entirely by B-type ordering will be converted into B-regions and thus vanish. After equilibrium is reached he supposed that the generated arrangement of APBs should be very stable. Eerenstein et al.[189] found that annealing of magnetite thin films leads to coarsening of the APB-domain sizes, suggesting the thermally activated migration of APBs. McKenna et al.[184] report straight APBs in Fe_3O_4 thin films. This migration is explained by the diffusion of iron cations through interstitial positions of the oxygen sublattice. In iron oxide nanoparticles the APBs are found primarily close to the particle center, but not entirely planar.[45,47] Considering the Bragg-model of APB development, an APB close to the surface would mean a smaller region of A close to the surface surrounded by region B. This would probably lead to the conversion of the A region to B. Additionally, close to the surface the structure is likely distorted and depleted of iron compared to the core possibly allowing faster diffusion and rearrangement of the Fe-lattice. It has to be noted, however, that the few published HRTEM images showing particles with APBs cannot provide statistical information on the placement and quantity of APBs in nanoparticles.

Chapter 3

Instruments and experimental techniques

3.1 Small-angle scattering

3.1.1 SAXS: GALAXI

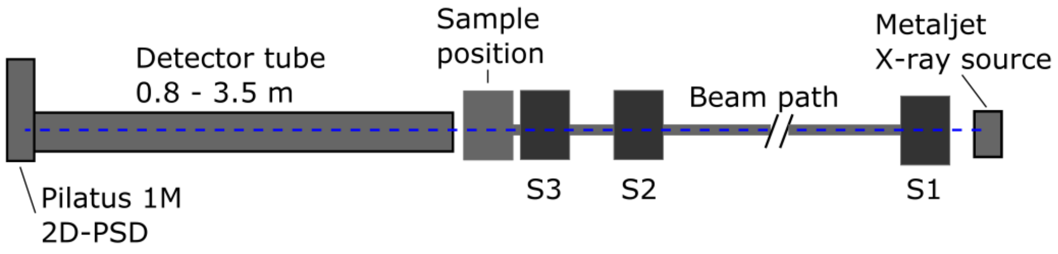


Fig. 3.1 Scheme of GALAXI, adapted from[190]. The X-ray beam generated by the liquid anode source is collimated by the slits labeled S1, S2 and S3. After passing through the sample capillary in the sample chamber the X-rays are recorded by the 2D position sensitive detector. The position of the detector can be adjusted between 0.8 and $3.5\,\mathrm{m}$ to collect signal in different Q-ranges. The beam path is fully evacuated.

Small-angle X-ray scattering (SAXS) experiments were performed on GALAXI (Gallium anode low-angle X-ray instrument) at Jülich Centre for Neutron Science (JCNS), Forschungszentrum Jülich GmbH[190]. GALAXI is operated with a GaInSn liquid anode producing Ga Kα radiation with 9243 eV photon energy (wavelength: $\lambda = 0.134\,\mathrm{nm}$), monochromatized by parabolic Montel-type optics. The wavelength spread for GALAXI of $\Delta\lambda/\lambda \approx 3 \times 10^{-3}$ (Ga Kα_1 and Ga Kα_2 are not resolved) is assumed to be small enough and is not considered in the data analysis.[119] The beam is collimated by two slits indicated with S1 and S2 in figure 3.1. The third slit labeled S3 in the figure is used to reduce the background. The data was recorded on a Pilatus 1M 2D position sensitive detector. The isotropic detector images were radially averaged using the fit2D software.[191] The scattering cross section per unit volume ($1/V\left(\frac{d\sigma}{d\Omega}\right)$) is obtained after correction for the empty cell and the solvent scattering contributions and normalization with the reference material FEP 1400 Å. The experiments were conducted at two detector distances (3535 and 835 mm) to cover the entire Q-range (4×10^{-2} to 8 nm^{-1}) available on the system. For the experiment borosilicate glass capillaries from Hilgenberg

GmbH with wall thickness of 0.05 mm and internal diameter of 2.0 mm[192] were filled with liquid nanoparticle dispersions and then sealed by melting the ends. This ensures no leakage of the sample in the evacuated sample chamber.

3.1.2 SANS: KWS-1

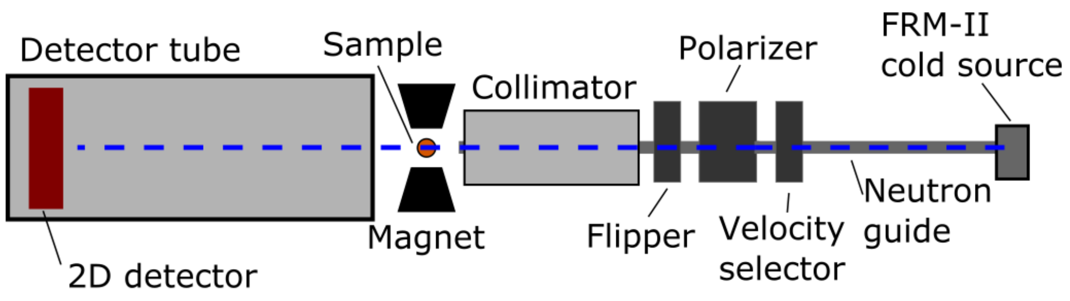


Fig. 3.2 Top view of the experimental setup for small-angle neutron scattering experiments at KWS-1.[193] The neutron flight path is symbolized by the dashed blue line. The magnetic field is applied at the sample position perpendicular to the neutron beam. The area detector is placed in an evacuated tube.

Small-angle neutron scattering with polarized neutrons (SANSPOL) was performed on KWS-1 operated by the Jülich Centre for Neutron Science at Heinz Maier-Leibnitz Zentrum (Garching, Germany)[193,194]. A sketch of the instrument is shown in fig. 3.2. Neutrons produced in the core of FRM II research reactor are moderated in the cold source and reach the instrument via a system of neutron guides. The mechanical velocity selector is used to select the mean wavelength of the neutrons with a spread of around 10 %. In this work neutrons with a wavelength of 4.9 Å were used. The transmission polarizer is used giving a spin down polarization of $P \approx 90.5\,\%$. To obtain the opposite direction in neutron polarization, i.e. spin up, the radio-frequency spin flipper is used, with the flipper efficiency of $\varepsilon = 0.998$. The collimation distances can be adjusted, but in this work for resolution purposes it was fixed at 8 m. At the sample position a horizontal magnetic field was applied perpendicular to the neutron beam. The scattered neutrons are detected by a 2D detector consisting of ^{3}He tubes placed inside the evacuated detector tube, which can be moved thereby adjusting the Q-range. For this work detector positions of 2.3 m and 8 m were used. The recorded data was normalized to absolute intensities by measuring a standard sample and the empty beam. Correction for the solvent contribution and the empty cell scattering was also performed. Sector averages in detector plane parallel and perpendicular to the applied field were performed for each polarization state in 10°-sectors to improve the statistics and to obtain the purely nuclear scattering contribution and the nuclear-magnetic interference terms (see section 2.3.4). Data reduction and fitting was carried out in QtiKWS and QtiSAS[195].

Diluted solutions of spherical nanoparticles were measured at 5 contrast conditions. As described in section 2.3.4.7 contrast variation experiments allow to study the nuclear inorganic core - organic shell structure of the nanoparticles used in this work. In addition, polarized neutrons serve to examine the magnetic structure from a simultaneous fit of the nuclear magnetic interference terms obtained from the different contrasts. Differences between the determined nuclear and magnetic particle sizes allow conclusions on the possible presence and thickness of a magnetically dead or depleted surface layer.

Different scattering contrasts were achieved by varying the isotope composition of the solvent by mixing regular toluene with deuterated one. Five solvent compositions were produced with 0, 25, 35, 50 and 80 vol. % of deuterated toluene. Quartz cuvettes (Hellma GmbH, Germany) of 1 mm thickness were filled with 300 µl of the samples.

3.2 X-Ray diffraction and PDF analysis

3.2.1 PSI: MS-X04SA

Synchrotron X-ray scattering experiments on dried nanoparticle powder of sample OC15 were performed at the Material Science Beamline MS-X04SA of the Swiss Light Source at the Paul Scherrer Institut (Villigen, Switzerland). At this beamline an energy resolution of $\Delta E/E = 1.4 \times 10^{-4}$ can be achieved with a flux of more than 10^{13} photons/s (at 12 keV).[196] Particles in solution and as a powder were measured. For the powder sample the original nanoparticle dispersion was dried yielding a sticky powder due to the oleic acid coating. This powder was transferred into a glass capillary with 0.9 mm interior diameter, which was inserted into a brass fitting and loaded into the sample cassette. For the measurement the sample was picked up by the sample changer robot and inserted horizontally in the sample space. During the measurement the capillary was rotated around its long axis. After collection of the data with a 1D-detector the instrument related corrections were applied immediately. Additional measurements of the empty capillary, the empty beam and the solvent contribution for the liquid samples were also recorded. The NIST standard LaB_6 was used to determine the X-ray wavelength. After normalization and correction of the obtained powder pattern the pair distribution function (PDF) was calculated with PDFgetX3[141] using $Q_{min} = 0.8\,\text{Å}^{-1}$, $Q_{max} = 18\,\text{Å}^{-1}$, $r_{poly} = 1.3$ and a composition mixture of iron oxide and oleic acid. Analysis of the PDF was carried out with PDFGUI[142]. Further analysis of the powder diffraction patterns was performed with GSAS-II[197] and with Python scripts written for this thesis.

3.2.2 APS: 11-ID-B

Additional synchrotron X-ray diffraction experiments were conducted at the 11-ID-B beamline of the Advanced Photon Source (APS) at the Argonne national laboratory (Lemont, IL, USA). For these experiments samples were shipped and the measurements were done by the beamline staff. The energy resolution of this beamline is $\Delta E/E = 1 \times 10^{-3}$ with a flux of 2.3×10^{12} photons/s (at 58.6 keV). Highly concentrated nanoparticle dispersions were generated inside Kapton capillaries by filling the capillaries and letting the solvent evaporate. This process was repeated until the capillary was sufficiently filled. Afterwards the capillaries were sealed with a two component glue and placed in a cartridge that was returned to the beamline staff. After the data acquisition was completed the 2D-detector data for all samples including a CeO_2 reference and the empty capillary was sent. Background subtraction, calibration and integration was performed with GSAS-II[197] by using the empty capillary and the reference sample data. The PDFs were generated with PDFgetX3[141] using $Q_{min} = 0.8\,\text{Å}^{-1}$, $Q_{max} = 23\,\text{Å}^{-1}$, $r_{poly} = 1.3$ and a composition mixture of iron oxide and oleic acid. Analysis of this data was carried out with the software mentioned in the previous section.

3.3 Elemental analysis (ICP-OES)

Inductively coupled plasma optical emission spectroscopy (ICP-OES) was carried out at the central Institute for Engineering, Electronics and Analytics (ZEA-3), Forschungszentrum Jülich GmbH. Samples used for magnetometry measurements were microwave digested in HNO_3 and H_2O_2 and then analyzed. Co, Fe, Ni, Gd, Cr and Al were considered for determination of the total weight present in the sample. The basic principle of ICP-OES is the spectral decomposition of light emitted by the atoms and ions that were excited by a plasma. The wavelengths of the emitted light are used to identify the elements. From the intensity of the emitted radiation the absolute quantities of the elements can be determined.[198]

3.4 SQUID magnetometry

Magnetometry was performed using a superconducting quantum interference device (SQUID) magnetometer (MPMS XL, Quantum Design). The reciprocating sample option (RSO) was used in this work. With this method the sample is moved in an oscillatory fashion leading to an induction current inside the pickup coils (fig. 3.3). The winding of the pickup coils ensures that a current is only induced for changes in the magnetic field and not for uniform fields or linear gradients. To convert these small changes in the magnetic flux to voltages a SQUID ring with a Josephson junction is used.[199] The magnetic moment of the sample can then be obtained by measuring the time and sample position dependent SQUID voltage. This technique allows the measurement of magnetic signals with a sensitivity of $10^{-11}\,\text{Am}^2$.[200]

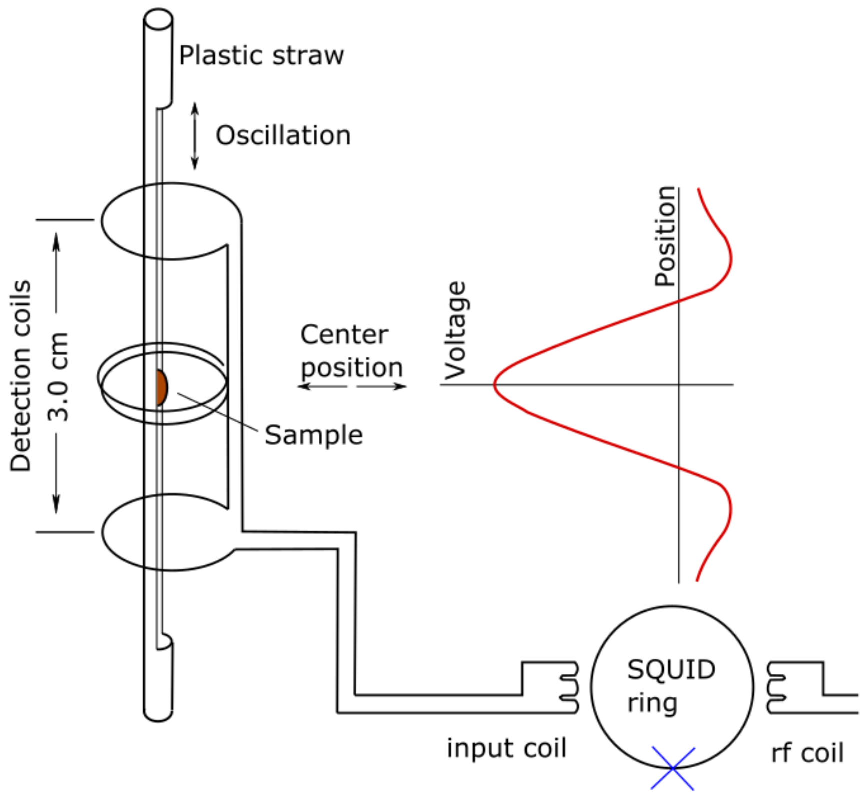


Fig. 3.3 Scheme of the experimental setup and the measurement process using a rf SQUID. Adapted from the technical data sheet for the MPMS-XL system.[200]

For the experiments performed for this work the nanoparticle dispersions were diluted in molten paraffin at temperatures of about 50 °C to 0.1 vol. % of the original dispersion to ensure negligible inter-particle interactions. After cooling of the nanoparticle-paraffin mixture pieces were extracted with ceramic tools to avoid contamination with ferromagnetic materials. The sample pieces were fixed onto a plastic straw with scotch tape that was inserted into the sample space (fig. 3.3). After the measurements the sample was removed from the straw and handed over to ZEA-3 for the elemental analysis with ICP-OES (see section 3.3).

Magnetization vs. temperature $M(T)$ curves were measured by first cooling the sample without an external magnetic field to 10 K and subsequent recording of the magnetization during heating to room temperature with a magnetic field applied. This measurement yields the zero-field cooled (ZFC) curve. After reaching room temperature the magnetization is recorded but upon cooling while keeping the applied field and the sweep rate at the same values as for the ZFC curve. This yields the field-cooled (FC) curve. The so-called aged zero-field cooled (aZFC) measurements were also performed by halting the ZFC process roughly at 50 K, i.e. below the maximum of the ZFC curve, for 10^4 s, then resumed to cool down and finally proceeding the measurement as for a regular ZFC curve. Comparison of the aZFC and ZFC curves allows conclusions on the presence of a collective superspin glass.[87] Magnetization vs. field $M(H)$ curves were obtained by sweeping over a field range of -1 to $1\,\mathrm{T}$ at constant temperatures. $M(H)$ curves after FC were recorded by applying various fields (0, 0.1 and $1\,\mathrm{T}$) during the cooling of the sample prior to the measurement.

3.5 Mössbauer spectroscopy

Mössbauer spectra were recorded at $4.3\,\mathrm{K}$ in transmission geometry and constant acceleration mode using a ^{57}Co(Rh) source in a liquid helium bath cryostat. As discussed in section 2.5 this source produces γ-rays in an energy range that is especially suited to study iron containing samples. The particle dispersions were dried and the obtained powders were mixed with chemically inert boron nitride to obtain a homogeneous sample of sufficient volume. The Mössbauer spectra presented in this work were recorded by Dr. Joachim Landers from the Faculty of Physics and Center for Nanointegration Duisburg-Essen (CENIDE) at the University of Duisburg-Essen.

3.6 TEM/ HRTEM

Transmission electron microscopy (TEM) and high resolution TEM (HRTEM) was carried out using a FEI Tecnai G2 F20 field emission transmission electron microscope, operated at $200\,\mathrm{kV}$ at the Ernst Ruska-Centre (ER-C), Forschungszentrum Jülich GmbH[201] with the help of Tanvi Bathnagar-Schöffmann (JCNS-2, FZJ). The sample was prepared by drop casting onto a carbon layer supported by a Cu-grid. Image simulations to verify the contrast were performed with the QSTEM software package[202]. Particle size measurements and image manipulations, such as the generation of fast Fourier transformed images and the inverse Fourier transformation after masking of Bragg peaks were performed with ImageJ[203]. Additional TEM measurements were performed at JCNS-4 (at MLZ in Garching) with the help of Dr. Marie-Sousai Appavou using a JEOL JEM-FS2200 field emission electron microscope, operated at $200\,\mathrm{kV}$. Again the nanoparticle dispersion was drop casted onto a carbon coated Cu-grid and the experiment was performed after the solvent evaporated.

3.7 Simulation programs

3.7.1 Monte Carlo program

3.7.1.1 Program structure

The structure of the Monte Carlo program used and developed for this work is sketched in fig. 3.4. The program is based on a previous implementation by O. Petracic and X. Sun. While the previous implementation written in C was purely procedural, in this work an object oriented approach was followed. A shell script

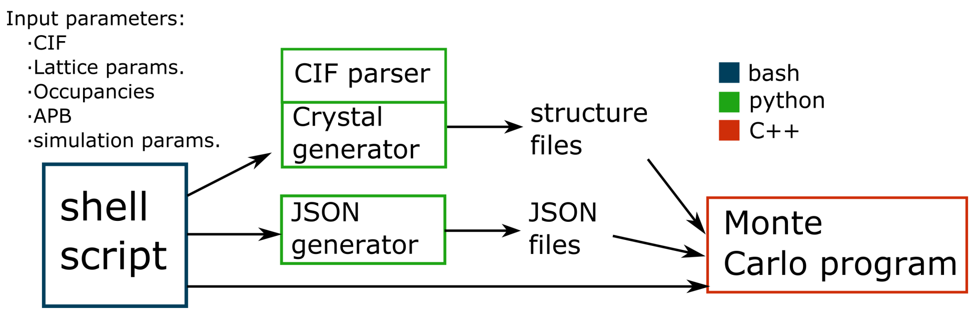


Fig. 3.4 Scheme for the structure of the Monte Carlo program developed in this work. The entry point is a shell script containing the input parameters, which in turn calls python scripts to generate the particle crystal structure as well as JSON files used as inputs in the main Monte Carlo program written in C++.

containing all input parameters can be executed from the command line. This script then calls Python scripts to setup the nanoparticle structure as well as JSON files that are both read into the main program written in C++.

Templates for the shell scripts have been developed for spin structure, magnetization vs. temperature and magnetization vs. field simulations. The parts written in *bash* calling the subroutines as well as the main program can be left unchanged. For different simulations only changes in the input parameters are necessary. For spin structure simulations these parameters include the output directory, the number of particle configurations to be simulated and a CIF (Crystallographic Information File) for the crystal structure. Since the program is developed for iron oxide nanoparticles additional parameters are the occupancies of iron positions and the lattice parameters. These are also specified in the CIF itself. However it was found to be more convenient to change these values in the shell script and keeping the CIF untouched.

The shape of the particle can be specified as well as the size and its orientation. Very important for this work is the possibility of generating an APB through the center of the particle. Further simulation parameters are the applied magnetic field, the temperature, the number of Monte Carlo steps, the sigma parameter of the Gaussian trial move and the mode of handling the dipole interaction calculations (brute force method, macrocell method or not considered). Upon execution of the shell script the crystal structure is generated by a separate Python script, where the unit cell parameters are parsed from the CIF and the unit cell is repeated a number of times to generate a $3D$ structure. Iron atoms are removed to obtain the specified occupancy at the iron positions and the shape is generated by removing all atoms outside of a volume defined by the shape and the size parameter.

The APB is optionally introduced prior to the shape cutting by shifting one half of the crystal along the [110] direction. The resulting structure is stored in a text file containing the x/a, y/b and z/c fractional coordinates, where a, b and c are the lattice parameters. Additionally, the position, i.e. octahedral

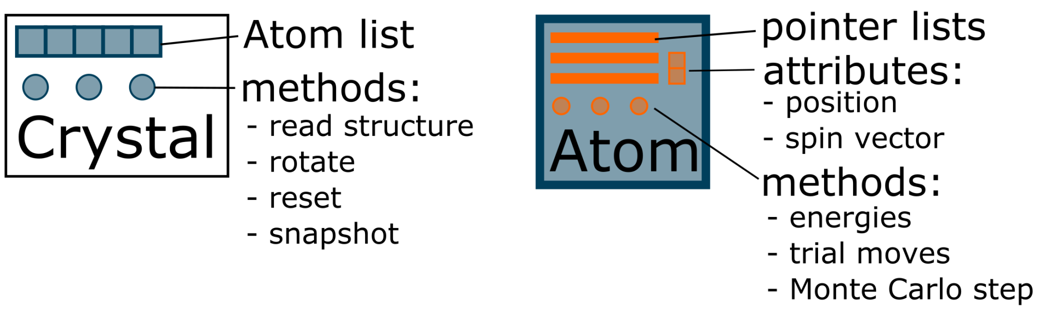


Fig. 3.5 Scheme for the basic structure of the crystal and atom objects used in the simulation program. The crystal instance contains a list of Atom objects at fixed memory locations. The atom objects in turn contain pointers to the other atom objects in the crystal to speed up calculations. A selection of the most important class methods is also shown.

or tetrahedral is encoded with 0 for the former and 1 for the latter. The fifth column has values of 1 if the atom is located at the APB and is subject to the modified exchange interactions, otherwise a 0 is entered. The last three columns specify the unit cell the atom belongs to. The JSON generator script sets up text files containing the simulation parameters in JSON format that can be read by the main program via the *JSON for Modern C++* (version 3.9.1) JSON-parser developed by Niels Lohmann[204]. The results of the simulations are written into new text files, with file names containing the most important parameters as well as all input parameters as comments within the text. In the case of spin structure simulations an average structure is calculated from the individual simulations by adding and normalizing the spins at each position. To deal with missing atoms at vacancy sites a template structure is generated without vacancies.

3.7.1.2 Main program

The main program is called from the shell script with the JSON file as input. Central to all simulation modes is the "Crystal" class defined in "Crystal.hpp" (fig 3.5). It contains a list of atom objects and has a number of methods. An atom object is an instance of the Atom class defined in "Atom.hpp". It contains the atom positions in space, the spin components, the exchange and anisotropy constants, the magnetic moment and other attributes relating to the position of the atom in the crystal. Important in speeding up the computation are lists of pointers to other atom objects, e.g. the nearest neighbours on both sub-lattices. The methods of the Atom class are the different energy contributions as discussed in section 2.1 as well as the trial move methods (see section 2.2.3). The "Monte-CarloStep" method performs one trial move on the atom object and determines if the move is accepted according to the Metropolis algorithm (see section 2.2.2). Upon starting the simulation a Crystal object is instantiated, where the Atom list is filled with instances of Atom, where in turn the position parameters are read from the input structure file generated by the Python crystal generator. After this

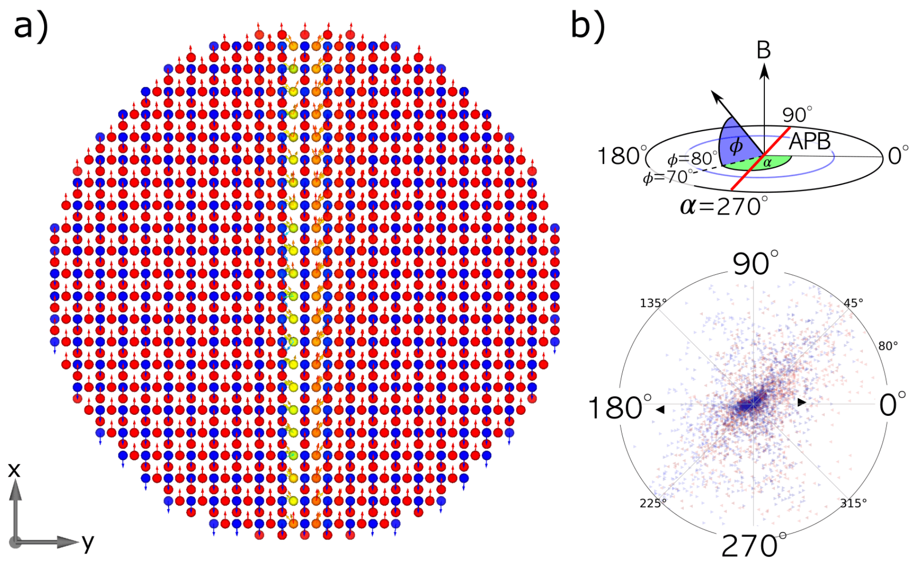


Fig. 3.6 a) Example of a simulated spin structure represented as color coded spin vectors on atomic positions with the same colors. Deviations from perfect alignment with the applied field oriented along x can be seen as colors deviating from red or blue. b) Example of the polar representation of spin structures. The top image shows the angles ϕ and α associated with each spin vector that are drawn in the figure below as colored dots. In this case the colors correspond to the spatial position of the spin vectors in the structure, i.e. positions to the left of the APB are drawn in red and positions to the right in blue. Other color coding could be used to distinguish octahedral and tetrahedral positions.

the crystal is rotated to the desired orientation and the neighbour lists are generated for each atom in the structure. If dipole interactions are calculated directly each atom gets also a list of pointers to all other atoms in the structure as well as pre-computed lists of all inter-atomic distances and distance unit vectors needed for the calculations. To speed up the dipole calculations the long sum of eq. 2.23 is calculated in parallel using OpenMP.

3.7.1.3 Spin structure visualization

Interpretation of the simulation results is much easier through some kind of graphical representation of the spin structures. In this work two types of visualizations are used. The first kind are 3D plots showing the spin vectors as arrows placed on their associated atomic positions allowing the direct assessment of the simulated spin structures (fig. 3.6a). However, from these plots it is sometimes difficult to see trends or collective spin canting as these effects may be very subtle or the respective region is covered by other spins. Thus, the second kind or representation is useful to provide a more statistical view on the spin structures by polar plots (fig. 3.6b).

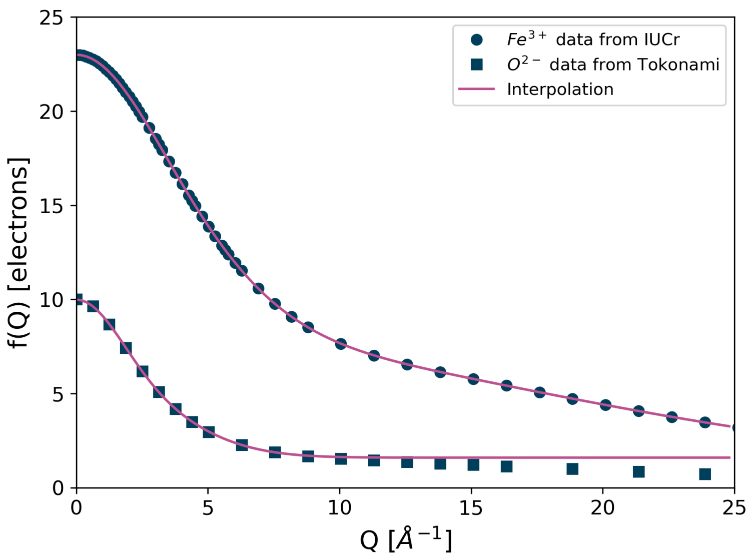


Fig. 3.7 Dots and squares show the tabulated values of the atomic form factors for Fe^{3+} and O^{2-} taken from Brown et. al[206] and Tokonami[207], respectively. The lines correspond to the interpolation according to eq. 3.1 with parameters given in tab. 3.1.

The 3D plots are produced with the Python package Mayavi.[205] Color coding is used to visualize the alignment of the spin vectors with an external magnetic field, e.g. with a red-green-blue color map, fully parallel or antiparallel alignment of the spin vectors with the field is displayed using red or blue color, respectively. Canted spins can then be identified by green or yellow colors. The implementation can be found in the appendix. The polar plots are generated with the Python library Matplotlib. For these plots the atomic spin vectors of the simulated structures are placed in the origin. A projection plane is chosen perpendicular to the applied field direction such that fully aligned spin vectors that are either parallel or antiparallel to the field vector correspond to points in the center of the plot. The position of the projected points is defined by an azimuthal (α) and an elevation angle (Φ). The elevation angle is defined as the angle between the spin vector and the plane perpendicular to the field vector, thus resulting in an angle of $\pm 90°$ for full alignment. For the plots the absolute values of the elevation angles are used. The closer the elevation angle of a spin vector is to $\pm 90°$ the closer the corresponding point is to the center of the plot. The azimuthal angle defines the orientation of the spin vector around the field vector. The APB is positioned on the line connecting azimuthal angles 90° and 270°. In general in such a statistical representation of the spin vectors a concentration of points shows the alignment of many spins in this particular direction, while scattered points show spin disorder.

3.7.2 Debye scattering equation program

The program to simulate X-ray powder diffraction patterns of nanoparticles via the Debye Scattering Equation (DSE) was written for this work entirely in Python.

Tab. 3.1 Coefficients for the form factor interpolation according to eq. 3.1 taken from the international tables for crystallography[206] and from Tokonami[207].

Ion	a_1	a_2	a_3	a_4	b_1	b_2	b_3	b_4	c
Fe^{2+}	11.0424	7.3740	4.1346	0.4399	4.6538	0.30530	12.0546	31.2809	1.0097
Fe^{3+}	11.1764	7.3863	3.3948	0.00724	4.6147	0.3005	11.6729	38.5566	0.9707
O^{2-}	4.758	3.637	0.0	0.0	7.831	30.05	0.0	0.0	1.594

The central part is a crystal class containing atom objects, similar to the Monte Carlo program. In fact a slight variation of the *crystal.py* module of this program was used as the crystal generator for the MC-program and the crystal generation for the DSE calculations is very similar to the previous description. The only difference is that here also the oxygen atomic positions are considered. The DSE involves a pair-wise sum over all atoms in the particle (see eq. 2.85) but since the distances between atoms do not change these distances can be computed in advance. Additionally, an array is set up containing labels for the form factor products to be used later for the calculation. Here atom pairs are associated with a number obtained from numbers corresponding to the elements. E.g. $Fe = 2$ and $O = 3$ thus the pair $Fe - Fe = 2 + 2 = 4$ and $Fe - O = 2 + 3 = 5$. This array contains numbers representing the product in the same order as the distances. The module *dse.py* is used to perform the calculations. It contains two classes, namely the *AtomicFormfactorCalculator* and the *DseCalculator*. The former is used to calculate the form factors for the elements in the different valence states present in the structure from the Cromer-Mann coefficients a_i, b_i and c according to

$$f(Q) = c + \sum_{i=1}^{4} a_i e^{(-b_i(Q/4\pi)^2)}, \tag{3.1}$$

where $Q = 4\pi/\lambda \sin\theta/2$, with the scattering angle θ and the X-ray wavelength λ.[206] The coefficients are taken from the international tables of crystallography vol. C[206] except for the O^{2-} ion which is not included in the tables, but given by Tokonami.[207] This parametrization is accurate for $Q < 25\,\text{Å}^{-1}$. Although for O^{2-} some deviation occurs already at $Q = 10\,\text{Å}^{-1}$ (fig. 3.7), this is still well above the Q-range needed for this work.

An *AtomicFormfactorCalculator* object is initialized in the constructor of the *DseCalculator* with the elements present in the input structure. Global variables are set up for the possible combinations of elements containing the pre-computed product of the respective form factor arrays. E.g. the atom pair $Fe - O$ is given the variable $FeO = f_{Fe}(Q) f_O(Q)$. The actual calculation of the powder diffraction pattern is then performed by setting up a list containing all the corresponding form factor pairs by filling in the form factor products for the numbers representing the pairs as well as taking the previously computed list of the distances and evaluating the expression

$$I_{ij}(Q) = \frac{f_{ij}(Q)\sin(Qr_{ij})}{Qr_{ij}}, \tag{3.2}$$

where f_{ij} and r_{ij} contain the form factor products and the distances, respectively, in the same order. For the evaluation the Python module *numexpr* by David M. Cooke et al. is used.[208] This module is able to evaluate expressions involving numpy arrays very fast by using parallel computing on a virtual machine written in C and by avoiding allocation of memory for intermediate results, yielding faster computation times than numpy. Finally, to obtain the scattered intensity the individual contributions $I_{ij}(Q)$ are added and the diagonal elements of the form factor product matrix are calculated separately and added to the final intensity array. The implementation of the function is given in the following code snippet.

```
1  def calculate_Dse(self, distances, f_ij):
2      aa = eval(list((self.atom_numbers.keys()))[0]*2)
3      ab = eval(list((self.atom_numbers.keys()))[0]
4               +list((self.atom_numbers.keys()))[1])
5      bb = eval(list((self.atom_numbers.keys()))[1]*2)
6  
7      f_ij_e = [ab if i == 5 else bb if i == 6 else aa for i in f_ij]
8      f_ij_e = np.array(f_ij_e)
9  
10     rij_reshaped = distances.reshape(len(distances),1)
11 
12     q = self.q
13 
14     fijsincqr = ne.evaluate("f_ij_e*sin(q*rij_reshaped)/(q*rij_reshaped)")
15     I_sum = fijsincqr.sum(axis=0)
16 
17     return 2*I_sum
```

The function arguments are the distance and the form factor product label arrays. These arrays are the flattened upper triangle of the pairwise matrix. Since the matrix is symmetric computation of the upper triangle is sufficient. The diagonal elements are not included and are added in a later step. The variables *aa*, *ab* and *bb* are lists of the form factor arrays for all the possible pairwise products. These are obtained by retrieving the atom element labels, e.g. *Fe* or *O*, stored in the dictionary *atom_numbers*. For a Fe_xO_y structure *atom_numbers.keys()* returns *dict_keys(['Fe','O'])*, where the first key is extracted by converting the dict_keys object to a list and using list indexing *[0]*. The value inside *eval()* is thus *'Fe'*2* which gives *'FeFe'*. Since *FeFe* was defined as global variable the *eval* function returns the form factor array associated with the variable. In line 7 the input form factor product array that contains only the place holder numbers 4, 5 or 6 is used to fill *f_ij_e* with the appropriate form factor arrays. The distances are obtained from the *crystal.py* module in Å. The $Q = 4\pi \sin\theta/2/\lambda$ array is obtained from the class attribute in line 12. The variable *fijsincqr* contains a nested numpy array as a result of the evaluation of the expression eq. 3.2 via the *numexpr* module imported as *ne*. Each array in *fijsincqr* corresponds to one pair of atoms, thus the scattered intensity is obtained by summing all these individual contributions giving an array of intensities as a function of q stored in *I_sum*. Finally this array is multiplied by 2 to account for the lower triangle of the pairwise matrix. To obtain the complete diffraction pattern the diagonal elements of the pairwise matrix are added later. This method of simulating powder diffraction patterns described here could be optimized by the use of several tricks, however the direct method is preferred for this work to not introduce artefacts. The methods used in other implementations of the Debye scattering equation include the grouping of terms with the same interatomic distances, using the crystal symmetry to reduce the number of terms and sampling the interatomic distances.[112] The latter method is called histogram approximation[209], implemented e.g. in DEBUSSY[112,210] and DISCUS[211].

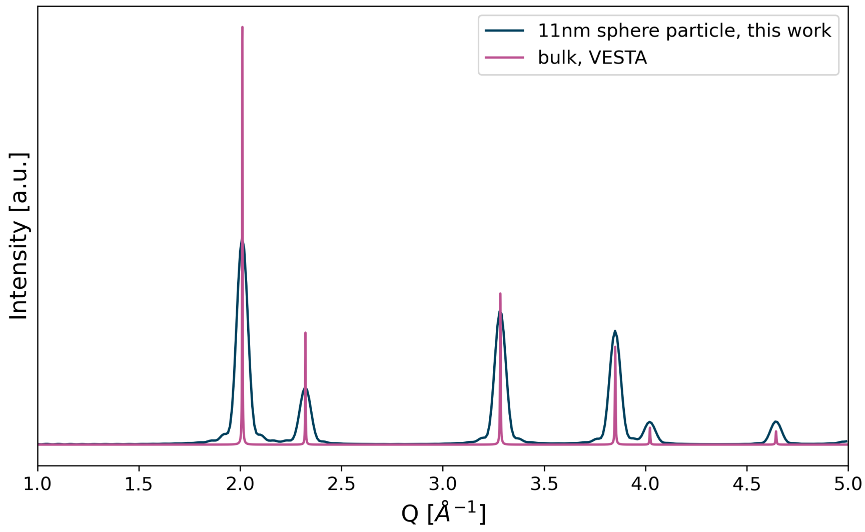


Fig. 3.8 Simulated powder diffraction patterns for the NIST standard $CeO_2 - 674b$ using the code of this work ($10.8\,\mathrm{nm}$ sphere) and the program VESTA (bulk). Peak broadening due to the finite size of the particle is visible. The peak positions of both program outputs match.

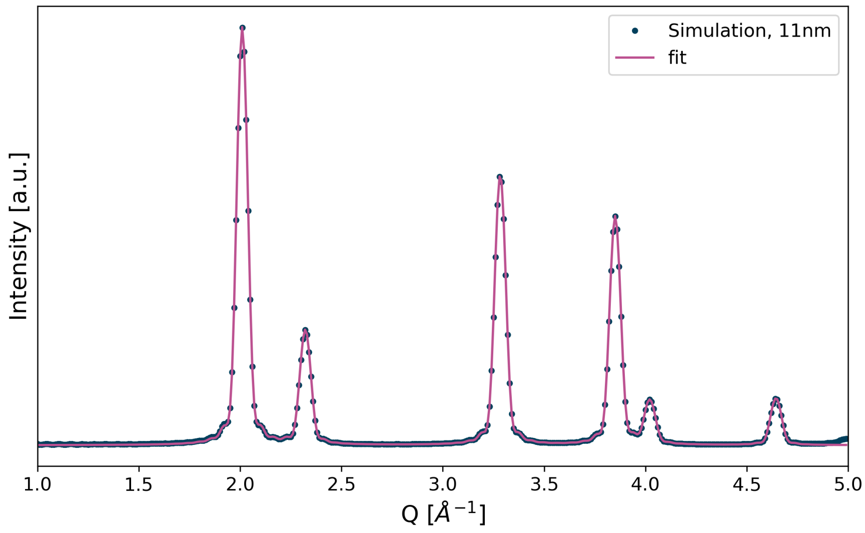


Fig. 3.9 Fit to the simulated powder diffraction pattern of the $10.8\,\mathrm{nm}$ spherical CeO_2 particle using eq. 2.149. The input size parameter is recovered with high precision.

The code has been tested by comparing the calculated powder diffraction patterns with bulk patterns obtained from VESTA.[153] In fig. 3.8 the simulated patterns are shown for $CeO_2 - 674b$, a NIST standard sample. A spherical particle with diameter of $10.8\,\mathrm{nm}$ was generated with the code developed for this work and the powder diffraction pattern was calculated as described above. The peak positions as well as the relative peak intensities match well for small Q. Some deviations in the relative peak intensities are present at large Q due to the fact that in the code of this work thermal displacements are not considered. The thermal displacements of atoms would lead to a decrease in peak intensity with increasing scattering angle.[109] To further check the correctness of the code a fit to the simulated data was performed using the peak profile for a spherical nanoparticle as given in eq. 2.149 (fig. 3.9). The input parameter of the simulation for the particle size is obtained by the fit. Additionally, the small satellite peaks originating from the spherical shape and the finite size are also reproduced precisely.

Chapter 4

SPIONs: Internal Magnetization Distribution

The study of the magnetization distribution within superparamagnetic iron oxide nanoparticles (SPIONs) is the subject of this thesis. As mentioned in the introduction chapter (section 1.2) there is a large body of previous works that addressed different parts of this problem for particles of various sizes, shapes and synthesis routes. In this work a number of different techniques is used to obtain an as complete as possible picture of the the different contributing effects. The focus here lies on particles synthesized by the thermal decomposition route, which allows the production of particles in large quantities and with narrow size distributions. As noted previously[45,46] however these kinds of particles are prone to develop antiphase boundaries that are proposed to be responsible in large parts to an often observed reduced saturation magnetization. Simulations of the influence of these defects on the spin structure and the net magnetization for magnetite nanoparticles were performed by Nedelkoski et al.[47] In this thesis simulations are performed for maghemite nanoparticles, also under explicit consideration of dipole-dipole interactions between atomic magnetic moments that were only approximated in the cited work. The Debye scattering equation is used to study the effect of the finite size and of APBs on X-ray powder diffraction patterns. In the third part of this chapter the investigation of nanoparticle samples with nominal sizes of 5, 10, 12, 15 and 20 nm is performed, where the previously established results from the simulations are used to help interpret the findings.

Parts of this chapter have been published previously. The Monte Carlo and Debye scattering equation simulation results were published under the title *"Signature of antiphase boundaries in iron oxide nanoparticles"* in the *Journal of Applied Crystallography.*[212] The present author developed the concept of the manuscript, wrote the simulation software, performed the analysis and interpretation of the simulated and experimental data, visualized the data, created the figures and wrote the manuscript. Review and editing of the original draft was done with the help of the co-authors. The right hand part of fig. 4.4 appears in the publication as fig. 2 and is reproduced here, the left part is shown in the supplementary fig. S5. Parts of fig. 4.6 are present in modified form in fig. 5 and supplementary figs. S6, S7 and S8 of the publication. Figs. 4.8a, 4.9 and 4.12 appear in modified form in fig. 6 of the publication. The derivation of the Fourier transform of the APB peak profile as shown in appendix B of the published work was performed by the present author and is presented in section 4.2.2.

The analysis of data for sample OC15 (see tab. 4.2) has been published in *Nanoscale* under the title *Mechanism of magnetization reduction in iron oxide nanoparticles.*[84] The present author contributed to the investigation by performing the experiments and analysing the data of the SAXS, magnetometry, X-ray, PDF, SANS and HRTEM (with the help of Tanvi Bathnagar-Schöffmann at the instrument) measurements. Mössbauer experiments were performed by Dr. Joachim Landers. Data analysis of the Mössbauer experiments was performed with the help of Dr. Joachim Landers. The data was visualized and the published figures were created by the present author. The original draft of the manuscript was written by the present author. Review and editing of the original draft was done with the help of the co-authors. Fig. 2 of the publication appears in a modified version in fig. 4.52 of the present work. Figs. 3a, 3b and 3c appear in the present work in figs. 4.33 and 4.34. Fig. 6 of the publication is shown in a modified form as part of fig. 4.24 of the present work. Fig. 7 of the publication is shown in a modified form in figs. 4.44 and 4.45.

4.1 Monte Carlo simulation results

In this section the effect of antiphase boundaries on the spin structure and the macroscopic magnetization are studied with Monte Carlo simulations. These results are part of an submitted manuscript entitled *"Signature of antiphase boundaries in iron oxide nanoparticles"*. In the last part also the influence of vacancy ordering on the magnetization is investigated.

4.1.1 Simulation parameters

Monte Carlo simulations using the Metropolis algorithm (see section 2.2) are performed in order to study the influence of antiphase boundaries on the spin structure of iron oxide nanoparticles. For more details on the program see section 3.7.1. Several approximations were used to keep the number of parameters at a minimum and not distract from the main goal, i.e. the study of the effect of an antiphase boundary. To this end the cubic space group $Fd\bar{3}m$ with a random vacancy distribution on the octahedral sites is assumed. However, the effect of vacancy ordering was also studied separately.

Additionally, idealized nanoparticles are simulated, meaning that no gradients of iron, oxygen or vacancy concentration were considered. The lattice was also assumed to be not distorted towards the particle surface as is likely for small real particles. Surface anisotropy was also not considered. The ground state structures are of interest, hence the temperature was set to a small value of 0.01 K. It should be mentioned here that at this temperature the Boltzmann factor in eq. 2.64 mostly vanishes and new states are almost always only accepted if the energy of the new state is lower than in the previous configuration. Additionally, the system might be trapped in a local minimum without finding the global minimum and thus the true ground state. To address this problem the simulated annealing approach has been proposed, where the system is cooled from high temperatures to the target temperature.[213] However, with the strong field of 5 T used in the simulations the energy landscape is expected to be rather smooth and in a first approximation the resulting configuration is assumed to be reasonably close to the true ground state. A test simulation without dipole-dipole interactions but with an annealing phase from 300 to 0.01 K with equilibration of 5000 Monte Carlo steps every 10 K temperature step resulted in very similar spin structures compared to results obtained from simulations without prior cooling of the system.

APBs of type I were created by shifting one half of the crystal structure by $1/4a$ along [110], where a is the cubic lattice parameter. This results in a planar APB through the particle center. As mentioned in section 2.6.3 the APB is expected to be planar after full equilibration of the system due to interfacial energies.[188] For real particles this is an additional approximation. However, as can be seen from the experimental data

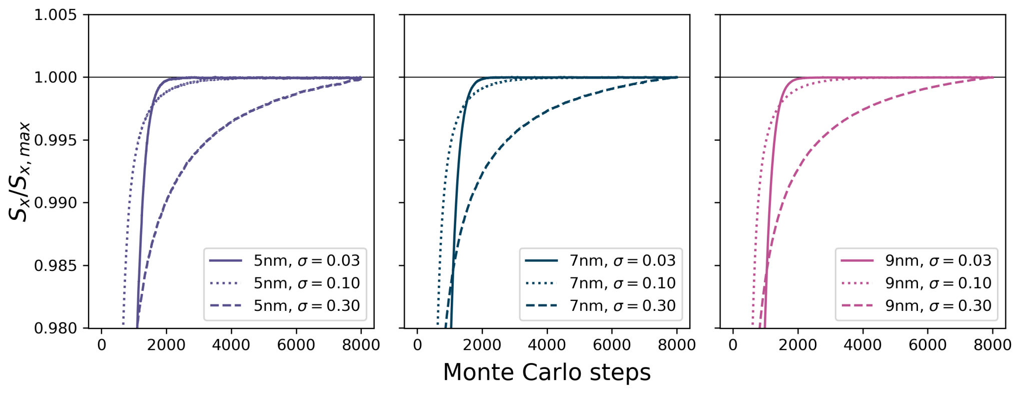


Fig. 4.1 Test of particle magnetization convergence towards saturation. A field of $5\,\mathrm{T}$ was applied along x at a temperature of $0.01\,\mathrm{K}$. For three particle diameters of 5 to $9\,\mathrm{nm}$ the sum of all spin vector components along x was calculated as a function of Monte Carlo steps and for different opening angles of the Gaussian trial move defined by the parameter σ (see section 2.2.3). Twenty independent relaxation measurements were averaged to obtain each curve.

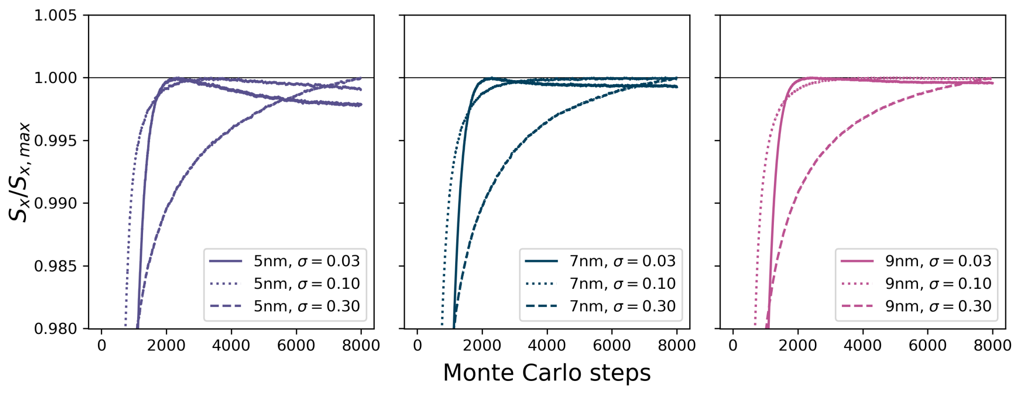


Fig. 4.2 Test of particle magnetization convergence towards saturation for particles with an APB in the center. A field of $5\,\mathrm{T}$ was applied along x at a temperature of $0.01\,\mathrm{K}$. For three particle diameters of 5 to $9\,\mathrm{nm}$ the sum of all spin vector components along x was calculated as a function of Monte Carlo steps and for different opening angles of the Gaussian trial move defined by the parameter σ (see section 2.2.3). Twenty independent relaxation measurements were averaged to obtain each curve.

shown in section 4.3 a single planar APB not in the particle center is not in agreement with experimental data, but most likely a more complex arrangement of multiple APBs is present in real particles. For such kinds of particles the introduced spin structure disorder will be stronger than for the simulated particles with a single planar APB. Thus the present simulation model can be considered as a lower limit of the possible APB influence on the magnetization distribution. Finally, it should be mentioned that the dipole-dipole interactions are computed directly for the spin structure simulations without further approximations such as the macrocell-method or cutoffs. This implicitly also considers shape anisotropy of the particles. Theoretically shape anisotropy is not present for a perfectly spherical sample, however especially for the smaller considered particles atomic columns might introduce step like features on the particle surface which could lead to a small contribution of shape anisotropy.

For reliable simulation results it is important to make sure that the system has converged to equilibrium. Therefore, a test was performed by recording the sum of the spin components parallel to the applied field as a function of the Monte Carlo steps. Different opening angles for the Gaussian trial move (see section 2.2.3) were also considered. The results are shown in fig. 4.1. For all particle sizes it can be seen that the opening angle influences the approach to equilibrium. A larger opening angle (larger σ) allows the spin vector to rotate farther from the previous orientation, leading to slower convergence. For a smaller σ of 0.10 the approach to equilibrium is faster at first, but the resulting saturation after approx. 1500 Monte Carlo steps is smaller than that obtained by an even smaller opening angle. Therefore, to reduce computation times and ensure equilibrium for the following spin structure simulations σ was set to 0.03. Performing the same test with particles containing an APB (fig. 4.2) shows that for $\sigma = 0.03$ equilibrium is only reached at approximately 5000 Monte Carlo steps for particles larger than 7 nm. For smaller sizes even more Monte Carlo steps are needed to fully equilibrate the system. This is likely due to the stronger influence of the APB on the particle in this size range.

4.1.2 Effect of APBs on the spin structure

Simulated spin structures for particles with diameters of 6.7 and 9.2 nm with an APB through the center are shown in fig. 4.3. The ferrimagnetic alignment of the octahedral and the tetrahedral sub-lattices is visible from the red and blue spin arrows denoting parallel and antiparallel alignment with the applied field of 5 T along the x-direction. The structures are viewed along the z-direction thus the lattice shift due to the APB is not immediately visible. However, the resulting spin disorder can be clearly seen from the differently colored spins near the APB showing deviations in the alignment. This disorder is better visible in the close-up views of the 9.2 nm particle shown in fig. 4.4 with the APB in the center. For the 6.7 nm the spin structure around the APB is very similar. At the applied field of 5 T the spins on both octahedral and tetrahedral lattice sites are canted. However this effect is confined to the octahedral iron magnetic moments directly affected by the modified exchange interactions and the moments of adjacent tetrahedral iron ions. On this local scale the dipole interactions appear to not have a large influence on the observed spin canting, which is expected considering the much larger energies involved in the exchange interactions.

To gain a better understanding of the spin structure on a larger scale polar plots are useful. The construction of these plots is described in section 3.7.1.3. From this

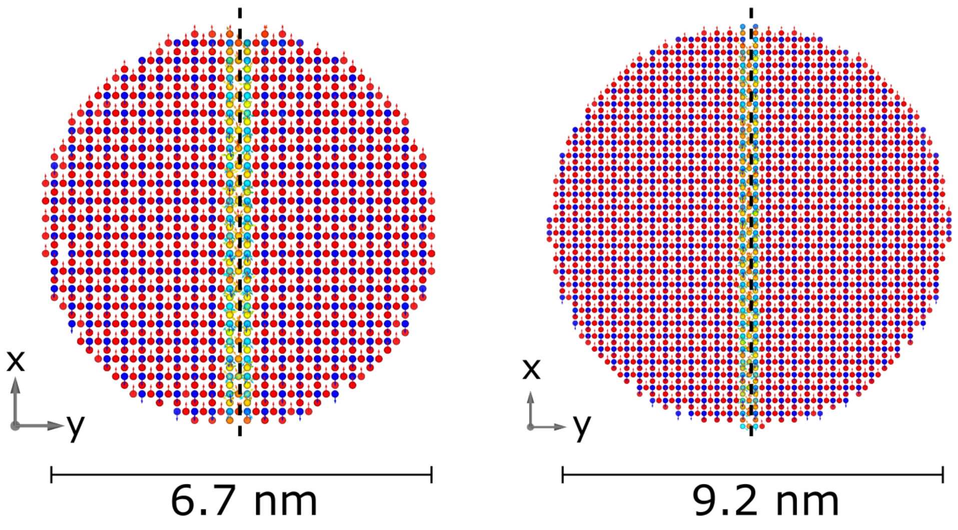


Fig. 4.3 Simulated spin structures for particles of different sizes with an APB through the particle center. A field of $5\,\mathrm{T}$ is applied along the x-direction. Full parallel and antiparallel alignment of spins are indicated with red and blue arrows, respectively. Clearly visible is the ferrimagnetic order that is disturbed by the APB as can be seen by the change of the colors. A close-up of the spin structure at the APB for the largest particle is shown in fig. 4.4.

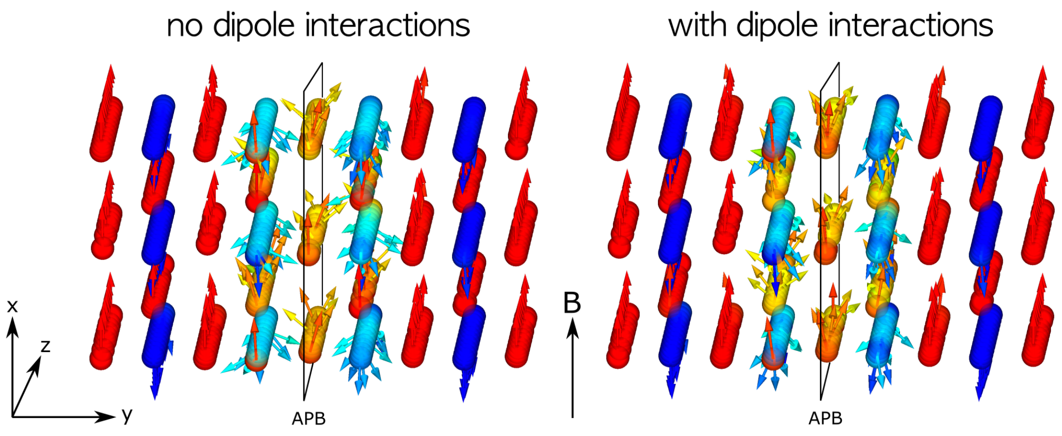


Fig. 4.4 Close-up view of the spin structure around the APB in the center. The view is along atomic columns in the z direction. A field of $5\,\mathrm{T}$ is applied vertically along the x direction, thus full parallel and antiparallel alignment are shown in red and blue, respectively. Spin disorder is clearly visible around the APB but order is restored at larger distances from the APB. The inclusion of dipole-dipole interactions in the simulations has only a small influence onto the result.

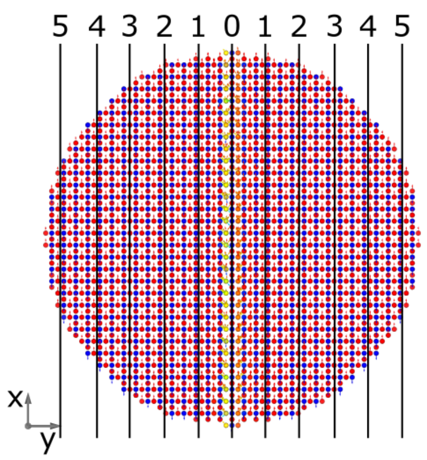


Fig. 4.5 Illustration of the labeling of slices shown in fig. 4.6. The APB lies in the $x - z$ plane. Each slice has a thickness of one unit cell.

statistical point of view collective spin misalignment can be more easily identified. To also be able to see the spatial variation of the spin structure the simulated particles are cut into slices parallel to the APB plane (fig. 4.5). For both considered particle sizes it can be seen that collective canting in the left and right slices in the very vicinity of the APB, i.e. slice 0-1, is present (fig. 4.6). Upon increasing the distance to the APB no difference between the net magnetic moment orientation in the left and right slices from the APB can be seen. The very small net transverse moment for the slice $0 - 1$ in the simulations of the 6.7 nm particles (fig. 4.6b) is a consequence of the strong spin disorder around the APB.

For both considered particle sizes it can be seen that the APB introduces a significant disorder in the spin structure that is also localized in the sense that collective spin canting is not observed already one unit cell distance away from the APB. A similar behavior was found experimentally in magnetite thin films.[187] This indicates that the APB does not lead to a magnetic multi-domain state as has been proposed in the literature[47,48], but that the particle as a whole remains in the single domain state, at least for the particle sizes examined here.

The inclusion of dipole-dipole interactions in the simulations in general leads to spin canting oriented more along the APB plane (along the 90°-270° line in fig. 4.6), while in the simulations without this energetic contribution the spin canting appears to be spatially more isotropic.

4.1.3 Effect of APBs on the macroscopic magnetization

To study the effect of APBs on the macroscopic magnetic properties of iron oxide nanoparticles $M(H)$ curves were simulated. No significant difference in the saturation magnetization can be observed between structures calculated with and without dipole-dipole interactions. The spin structure in the fields considered here is mainly a result of the exchange interactions. Due to the spherical shape of the particles the shape anisotropy effect is negligible and with the small particle size below the superparamagnetic limit domain formation is not energetically favorable. Therefore, atomic dipole-dipole interactions are neglected in the following. To obtain data that is better comparable to

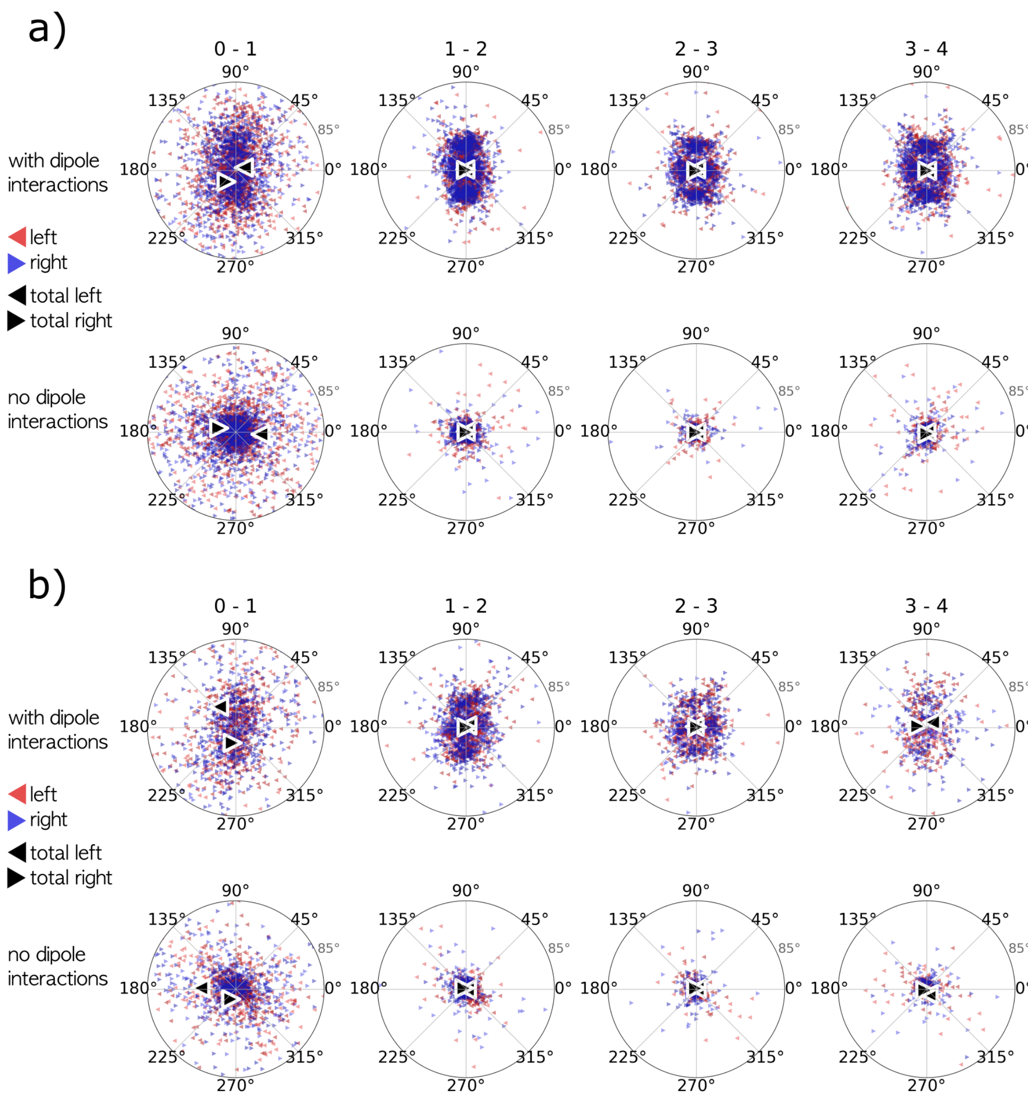


Fig. 4.6 a) Polar plots of slices to the left and the right of the APB in the particle center for particles of $9.2\,\mathrm{nm}$ in diameter. Twenty independent simulated configurations were averaged. Red and blue triangles correspond to atomic spin vectors in a slab to the left or to the right from the APB. The black triangles show the net magnetization of the slabs, where again a left facing triangle corresponds to a slab to the left and vice versa. Collective canting of the magnetic moment in the slices is only observed directly at the APB as indicated by the difference in placement of the net magnetization triangles. In addition spin disorder is clearly visible in the region around the APB. b) Polar plots of slices for particles with a diameter of $6.7\,\mathrm{nm}$. Spin canting of the net magnetic moment in a slice is again only observed directly at the APB $(0-1)$ as indicated by the difference in placement of the net magnetization triangles. Also similar to the larger particles depicted in a) significant spin disorder is present in the region around the APB.

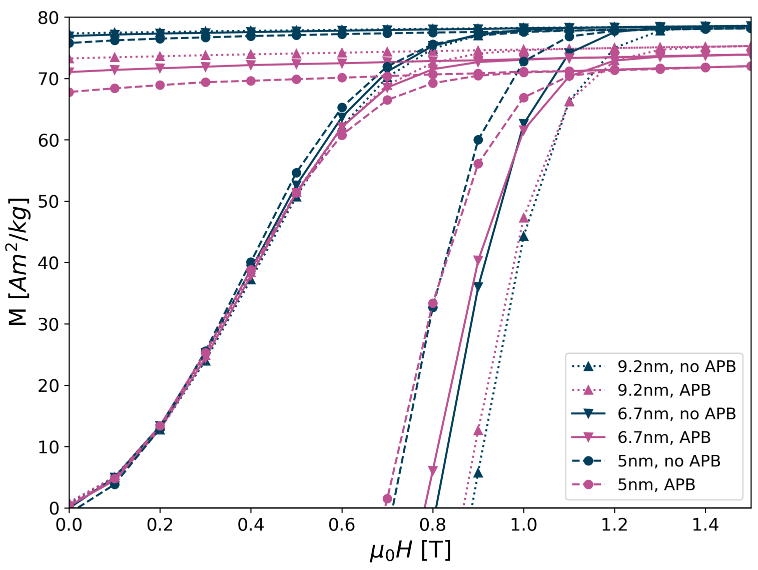


Fig. 4.7 Simulated $M(H)$ curves for three different particle sizes (5.0, 6.7 and $9.2\,\mathrm{nm}$) with (blue lines) and without APBs through the center (pink lines). Only the first quadrant is shown. Simulations results from averaging over twenty particle configurations for twenty different orientations each, thus giving 400 hysteresis loops. At each field step of $0.1\,\mathrm{T}$ the spin structure of the particle was allowed to relax for 8000 MCS.

real measurements twenty particle configurations each with twenty randomly selected orientations in space are averaged. The simulation was performed over a field range of -1.5 to $1.5\,\mathrm{T}$ with 8000 MCS per field step of $0.1\,\mathrm{T}$ at a temperature of $5\,\mathrm{K}$. With 8000 MCS every spin in the structure is statistically rotated 8000 times thus allowing the relaxation of the particle spin structure at each field step. More steps would result in a smaller coercive field of the hysteresis loops as the relaxation time-scale approaches that of the measurement. The simulations were performed for three particle sizes with diameters of 5.0, 6.7 and $9.2\,\mathrm{nm}$ (i.e. 6, 8 and 11 unit cells in diameter). The net mass magnetization, M, of the simulated particles in units of $\mathrm{Am}^2/\mathrm{kg}_{\mathrm{Ferrite}}$ is obtained from the sum of the spin vectors according to

$$M = \frac{\mu}{V\rho}\sum_{i}^{N} S_{x,i}, \tag{4.1}$$

where μ is the magnetic moment of the iron ion taken as $5\mu_B$, $V = 4/3\pi R^3$ is the spherical particle volume with the radius R, $\rho = 4.9\,\mathrm{g/cm}^3$ is the mass density and S_x is the x-component of the spin vector, i.e. the component along the applied field. The results of this simulation are shown in fig. 4.7.

A reduced saturation magnetization for particles containing an APB compared to those without one can be observed. For the smallest particles the magnetization at $1.5\,\mathrm{T}$ is reduced by about $8.1\,\%$ and $8.5\,\%$ at $1\,\mathrm{T}$. For the intermediate particle size a reduction of $5.9\,\%$ at the maximum field is obtained that is slightly increased to $6.5\,\%$ at $1\,\mathrm{T}$. For the largest particles this reduction in magnetization is by about $4.8\,\%$ at the largest field and $5.1\,\%$ at $1\,\mathrm{T}$. The dependence of the relative reduction of saturation magnetization on the particle size is correlated with the relative volume of the APB inside the particle. As seen in the previous section the spin disorder is spatially confined at saturation fields and thus the relative amount of disordered spins decreases with increasing particle size. The decrease in magnetization reduction with increasing magnetic fields suggests also a field dependence of this effect, which is linked to the gradual alignment of spins at the APB for larger fields.

Experimentally perfectly crystalline nanoparticles were found with almost bulk magnetization,[33,35,214] while particles with proven APB presence show strongly reduced magnetization.[46,47,84] E.g. a reduction of about 13 % for 15.6 nm particles was attributed mostly to APBs.[84] This is more than what is observed for the largest particle size considered in the simulation. However, as mentioned in the beginning, the assumption of a single planar APB with no additional lattice distortions has to be taken as the lower limit of the possible influence of APBs on the magnetization. For larger particles it is possible that multiple APBs form that would lead to a stronger reduction in magnetization.

4.1.4 Simulation of vacancy ordering

As described in section 2.6.2 the vacancies present in the maghemite structure can be ordered, which leads to a reduction of the space group symmetry from cubic to tetragonal. Ordering of vacancies was found to increase the saturation magnetization of otherwise defect free nanoparticles up to the bulk values.[31,33]

To investigate the effect of vacancy ordering on the saturation magnetization particles with randomly placed vacancies and particles with vacancies restricted to certain iron sites were simulated. For the random placement in the $P4_32_12$ space group that means setting the occupancy of $Fe2$, $Fe3$ and $Fe4$ to 0.83 and leaving $Fe1$ fully occupied. Vacancy ordering while keeping the bulk occupancy, i.e. an average iron site occupancy of 0.89, was introduced by setting the occupancy for $Fe2$ to 0.33 and leaving all other iron sites fully occupied. Twenty simulations at 5 T, 0.01 K and with 5000 MCS were averaged for each particle size. Dipole-dipole interactions were considered as the vacancy ordering is expected to produce a long range influence on the spin structure. For 5.0 nm particles with random vacancy distribution the net magnetization obtained by a sum over the spin vector components parallel to the applied field is reduced by 1.2 % compared to the particles with vacancy ordering. Similarly, for 6.7 nm a reduction of 1.4 % is found and for 9.2 nm particles of 1.0 %.

The simulations suggest that vacancy ordering does increase the saturation magnetization, however this effect is relatively weak. Li et al.[33] correlate an increase in saturation magnetization from 20 to about 65 $\mathrm{Am^2/kg}$ at 300 K to an increase in vacancy ordering from fully random vacancy placement for particles of size 6 nm to complete vacancy order in the tetragonal space group $P4_32_12$ for particles with diameters of 53 nm. However, the maximum observed saturation is already reached for particles with diameters of 13 nm, where full vacancy ordering is not yet reached. Morales et al.[31] relate an increase in saturation magnetization from 64 to 72 $\mathrm{Am^2/kg}$ to an increase in vacancy ordering from fully random to fully ordered. In this work the studied particles were different in shape and much larger. The highest saturation magnetization was found for spherical particles with diameters of 120 nm, while lower magnetization was found in elongated particles with partial vacancy order and length and width of 295 nm and 98 nm, respectively.

The lowest saturation magnetization was found in the largest elongated particles with dimensions of 530 nm and 84 nm with fully random vacancy placement.

4.1.5 Summary

The influence of APBs on the spin structure of maghemite nanoparticles at applied fields of 5 T was shown to be confined to the immediate vicinity to the APB without indication for a magnetic multi-domain structure. For the single APB considered the reduction in the saturation magnetization is size dependent and largest for the smallest particles. At 1.5 T and 5 K the relative reduction changes from 8.1 % for 5.0 nm particles to 4.8 % for 9.2 nm particles. Application of a weaker magnetic field leads to a larger reduction in the saturation magnetization. Finally, simulations of vacancy ordering showed that a higher degree of ordering in perfectly crystalline structures leads to a slight increase in saturation magnetization.

4.2 Debye Scattering Equation Simulation Results

Experimental X-ray powder diffraction patterns are the result of the convolution of many different effects originating from the sample and from the instrument.[133] In the context of nanoparticles, line broadening due to the particle size and shape is especially important. An additional effect that is considered in this work is the line broadening due to structural defects such as APBs. As mentioned in section 2.3.3 the use of the Debye scattering equation (DSE) results in powder diffraction patterns that contain all features originating from the finite size and structural disorder. Thus, as the first step to check the correctness of the program output and to gain a proper understanding of the finite size effect the resulting patterns obtained from spherical particles with different sizes are analyzed in the following. The model particles for this part do not contain further defects. In the second part of this section the influence of APBs on the line profiles is investigated. Finally, the influence of APBs on the pair distribution function is analyzed. With the exception of this last part the presented results are a part of an submitted manuscript entitled *"Signature of antiphase boundaries in iron oxide nanoparticles"*.

4.2.1 Finite size effect

In fig. 4.8 the simulated powder diffraction patterns for spherical particles with sizes of 5.0 nm, 6.7 nm and 9.2 nm are shown. The fit curves in fig. 4.8a) are obtained by a superposition of peak profiles according to

$$I(Q, Q_{\text{peak}}, D) = \int_0^D A_{\text{size}}(L, D) e^{i\pi(Q-Q_{\text{peak}})L} dL, \qquad (4.2)$$

where $A_{\text{size}}(L, D)$ is given for a sphere in section 2.3.5.3, D is the particle diameter and Q_{peak} is the peak position. The peak positions are fixed to the theoretically expected values for the implemented unit cell. The resulting particle diameters are in excellent agreement with the input parameters of the simulation with 5.0(1), 6.7(1) and 9.2(1) nm. The use of the Scherrer equation (eq. 2.146) to determine the particle size gives 5.3(1),

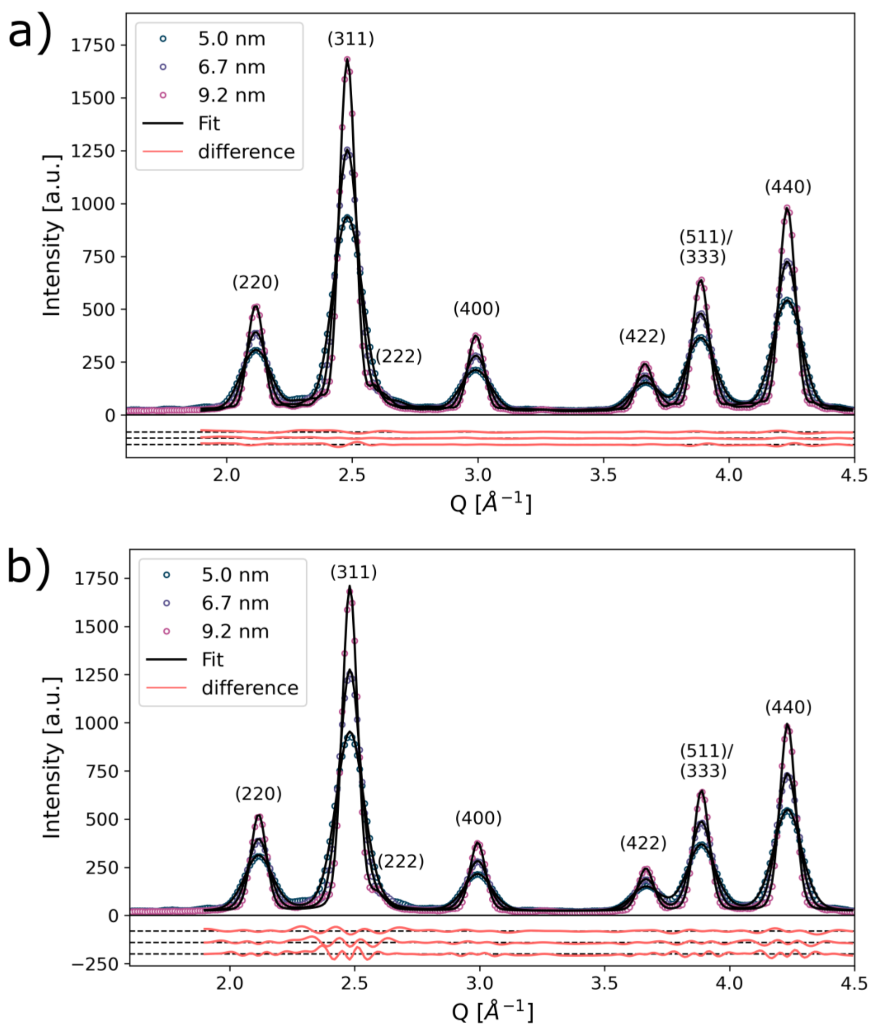


Fig. 4.8 a) Simulated powder diffraction patterns for different sizes of spherical maghemite nanoparticles using the Debye scattering equation (eq. 2.85), normalized to the number of atoms in the structure. The fit curves are obtained by a superposition of Patterson peak profiles according to eq. 2.149. The difference curves for the three fits are shown below with an offset for clarity, with the fit difference for the smallest particle at the top and for the largest at the bottom. The parameter Q is defined as $Q = 4\pi/\lambda \sin\theta/2$. b) The same simulated data as shown in a) fit with a superposition of Voigtian peak profiles according to eq. 2.154. The difference curves for the three fits are shown below with an offset for clarity, with the fit difference for the smallest particle at the top and for the largest at the bottom.

7.2(1) and 9.9(1) nm for $K = 0.94$. These values are larger than the input parameters. As described in section 2.3.5.3 this discrepancy is due to the Voigtian peak profiles used to determine the full width at half-maximum (FWHM), which have no physical relation to the observed peak shape of the nanospheres and lead to larger mismatch in the fits (fig. 4.8b). Therefore, eq. 4.2 is preferred. The use of the integral form also allows the inclusion of further Fourier coefficients to describe other effects in a process called whole powder pattern modelling (WPPM) proposed by Scardi and Leoni.[215]

4.2.2 APB effect

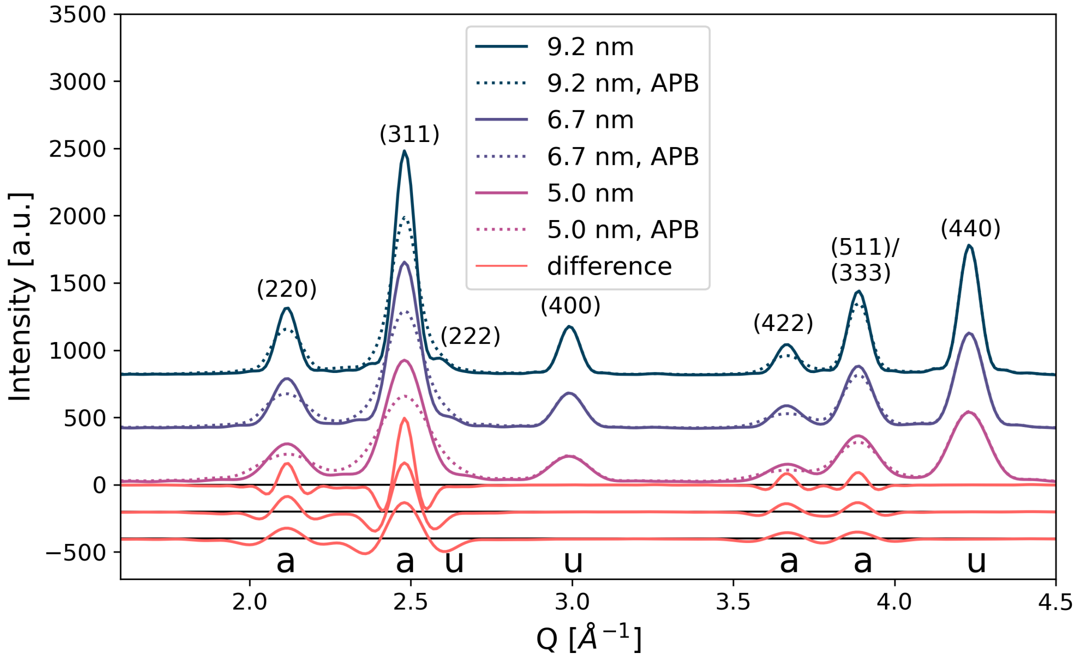


Fig. 4.9 Simulated powder diffraction patterns for different sizes of spherical maghemite nanoparticles without (solid lines) and with APB (dotted lines) through the particle center using the Debye scattering equation (eq. 2.85). From the difference curves (no APB - APB) it can be seen that some peaks are completely unaffected by the APB. These are marked with "u" below in contrast to the affected peaks marked with "a". The patterns are normalized to the number of atoms in the particle and offset for clarity. The difference curves are in the same order as the simulated patterns and also offset for better visibility.

From the difference of patterns obtained for particles with and without APB through the particle center a distinct *hkl*-dependence of peak broadening can be observed (fig. 4.9). Certain peaks appear to not have changed upon introduction of the APB, while others are significantly broadened. The powder diffraction pattern calculated from the DSE contains all effects resulting from the structural properties of the input structure. For the simulated particles this means that only a finite size effect and the effect of antiphase boundaries is expected since no other defects have been introduced into the model structure. Additionally, no superstructure peaks due to vacancy ordering are expected as the vacancies were distributed randomly. The resulting diffraction pattern is therefore a convolution of the profile originating from the finite size and from the APB according to

$$I(Q) = \int_0^D A_{APB}(L) A_{size}(L, D) e^{i(Q-Q_{peak})L} dL, \tag{4.3}$$

where Q_{peak} is the peak position, $A_{APB}(L)$ is the Fourier transform of the profile function resulting from the APB and $A_{size}(L, D)$ corresponds to the Fourier transform of the profile function originating from the finite size. The integral is over the real space

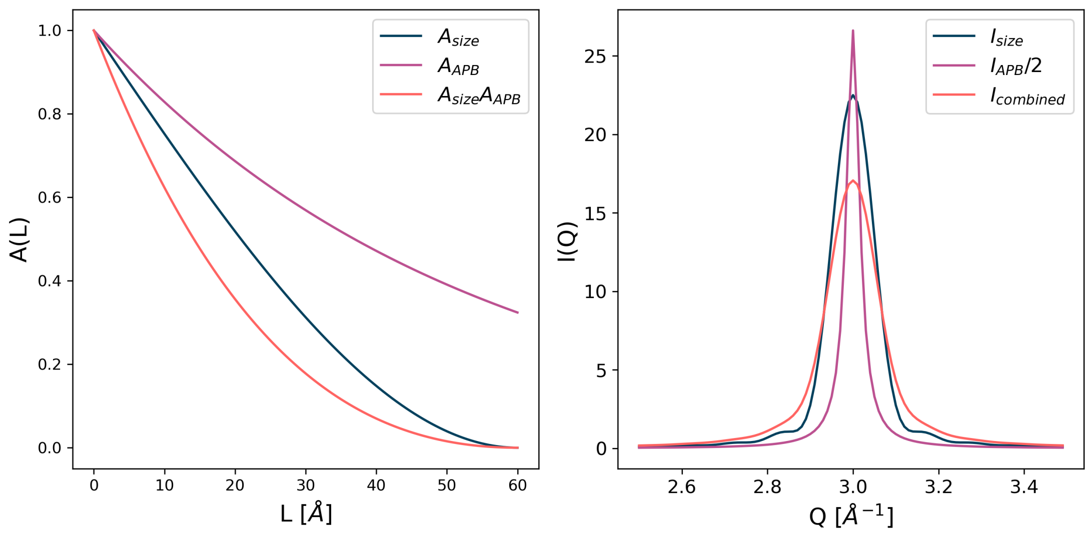


Fig. 4.10 Simulations of the effect of both the presence of an APB and a small crystallite size with diameter $D = 6\,\mathrm{nm}$. a) Real space Fourier coefficients. b) Resulting peak profile after Fourier transformation.

distance L with an upper integration limit of the particle diameter D. As shown by Scardi et al.[175] the integration over the APB component results in a Lorentzian peak profile, while the integration over the size component results in the previously mentioned Patterson peak profile (eq. 2.149).[135] A simulation of the individual components and the resulting peak profile is shown in fig. 4.10, where the decrease in intensity and increase in peak width can be seen.

To be able to completely model the simulated powder diffraction pattern the precise form of A_{APB} has to be known, which is developed later in this section. A first understanding of the influence of APBs can be obtained by fitting all peaks with the Patterson peak profile or with eq. 2.147 and determine the coherent structural sizes, in this case directly obtained from the fitting parameter. For each peak a coherent domain size in the direction perpendicular to the lattice planes corresponding to this peak can be determined. Deviations of the coherent sizes from the previously determined particle sizes thus show the influence of the APB. As seen from fig. 4.9 the affected peaks are broadened and are reduced in intensity. Warren[109] pointed out that this is due to the fact that the integral intensity of the peaks is not influenced by the presence of APBs, thus in order to conserve the integral intensity a broader peak must have a smaller maximum peak height, which is also shown in the simulation of the peak profile in fig. 4.10. The existence of unaffected peaks can be explained by considering the phase changes originating from the displacement of atoms in a crystal structure. For example across the APB the oxygen sub-lattice does not change, therefore no phase shift is expected and the corresponding peak should not be affected by the antiphase boundary, which is indeed the case for the (400) peak whose corresponding lattice spacing coincides with the oxygen lattice (fig. 4.9). Other unaffected peaks in the considered Q-range are the peaks (222) and (440).

The coherent sizes determined from a fit to the peaks (fig. 4.11) are given in tab. 4.1. The affected peaks show different degrees of broadening and the coherent size ratios are

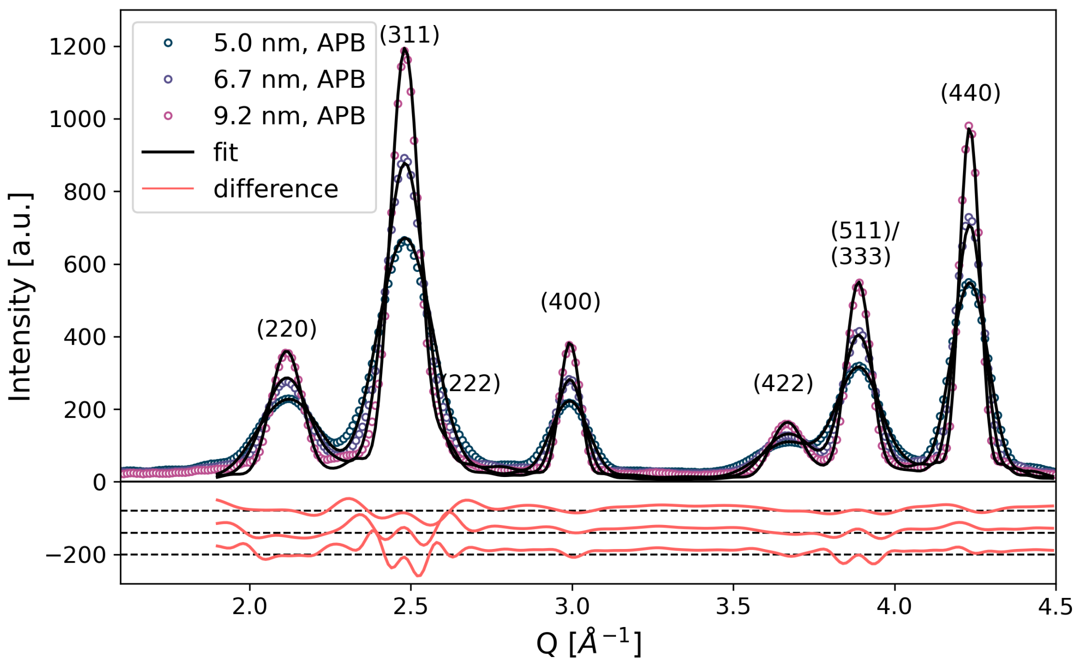


Fig. 4.11 Simulated powder diffraction patterns for different sizes of spherical maghemite nanoparticles with an APB through the particle center using the Debye scattering equation (eq. 2.85). The fit curves are obtained by a superposition of peak profiles according to eq. 2.149. The difference curves are shown below, corresponding from top to bottom to $9.2\,\mathrm{nm}$,$6.7\,\mathrm{nm}$ and $5.0\,\mathrm{nm}$. The patterns are normalized to the number of atoms in the particle.

almost independent of the particle size, differing only slightly for the smallest simulated particles. Since the APB was introduced in the center of the particles the coherent domain size perpendicular to the APB plane is halved for all particle sizes, thus the relative decrease in the coherent structure size is the same for all particle sizes. The slightly different values obtained for the smallest particles are most likely related to a non-ideal spherical particle shape and a stronger influence of the origin choice for the generation of the particle structure from periodic repetition of the unit cell. The feature of the independence of the coherent size ratios from the particle size is useful for a comparison between peak widths obtained from different particle sizes. For the same amount of APBs in a particle regardless of the size similar coherent size ratios are expected. However, as seen in fig. 4.11 the fits show some deviations, which is due to the modification of the peak profile due to the APBs as seen in fig. 4.10, that are not correctly described by eq. 2.149.

To explain the difference in the degree of broadening and the resulting peak shapes for the affected peaks the theory by Wilson, Zsoldos and Warren can be used.[109,172,174] The approach of Wilson and Zsoldos is based on infinitesimal increments of the distance within the structure along a certain crystallographic direction. The approach of Warren is based on finite differences of distances. As shown by Scardi et al.[175] both approaches are almost equivalent, however the finite differences are closer to the physical reality of distinct atomic positions in a crystal. Therefore, for this work the theory of Warren is

Tab. 4.1 Results of the fits depicted in fig. 4.11. For the APB-patterns a single particle size was refined for the unaffected peaks (u) and for affected peaks (a) it was left unconstrained. The ratio of apparent size D$_{app.}$ to the nominal particle size D is also shown. Standard deviations are given in parentheses.

Particle size		5.0 nm		6.7 nm		9.2 nm	
hkl	refl. type	$D_{app.}$ [nm]	$D_{app.}$/D	$D_{app.}$ [nm]	D$_{app.}$/D	$D_{app.}$ [nm]	D$_{app.}$/D
220	a.1	3.20(5)	0.64(1)	4.08(8)	0.61(1)	5.86(11)	0.63(2)
311	a.2.1	3.46(3)	0.69(1)	4.75(4)	0.71(1)	6.58(4)	0.71(1)
222	u	49.6(3)	0.99(1)	6.74(4)	1.00(1)	9.28(4)	1.00(1)
400	u	49.6(3)	0.99(1)	6.74(4)	1.00(1)	9.28(4)	1.00(1)
422	a.1	3.32(13)	0.66(3)	4.16(16)	0.62(3)	6.00(18)	0.63(3)
511/333	a.2.2	4.49(5)	0.90(1)	5.90(7)	0.88(1)	8.10(7)	0.88(1)
440	u	5.04(5)	1.01(1)	6.74(4)	1.00(1)	9.28(4)	1.00(1)

followed. As seen before the $1/4a[110]$ APB leads to a phase shift in the diffracted wave from some lattice planes resulting from the difference in the spatial arrangement of the atoms. The scattered intensity from a coherent crystal (eq. 2.139) is modified by the phase shift resulting from the APB. The scattered intensity for the crystal is obtained by the absolute square of the sum over the amplitudes from all unit cells. Each cell has an associated phase factor that depends on the position of the cell in the crystal. The phase difference between a cell at the origin and a cell at $n_1n_2n_3$ can be expressed by an average phase factor $\langle e^{i\Phi(n_1n_2n_3)}\rangle$, where $\Phi(n_1n_2n_3)$ denotes the change in phase between the two cells. Noticing that the phase changes along the three directions in real space are independent of each other the phase factor can be written as the product of the average phase factor in each direction according to

$$\langle e^{i\Phi(n_1n_2n_3)}\rangle = \langle e^{i\Phi(n_1)}\rangle\langle e^{i\Phi(n_2)}\rangle\langle e^{i\Phi(n_3)}\rangle. \tag{4.4}$$

Furthermore, the average phase factor along one direction can be described as an average factor $\langle e^{i\Phi}\rangle$ between pairs of cells that occurs n times, thus resulting in

$$\langle e^{i\Phi(n_1n_2n_3)}\rangle = \langle e^{i\Phi}\rangle_a^{|n_1|}\langle e^{i\Phi}\rangle_b^{|n_2|}\langle e^{i\Phi}\rangle_c^{|n_3|}, \tag{4.5}$$

where n_1 is the number of unit cells traversed per unit distance along $[hkl]$ by the component of the vector $[hkl]$ parallel to a and correspondingly for the other directions. Along the a direction the possible phase changes due to a type-I APB, i.e. a sublattice displacement by $\frac{1}{4}a[110]$ (section 2.6.3) are $\Phi = 0$, i.e. no change and $\Phi = 2\pi\left(\frac{1}{4}(k+l)\right) = \pi\frac{1}{2}(k+l)$ if change occurs. Thus for all three directions the following relations can be set up

$$\begin{aligned}\langle e^{i\Phi}\rangle_a &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(k+l)}\\ \langle e^{i\Phi}\rangle_b &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(h+l)}\\ \langle e^{i\Phi}\rangle_c &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(h+k)},\end{aligned} \tag{4.6}$$

where δ is the probability for the occurrence of a change in phase, i.e. the probability of crossing an APB, that in this theory is assumed to be equal for all directions. For certain combinations of hkl a phase factor equal to 1 is obtained, i.e. no observable difference in the peak width compared to a structure without APB. These are the unaffected peaks with indices $h+k=4n$, $k+l=4n$ and $h+l=4n$ (see fig. 4.9).

The different degrees of broadening for the affected peaks observed in the simulated pattern can be understood by considering different groups of peaks which lead to different phase factors. A first distinction can be made between peaks with only even (a.1) or only odd indices (a.2). For the a.1 peaks eq. 4.5 gives

$$\begin{aligned}\langle e^{i\Phi}\rangle_a &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(4n-2)} = 1-2\delta \\ \langle e^{i\Phi}\rangle_b &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(4n-2)} = 1-2\delta \\ \langle e^{i\Phi}\rangle_c &= (1-\delta)e^0 + \delta e^{i\pi\frac{1}{2}(4n)} = 1-\delta+\delta = 1,\end{aligned} \tag{4.7}$$

exploiting the fact that for these reflections one pair of indices is divisible by four. The cubic symmetry allows rearrangement of the indices for this pair to always be $h+k$. Thus using eq. 4.5 yields

$$\langle e^{i\Phi(n_1 n_2 n_3)}\rangle = (1-2\delta)^{|n_1|+|n_2|}. \tag{4.8}$$

The peaks of group a.2 can further be subdivided into those peaks with two pairs of indices divisible by four (a.2.1) and peaks where no sum of pairs is divisible by four (a.2.2). For a.2.1 using the same reasoning as before eq. 4.5 gives

$$\langle e^{i\Phi(n_1 n_2 n_3)}\rangle = (1-2\delta)^{|n_1|}. \tag{4.9}$$

Similarly, for a.2.2

$$\langle e^{i\Phi(n_1 n_2 n_3)}\rangle = (1-2\delta)^{|n_1|+|n_2|+|n_3|} \tag{4.10}$$

is obtained. The equations for the average phase factors are the Fourier coefficients to be used in eq. 4.3, i.e.

$$A_{APB}(L) = (1-2\delta)^{nL}, \tag{4.11}$$

where n is the appropriate exponent for the average phase factor, e.g. $|n_1|+|n_2|$ and L is a distance in real space in the direction perpendicular to the respective lattice planes, i.e. in direction $[hkl]$. The associated apparent domain size due to the APBs can then be obtained via[175]

$$D_{\text{eff.}} = 2\int_0^\infty (1-2\delta)^{nL}dL = -\frac{2}{\ln(1-2\delta)}\frac{1}{n} \approx \frac{1}{\delta n}, \tag{4.12}$$

For the last equality the approximation $\ln(1-2\delta) = -2\delta$ was used, valid for small δ. This contribution is convoluted with the finite size broadening, but the observed difference in the degree of peak broadening can be seen to be related to the different average phase factors for the kinds of reflections and the APB probability δ.

Quantification of the number of APBs is possible by fitting the diffraction pattern with a superposition of peak profiles generated by eq. 4.3 and extracting the parameter δ. For this the precise exponents $|n_1|+|n_2|$, $|n_1|$ and $|n_1|+|n_2|+|n_3|$ expressed in terms of the hkl-indices of the respective lattice planes have to be known resulting in the parameter n of eq. 4.11. Adapting considerations of Wilson and Zsoldos[174] for the

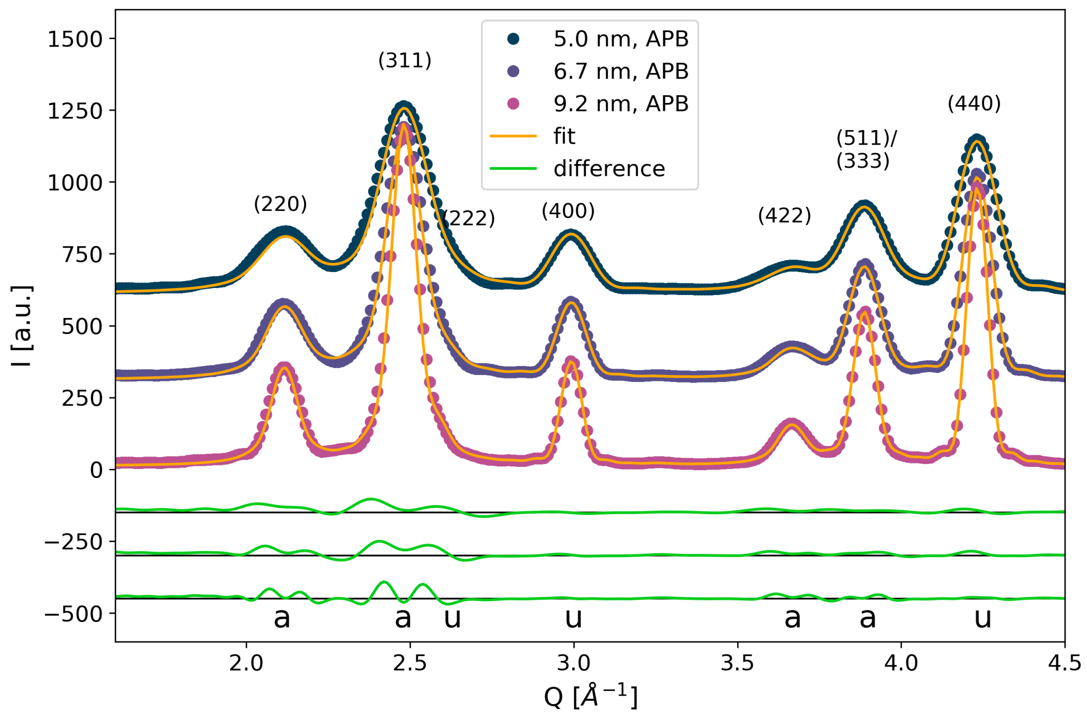


Fig. 4.12 Comparison of a model diffraction pattern and the simulated pattern for 5.0, 6.7 and $9.2\,\mathrm{nm}$ particles with one APB in the center. The model was calculated with peak profiles according to eq. 4.3 using the hkl-dependence of the APB effect as given in eqs. 4.17, 4.18 and 4.19 and δ values of 0.760, 0.567 and 0.413 with increasing particle size. The scaling factors, peak positions and the particle size were kept fixed from a fit to the patterns of a particles without APB (fig. 4.8).

use in Warren's theory the number of cells along $[HKL]$ traversed by dL pointing in the direction $[hkl]$ is proportional to the cosine of the angle α between $[hkl]$ and $[HKL]$, i.e.

$$\cos\alpha = \frac{\boldsymbol{a}\cdot\boldsymbol{b}}{ab} = \frac{|Hh+Kk+Ll|}{\sqrt{H^2+K^2+L^2}\sqrt{h^2+k^2+l^2}}, \tag{4.13}$$

where $\boldsymbol{a}$ denotes the vector in direction $[HKL]$ and $\boldsymbol{b}$ corresponds to the vector in direction $[hkl]$. The cosines obtained for different orientations of $[HKL]$ that are related by symmetry have to be added. For the case $|n_1|+|n_2|$ these are the orientations of $\langle 110\rangle$. Due the cubic symmetry only six orientations have to be considered, namely $[110]$, $[101]$, $[011]$, $[01\bar{1}]$, $[\bar{1}01]$, and $[1\bar{1}0]$. The sum of the equations obtained by inserting the respective $[HKL]$ is thus

$$\begin{aligned}&[(h+k+0)+(h+0+l)+(0+k+l)+(0+k-l)+(-h+0+l)\\ &\qquad +(h-k+0)]\,/\sqrt{2(h^2+k^2+l^2)} = \frac{2h+2k+2l}{\sqrt{2(h^2+k^2+l^2)}}.\end{aligned} \tag{4.14}$$

Similarly for the case $|n_1|$ the orientations are [100], [010], [001] and the corresponding cosines are

$$\begin{aligned}&[(h+0+0)+(0+k+0)\\&\qquad +(0+0+l)]\,/\sqrt{(h^2+k^2+l^2)} = \frac{h+k+l}{\sqrt{(h^2+k^2+l^2)}}.\end{aligned} \tag{4.15}$$

Finally, for $|n_1|+|n_2|+|n_3|$ the orientations are $[\bar{1}11]$, $[\bar{1}\bar{1}1]$, $[1\bar{1}1]$, and $[111]$ resulting in

$$\begin{aligned}&[(-h+k+l)+(-h-k+l)+(h-k+l)\\&\qquad +(h+k+l)]\,/\sqrt{3(h^2+k^2+l^2)} = \frac{4l}{\sqrt{3(h^2+k^2+l^2)}}.\end{aligned} \tag{4.16}$$

With these considerations the Fourier coefficients of the APB contribution for the different groups of peaks are

$$A_{APB}(L,D)_{a.1} = (1-2\delta)^{\frac{L(2h+2k+2l)}{a_0\sqrt{2(h^2+k^2+l^2)}}} \tag{4.17}$$

$$A_{APB}(L,D)_{a.2.1} = (1-2\delta)^{\frac{L(h+k+l)}{a_0\sqrt{(h^2+k^2+l^2)}}} \tag{4.18}$$

$$A_{APB}(L,D)_{a.2.2} = (1-2\delta)^{\frac{L(4l)}{a_0\sqrt{3(h^2+k^2+l^2)}}}. \tag{4.19}$$

An implementation of eq. 4.3 is given in appendix E. A fit using eq. 4.3, with the appropriate A_{APB} for the different groups inserted, is shown in fig. 4.12. For the fit the peak positions, the particle sizes and the scaling factors for the peaks were fixed to the values obtained from the simulation of particles without APBs (fig. 4.8). Only the parameter δ was varied, which gave values of 0.760, 0.567 and 0.413 for the three particle sizes of 5.0, 6.7 and 9.2 nm. To determine the actual number of APBs within the particle it can be assumed that an APB of the type considered in this work is always parallel to one real-space direction. The total probability of crossing an APB is obtained from the sum of the individual probabilities along x, y and z. Assuming the APB lies parallel to the z direction the probability along this direction is zero. Further assuming that the probability along x and y is equal, the total probability is given by $\delta = \frac{2}{3}\delta_x$. The probability of crossing an APB along a direction is given by the number of APBs n_{APB} divided by the number of unit cells along the sphere diameter D, i.e. $n_{\text{unit cells}} = D/a_0$. Thus,

$$\delta_x = \frac{n_{\text{APB}}}{n_{\text{unit cells}}} = \frac{n_{\text{APB}}}{(D/a_0)}. \tag{4.20}$$

Solving for the number of APBs gives

$$n_{\text{APB}} = \frac{3}{2}\delta\frac{D}{a_0}, \tag{4.21}$$

where δ_x has been substituted for $\frac{3}{2}\delta$ since δ is the fitting variable. Finally, the APB is parallel to (110) crystallographic planes in the crystal structure and hence forms a 45° angle with the x-axis. Due to this angle not all paths along x through the particle cross the APB. Viewing the particle along the direction x only $1/\sqrt{2}$ of the total cross

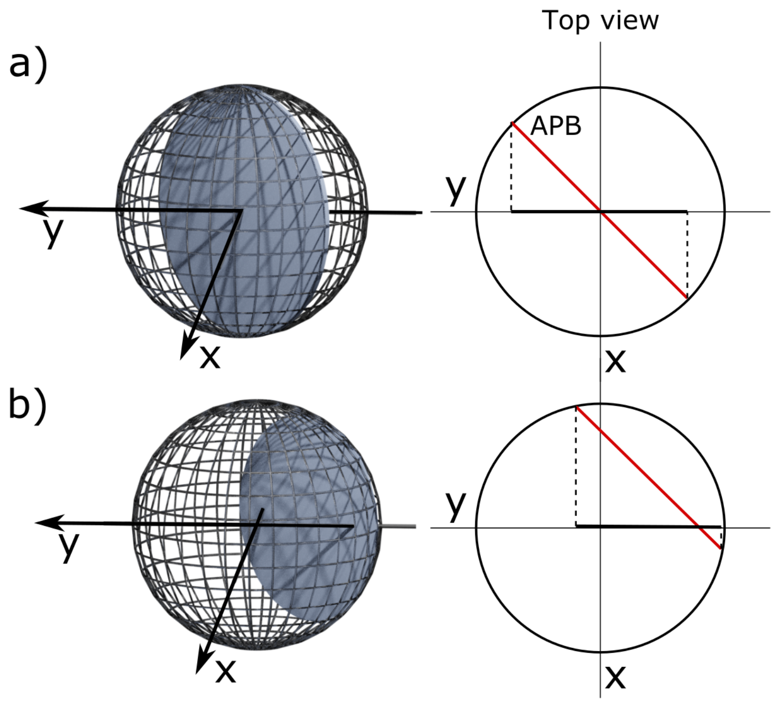


Fig. 4.13 Sketch of the APB orientation in the center of a spherical nanoparticle. On the right hand side the top view is shown. The APB, drawn with a red line, forms a $45°$ angle with the x-axis. In a) the APB is placed in the center, while in b) the APB is moved away from the center leading to a smaller fraction of the APB projection plane onto the yz-plane to the particle cross section.

sectional area is covered by the projection of the APB onto the yz-plane (see fig. 4.13) and the resulting number of APBs from eq. 4.21 has to be divided by this fraction. Hence, the number of APBs in the center of a spherical nanoparticle can be estimated with

$$n_{\mathrm{APB}} = \frac{3}{2}\sqrt{2}\frac{D}{a_0}\delta. \tag{4.22}$$

Application of this equation to the values determined for the three simulated particle sizes results in numbers of APBs in the particle center of 0.96 for all particle diameters. The small deviation from the expected value of one APB in the particle center might be related to the assumption of large crystals in the theory of Warren and Wilson to describe the APB peak broadening. With respect to real data, additional contributions to the peak profile may have to be considered that are convoluted with the effects discussed here. These contributions include surface relaxation of the crystal lattice, strains, dislocations and twin faults. An estimate on the severity of these additional contributions can be made by studying the peak shape and width of the peaks that are not affected by the APB (e.g. peaks (400) and (440)). As shown by Nunes and Lin[216,217] surface relaxation leads to an asymmetry of the peak shape, where the low-angle peak tail is increased in intensity compared to the higher angle tail. Leoni and Scardi[217] note that surface relaxation also leads to a peak shift of the large Q peaks towards lower angles. No significant peak broadening results from surface relaxations.

Dislocations and strain lead to an hkl-dependent line broadening.[218] However, all peaks are affected to some degree[219] and for cubic crystal systems the effect is different for $h00$ and $hk0$ reflections.[220,221] Hence, a comparison of the profiles obtained from peak (400) and (440) should reveal this contribution if present. The typical {111} twin faults observed for magnetite and maghemite nanoparticles[222,223] lead to a change also in the oxygen sub-lattice and should therefore affect the peaks (400) and (440) as well. Thus, if the coherent structure sizes obtained from peaks (400) and (440) coincide well with each other and with expected values for the studied particles additional effects may be ignored in a first approximation. For the use of eq. 4.3 this means if satisfactory fits can be produced by considering only one particle size, these additional contributions are likely negligible. For most of the described effects Fourier coefficients have been given in the literature, which means they can be included in eq. 4.3 if necessary. It has to be noted, that a large background may make the assessment of the coherent sizes difficult. Additionally instrumental broadening might be strong for laboratory X-ray sources. Synchrotron data is thus preferred. The inclusion of the instrumental broadening in eq. 4.3 is straight forward with the normalized Fourier transform of a pesudo-Voigt function as given in eq. 2.155.

Finally, the APB in a real particle might not be positioned exactly in the center. The effect of different degrees of off-centering is shown in fig. 4.14. The peak broadening is reduced significantly for larger distances of the APB to the particle center. The number of APBs in this case is reduced to smaller values than 1 due to a decreasing ratio of the APB-plane projected onto the yz-plane to the particle cross section (fig. 4.13b)). The number of APBs estimated with eq. 4.22 thus corresponds to the theoretical number of APBs placed in the particle center that could lead to the observed peak broadening. It has to be noted, however, that more APBs might be present at varying distances to the particle center leading to a similar peak broadening.

4.2.3 Influence of APBs on the PDF

The effect of an antiphase boundary on the PDF of the particle is shown in fig. 4.15. The PDF for two particle sizes was calculated from powder diffraction patterns simulated with the Debye scattering equation. Particles with diameters of 5 and 6.7 nm were used. The patterns were calculated over a Q-range of 0.01 and 20.0 Å^{-1}. The PDF was calculated from these patterns by excluding the small-angle scattering region of the diffraction pattern, i.e. setting Q_{min} to 1.0. Since the simulated patterns have no strong decay in peak intensity towards higher Q, a damping was introduced that reduces cutoff artefacts. Finally the PDF, $G(r)$, was calculated via eq. 2.161, i.e.

$$G(r) = \frac{2}{\pi} \int\limits_{Q_{min}}^{Q_{max}} F(Q) \sin QrdQ. \tag{4.23}$$

As can be seen from the difference between both calculated PDFs and from the calculated R-values, there is a significant difference in the medium and large r-range. The first coordination sphere is almost not affected (fig. 4.16). A very small difference is observed for the first maximum of the PDF, that corresponds to the distance between tetrahedral iron (Fe_A) and oxygen. With the idealized structure used for the simulations the interatomic distances between octahedral iron (Fe_B) and oxygen is not modified (max-

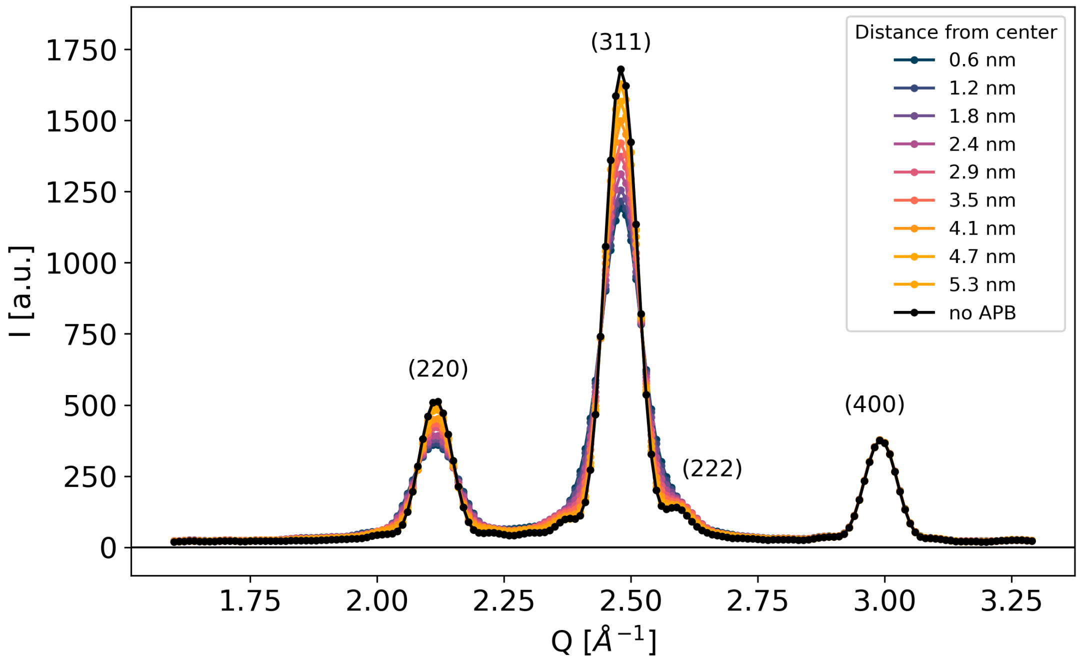


Fig. 4.14 Simulation of powder diffraction patterns for different positioning of the APB within a particle of 9.2 nm in diameter. The peak broadening depends on the position and is strongest for an APB through the particle center.

imum 2). The oxygen to oxygen distance as well as the distances between Fe_B atoms is also not affected by the APB (maximum 3). A slight increase in $G(r)$ due to the APB is visible for the maximum 4 corresponding to the interatomic distances between iron atoms on both sites as well as Fe_B-O and Fe_A-O (octahedral oxygen). However, all these differences are negligible compared to the strong deviations observed at larger interatomic distances. For both particles also the PDF decays to zero at the maximum interatomic distance possible, i.e. the particle diameter. This is also expected as the APB leaves the total particle size unchanged.

If a fit of the PDFs is performed with a small-box approach, as used by PDFGui[142], the features originating from the APB cannot be described, since in this approach a periodic arrangement of unit cells is assumed. This leads to a decrease in the fit quality especially at medium, i.e. larger than the first coordination sphere, to large interatomic distances, where the differences between the PDF with and without APB are strongest.

4.2.4 Summary

In this section it was demonstrated that the finite size broadening of peaks in X-ray powder diffraction patterns can be well described by the Fourier transform of the particle shape function. The APB was shown to produce a distinct hkl-dependent peak broadening that leaves certain peaks completely unchanged. This effect can be used to determine the presence of APBs in real samples. With the developed APB Fourier coefficient a correct description of the resulting peak profiles is achieved and the number of APBs can be assessed with the fitting parameter related to the APB probability. The value

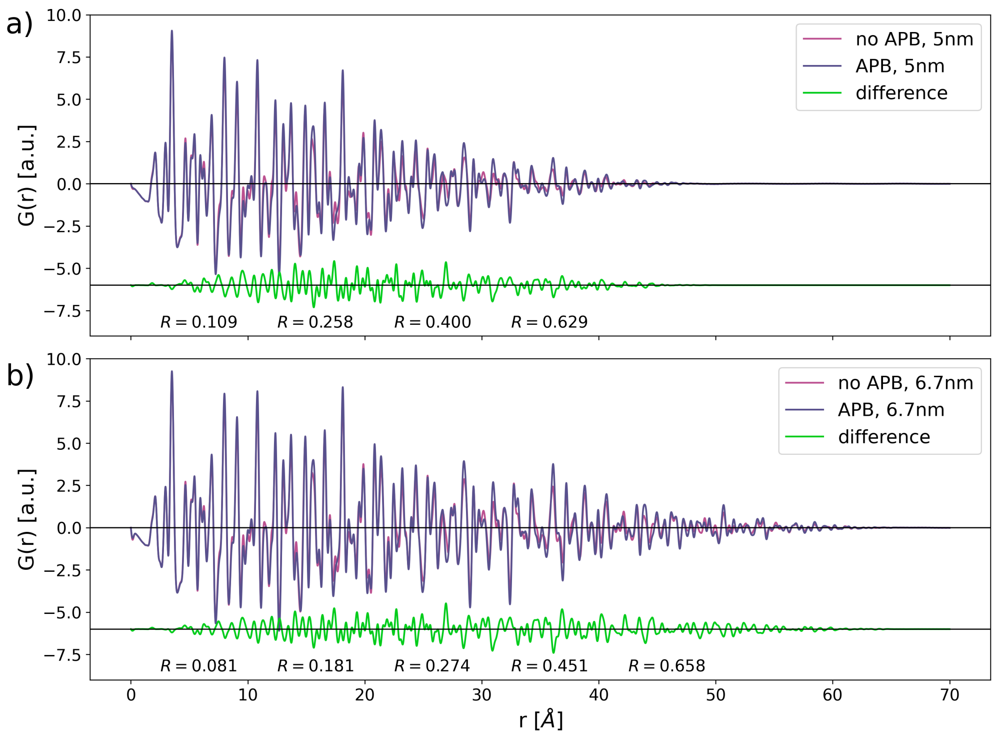


Fig. 4.15 Comparison of PDFs calculated from simulated powder diffraction patterns of particles with and without APB. The difference (no APB - APB) is shown below. Two particle sizes were simulated: a) diameter of $5\,\mathrm{nm}$ with 6103 (with APB) and 6106 (without APB) atoms in the structure and b) diameter $6.7\,\mathrm{nm}$ with $14\,527$ (with APB) and $14\,537$ (without APB) atoms. The first coordination sphere, i.e. the low r peaks are not significantly affected, however some difference becomes apparent at larger interatomic distances. R values are given below to quantify the difference in the curves.

for a single APB in the center of a spherical particle may provide a reference point for experimental studies on real samples. Finally, the calculation of PDF for particles with and without APB showed a significant difference in the curves, possibly complicating fits to real data from particles containing such defects.

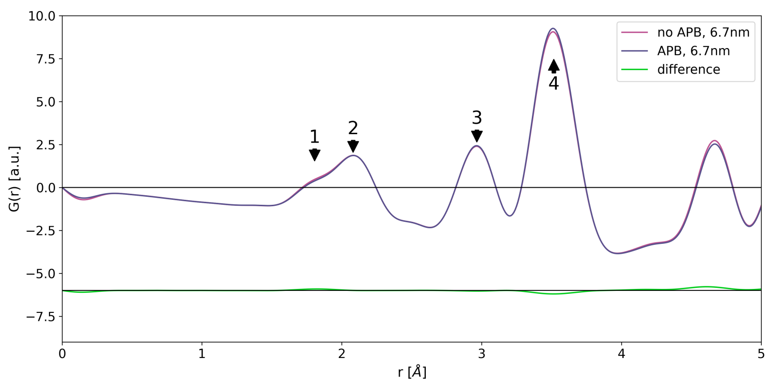


Fig. 4.16 Close up of the comparison of PDFs calculated from simulated powder diffraction patterns of particles with and without APB for 6.7 nm particles shown in fig. 4.15b). The difference (no APB - APB) is shown below. The maxima of the PDF marked with numbers correspond to interatomic distances between tetrahedral iron (Fe_A) and oxygen (1), octahedral iron (Fe_B) and oxygen (2), oxygen and oxygen as well as Fe_B-Fe_B (3) and Fe_A-Fe_B, Fe_A-Fe_A, Fe_B-O and Fe_A-O (4).

4.3 Experimental results for varying particle sizes

In this section the results of complementary experimental techniques applied to study particles of various sizes are presented. First an overview of the samples used and the applied techniques is given. A precise analysis of the inorganic core sizes and size distributions using SAXS and TEM follows. SANS is used to determine the organic shell and the non-magnetic surface layer thickness. With Mössbauer spectroscopy the particle composition is determined. X-ray powder diffraction and PDF analysis are used to determine the structural properties both on a larger scale and in the local environment. The presence of APBs is checked with the previously developed method (see section 4.2). This is complemented by HRTEM data. The combined information allows the evaluation and interpretation of the SQUID magnetometry data.

4.3.1 Overview of samples and techniques

The samples of spherical iron oxide nanoparticles used in this work were in part obtained from Ocean NanoTech LLC and also synthesized by Dr. Sascha Ehlert at JCNS-1 (Forschungszentrum Jülich GmbH). From Ocean NanoTech particles with nominal sizes of 5, 10, 15 and 20 nm were purchased. Two bottles from the same batch were obtained for each particle size, in the following labelled with B1 and B2. Two bottles with particles of nominal size 15 nm were available from a previous order, labelled with lowercase "b". From JCNS-1 two samples were used with sizes of around 12 nm (SE11 and SEs12). The sample names as well as the nominal sizes and further information are presented in tab. 4.2. All nanoparticles in this work are coated with oleic acid to prevent agglomeration by steric repulsion and are dispersed in toluene.

The samples SE11 and OC15b1 were used for SANS experiments on KWS-1 operated by JCNS at MLZ. Synchrotron X-ray powder diffraction experiments were performed on samples OC05B2, OC10B2, OC20B2 and SE11 at APS remotely and for samples OC05B2, OC10B1, OC15b1, OC15b2 and OC20B1 on site at PSI. SEs12 and OC15b1 were used for HRTEM measurements at ER-C (Forschungszentrum Jülich GmbH) with the help of Tanvi Bathnagar-Schöffmann. Additional TEM measurements were performed at JCNS-4 (Forschungszentrum Jülich GmbH) with the help of Dr. Marie-Sousai Appavou on samples OC05B2, OC10B2, SE11 and OC20B2. Mössbauer spectroscopy was performed by Dr. Joachim Landers (University of Duisburg-Essen) on samples OC10B1, OC15b1, SEs12 and OC20B1. SAXS measurements were performed for all samples at JCNS-2 (Forschungszentrum Jülich GmbH). SQUID magnetometry data was recorded at JCNS-2 as well, for the samples as indicated in tab. 4.2.

The particles of samples SEs12 were originally dispersed in tetrahydrofuran (THF). Since all other particles are dispersed in toluene a solvent exchange was attempted for sample SEs12. However, during this procedure the particles agglomerated making them unusable for the subsequent experiments. The results for sample OC15b1 have been published in *Nanoscale* under the title *"Mechanism of magnetization reduction in iron oxide nanoparticles"*.[84]

Tab. 4.2 Table of the samples used in this work. The nominal particle diameters in nm are given which might deviate from the actual particle sizes. The manufacturer abbreviations OC and SE in the sample names stand for Ocean NanoTech LLC and Dr. Sascha Ehlert, respectively. Mössbauer spectroscopy is abbreviated with Mössb.

Name	size	comment	SAXS	TEM	SQUID	Mössb.	SANS	XRD
OC05B1	5	Bottle 1	X		X			
OC05B2	5	Bottle 2	X	X				X
OC10B1	10	Bottle 1	X		X	X		X
OC10B2	10	Bottle 2	X	X				X
OC15b1	15	old batch bottle 1	X	X	X	X	X	X
OC15b2	15	old batch bottle 2	X					X
OC20B1	20	Bottle 1	X		X			X
OC20B2	20	Bottle 2	X	X		X		X
SEs12	12	aggregation during solvent exchange	X	X		X		
SE11	12	same synthesis procedure as SEs12	X	X	X		X	X

4.3.2 Particle sizes and size distributions

In this section the particle sizes and size distributions of the samples listed in the previous section are determined with SAXS and TEM. First, the smallest particles of sample OC05 are considered.

4.3.2.1 OC05

SAXS measurements were performed on two samples from each bottle of the purchased nanoparticles (OC05B1 and OC05B2 in tab. 4.2). Ocean NanoTech provided the nominal size as 5 nm. For the SAXS experiments the as purchased nanoparticle dispersion

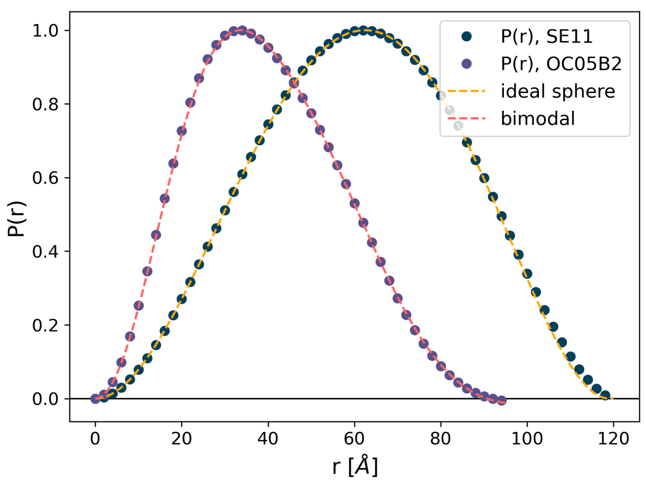


Fig. 4.17 Comparison of P(r) functions obtained from SAXS curves from OC05 and SE11. The curves have been normalized to their maximum value. The yellow dashed line corresponds to an ideal sphere with radius 5.9 nm. The dashed pink line is a fit with a superposition of two sphere models.

was diluted to 1 vol. % to reduce inter-particle interactions. The results are depicted in fig. 4.18a). The Guinier fit shown in the inset gives a particle radius of 3(1) nm. The linear behaviour suggests the absence of particle aggregates which indicates negligible inter-particle interactions. However, from the scattered intensity it can be seen that a simple sphere model is not able to fully describe the data. The dip in intensity at around $Q = 0.1\,\text{Å}^{-1}$ indicates the presence of two particle sizes or of particle clusters. A superposition of two sphere models with different particle sizes would capture all features yielding the same parameters for both bottles. Differences in total scale are due to slightly different concentrations resulting from the dilution of the original dispersion. The sphere models result in a majority contribution of particles with a radius of 2.4(1) nm and a smaller fraction (15(1) %) of particles with a radius of 3.9(1) nm. A single log-normal distribution parameter $\sigma = 0.12(1)$ was determined. However, as mentioned, it is also possible that not two particle sizes are present but that particles with diameters of 2.4(1) nm form clusters, i.e. two particles aggregate before the oleic acid can prevent this. To check this possibility the pair distribution function from the SAXS curve was calculated via

$$P(r) = \frac{r^2}{2\pi^2} \int_0^\infty Q^2 I(Q) \frac{\sin Qr}{Qr} dQ, \tag{4.24}$$

where r is the distance between electrons in Å. From the shape of this function conclusions can be drawn on the morphology of the particles. In fig. 4.17 two $P(r)$ curves are shown calculated from the SAXS curves of sample OC05B2 and SE11. The latter is highly monodisperse and thus leads to an almost symmetric $P(r)$ function with a central peak corresponding to the mean particle radius following the ideal case of a monodisperse sphere. The pair distribution function for a sphere is given by

$$P(r) = \left(1 - \frac{3}{2}\frac{r}{D} + \frac{1}{2}\left(\frac{r}{D}\right)^2\right) r^2, \tag{4.25}$$

with the particle diameter D. The term in the brackets is the shape function of a sphere as given in section 2.3.5.3. For sample OC05B2 a slanting towards smaller r leading to an asymmetric function shape is observed. A $P(r)$ function with two maxima is attributed to dumbbell particles.[224] However, for this sample only a small shoulder is visible. The $P(r)$ function can be reproduced either with strongly intersecting dumbbells or by assuming a superposition of particle sizes similar to the SAXS fit, here this yields particle radii of 2.5(1) nm with $\sigma = 0.09(1)$ and 4.1(1) nm with $\sigma = 0.09(1)$. The latter seems more likely since if two particles had aggregated to form a dumbbell the individual particle centres would be farther apart. Indeed, TEM images obtained for this sample clearly show that multiple particle sizes are present instead of dumbbell shaped particles (fig. 4.18b)). The two peaks in the particle size histogram match well with the obtained sizes from SAXS. Differences can be attributed to the limited statistical information provided by TEM as only 322 particles could be measured.

4.3.2.2 OC10

The next particles under consideration are those of samples OC10B1 and OC10B2 with a nominal core size of 10 nm according to Ocean NanoTech. For the experiments the as purchased nanoparticle dispersion was again diluted to 1 vol. % to reduce inter-particle interactions. Despite the dilution of the sample some particle aggregations are still present as evident from the deviation of linear behaviour in the Guinier plot shown in the inset on the lower left (fig. 4.19a). The slope of the scattering curve in this region shows a $Q^{-0.5}$ dependency. Ignoring this linear region a solid sphere model with a log-normal size distribution gives $R_0 = 5.8(1)$ nm with $\sigma = 0.10(1)$ for both bottles. Differences in the SAXS curves of the two samples are again due to concentration deviations originating from the sample preparation. From TEM a size distribution with parameters $R_0 = 5.5$ nm and $\sigma = 0.10$ is obtained in agreement with the SAXS results, however with a slightly smaller particle radius (fig. 4.19b). This difference is likely due to a region of smaller particles being sampled by TEM, whereas SAXS provides the average particle size of a much larger number of particles.

4.3.2.3 SE11 and SEs12

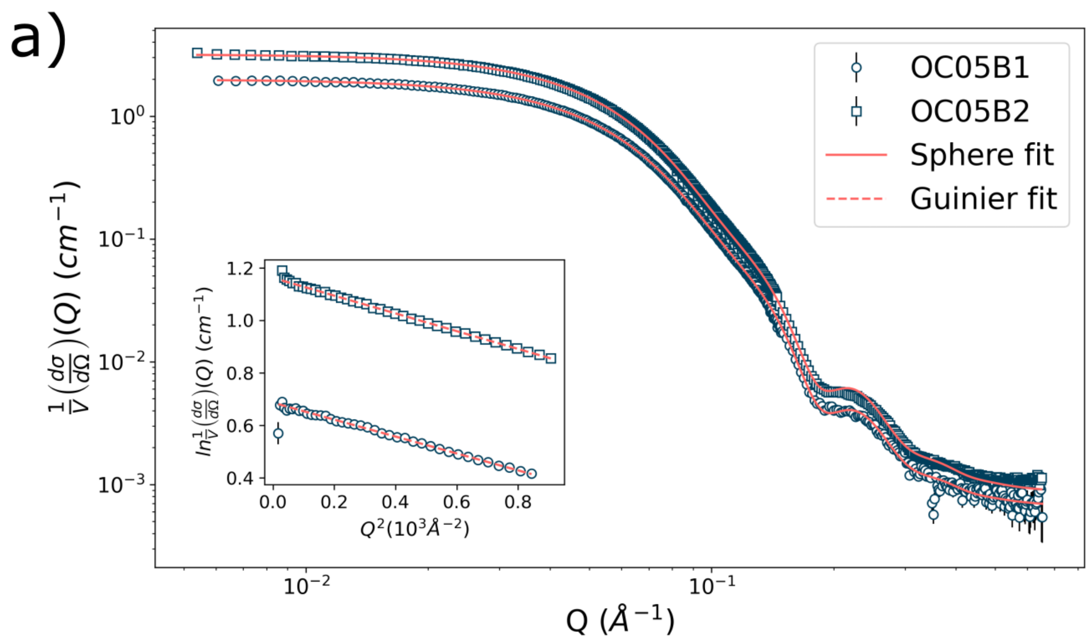
In the same way as for the previous samples a diluted dispersion of nanoparticles was used for a size determination of the particles of sample SE11. The fit is presented in fig. 4.20a). Here no deviation from linear behaviour can be seen in the Guinier plot shown in the inset. The solid sphere model fit with a log-normal distribution of particle sizes gives the parameters $R_0 = 5.9(1)$ nm and $\sigma = 0.06(1)$. This very narrow size distribution is also reflected in the large number of oscillations visible in the SAXS curve as well as the well defined minima. A comparison of the $P(r)$ function with that of an ideal sphere is shown in fig. 4.17. Here only very small differences can be observed. From TEM a log-normal size distribution with parameters $R_0 = 6.0(1)$ nm and $\sigma = 0.07(1)$ is obtained, in good agreement with the SAXS results (fig. 4.20b). These particles are very similar in size to the previously analysed OC10 sample ($R_0 = 5.8(1)$ nm), however the size distribution is narrower for SE11. As mentioned in section 4.3.1 the particles of SEs12 aggregated during the solvent exchange making the evaluation of SAXS data very difficult. TEM data suggest a mean particle radius of 6.6(1) nm with a log-normal size distribution parameter $\sigma = 0.05(1)$. The particles of SEs12 were synthesized under similar conditions as SE11 yielding slightly larger particle sizes. Mössbauer data is only available for sample SEs12 but conclusions may be drawn on the composition of particles of sample SE11, however the difference in size has to be taken into account.

4.3.2.4 OC15

The next sample in the size series is OC15 with a nominal core size of 15 nm. Again the inorganic particle size as well as the size distribution were determined via SAXS from a diluted dispersion of nanoparticles, where the original dispersion was diluted to 1 vol. %. The fit is presented in fig. 4.21a). No deviation from linear behaviour can be seen in the Guinier plot shown in the inset, indicating the absence of particle aggregation and thereby negligible inter-particle interactions. The radius of gyration, R_g, of 6.4(1) nm is obtained, which corresponds to a spherical radius of 8.1(2) nm in agreement with the particle radius determined from a solid sphere model with $R_0 = 7.8(1)$ nm and a log-normal distribution parameter $\sigma = 0.07(1)$. The different background levels for both curves are related to an overestimation of the particle number density in sample OC15b1, thus not enough of the solvent background has been subtracted. This background does however not influence the particle size determination. Similar to sample SE11 this very narrow size distribution is also reflected in the large number of oscillations visible in the SAXS curve as well as in the well defined minima. TEM data is in agreement with the SAXS results, however only 75 particles could be analyzed due to lack of larger scale images and thus the differences to the size distribution from SAXS may be attributed to insufficient statistics (fig. 4.21b). Still, the TEM images provide a confirmation of the spherical shape and particle size distribution.

4.3.2.5 OC20

Finally, the particles of sample OC20 are analyzed. Their nominal core size was given by Ocean NanoTech as 20 nm. For the experiments the as purchased nanoparticle dispersion was diluted to 1 vol. % to reduce inter-particle interactions. The results are depicted in fig. 4.22a). Differences in the scaling of the SAXS curves may again be attributed to concentration deviations originating from the sample preparation. A solid sphere fit provided the parameters of the log-normal distribution according to eq. 2.98, giving $R_0 = 10.7(1)$ nm with $\sigma = 0.09(1)$ for both bottles. The radius of gyration from the Guinier fit constitutes 9.5(2) nm, which corresponds to a particle radius of 12.2(2) nm. This is slightly larger than the value determined from the solid sphere fit, which is due to the small deviation from linear behavior at small Q in a $\log I$ vs. Q^2 plot. This indicates the onset of particle aggregation. From TEM images the parameters for the log-normal distribution a determined to $R_0 = 11.1(1)$ nm and $\sigma = 0.08(1)$ in agreement with the SAXS data, with a slightly larger radius (fig. 4.22b). This might be due to non-perfect spherical particle shapes that are not entirely correctly reproduced by the solid sphere model used for the SAXS data. Additionally, an error might be introduced by the calculation of the particle radius from the dark areas determined from TEM under the assumption of a perfect circle.



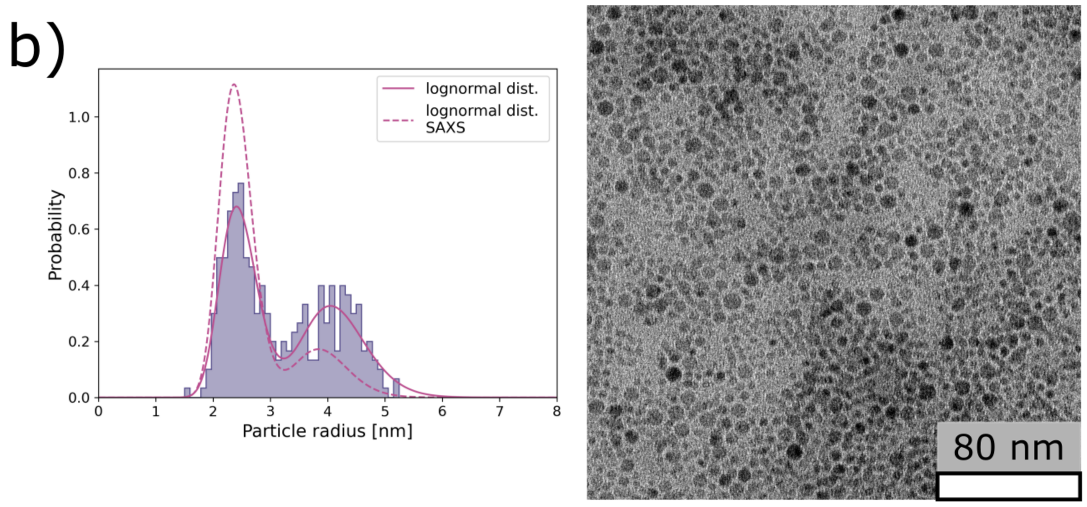
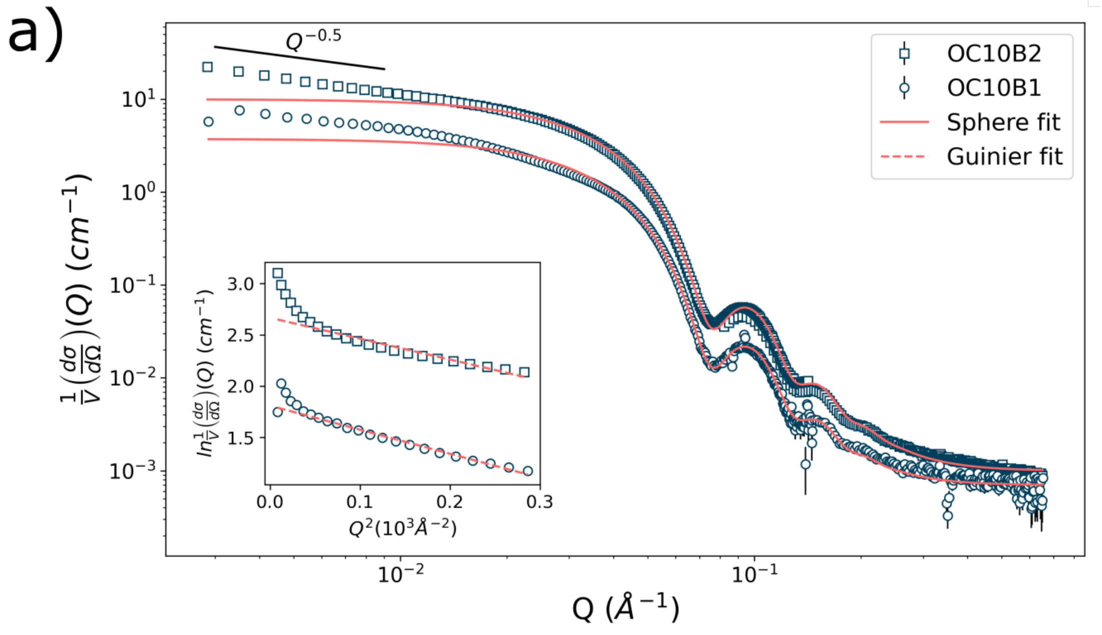


Fig. 4.18 a) Small-angle X-ray scattering curves for samples OC05B1 and OC05B2. A solid sphere model with two particle sizes was used for the fits (solid lines). The inset shows a $\ln(I)$ vs. Q^2 plot of the Guinier region with Guinier fits shown with dashed lines. b) Representative TEM micrograph obtained for sample OC05. On the left a histogram of 322 measured particles is shown. The solid and dashed line correspond to the log-normal distributions determined from SAXS and TEM, respectively.



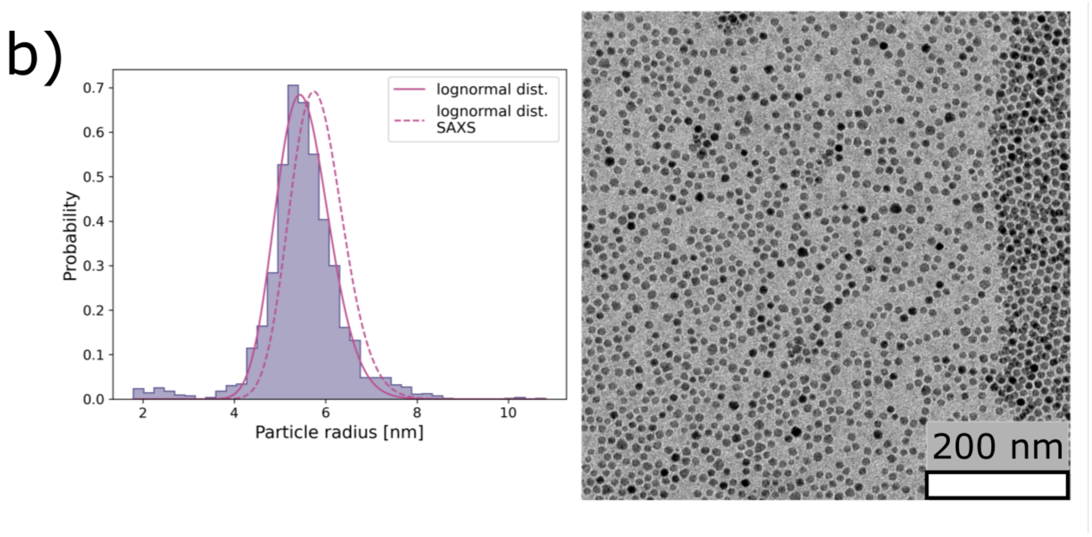
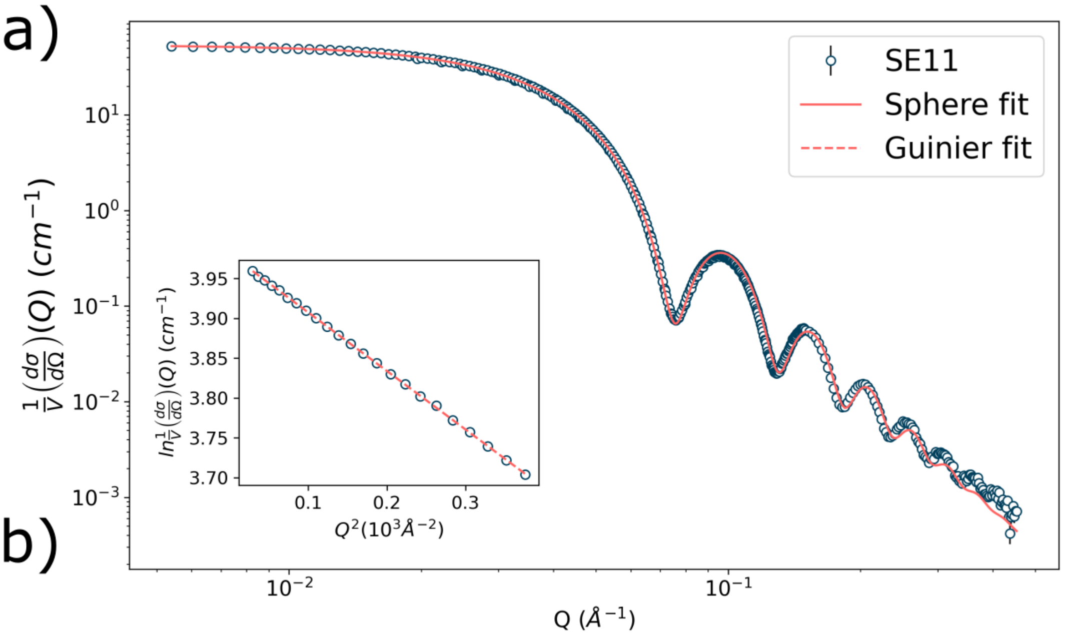
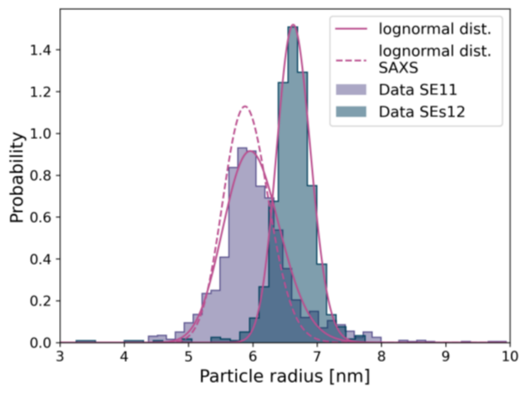


Fig. 4.19 a) OC10B1 and OC10B2 SAXS fit. A certain particle aggregation is present in the sample as seen from the slope of $Q^{-0.5}$ in the scattering curves and in the non-linear behaviour in the Guinier plot on the lower left. Excluding the small Q region from the fit allows the extraction of the particle size with a solid sphere form factor model. The fit parameters are given in the text. b) A representative TEM image of sample OC10 is shown in the right. The histogram of the particle sizes obtained from 2482 measured particles is given on the left with a solid line corresponding to the log-normal size distribution determined from the histogram and a dashed line that shows the distribution obtained from SAXS.

a)



b)



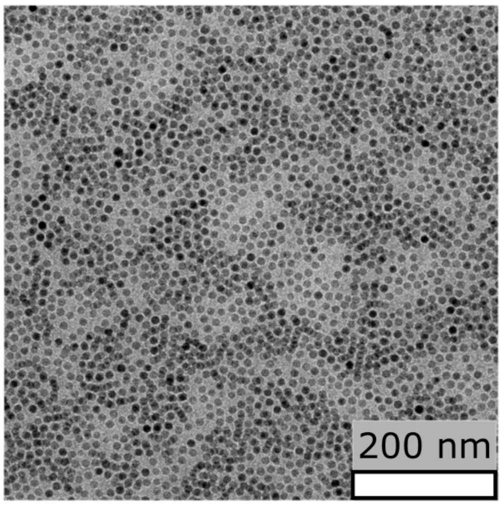
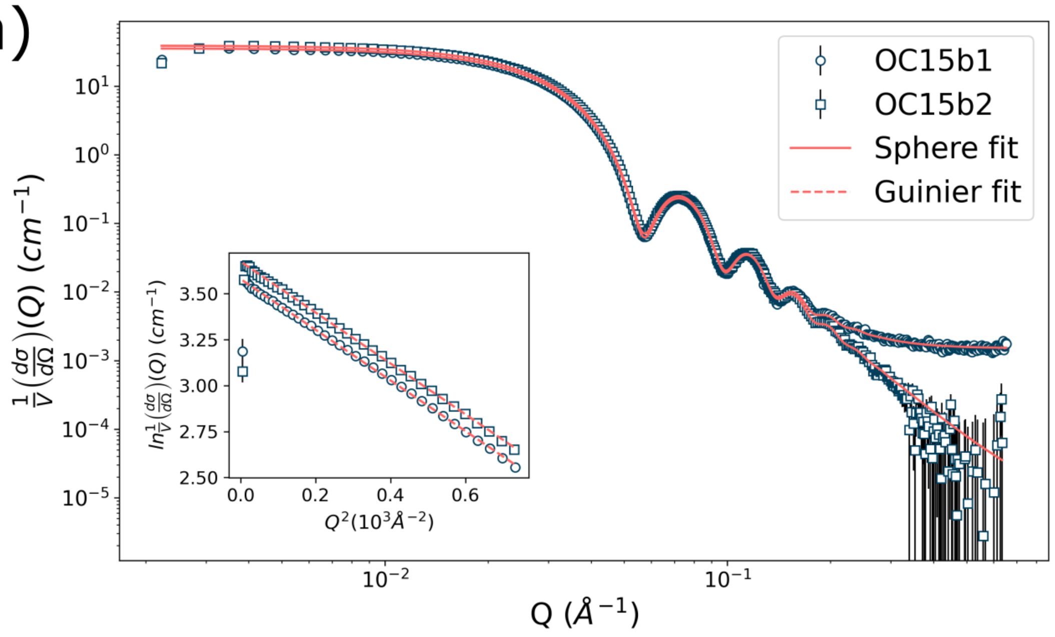


Fig. 4.20 a) SE11 SAXS fit with a solid sphere model with log-normal size distribution. The inset shows the Guinier region. b) On the left the histograms corresponding to the particle sizes determined from TEM images for samples SE11 and SEs12 are shown. For SE11 1445 particles were analysed, for SEs12 only 801. The solid lines are fits of a log-normal distribution to the data, the dashed ones correspond to the distribution determined from SAXS. On the right a representative image of the particles in SE11 is shown.

a)



b)

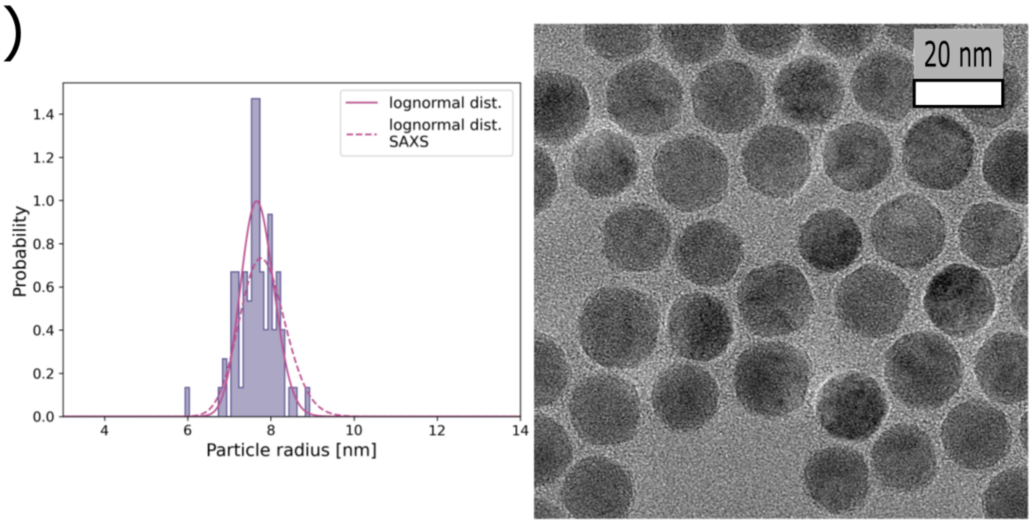


Fig. 4.21 a) OC15b1 and OC15b2 SAXS fit. Both curves result in the same fitting parameters of a log-normal size distribution with $R_0 = 7.8(1)\,\mathrm{nm}$ and $\sigma = 0.07(1)$. b) Histogram of the particle sizes in sample OC15 determined from 75 particles. On the right an image of the particles is shown. The dashed line shows the log-normal distribution obtained from SAXS, the solid line is a fit to the histogram data.

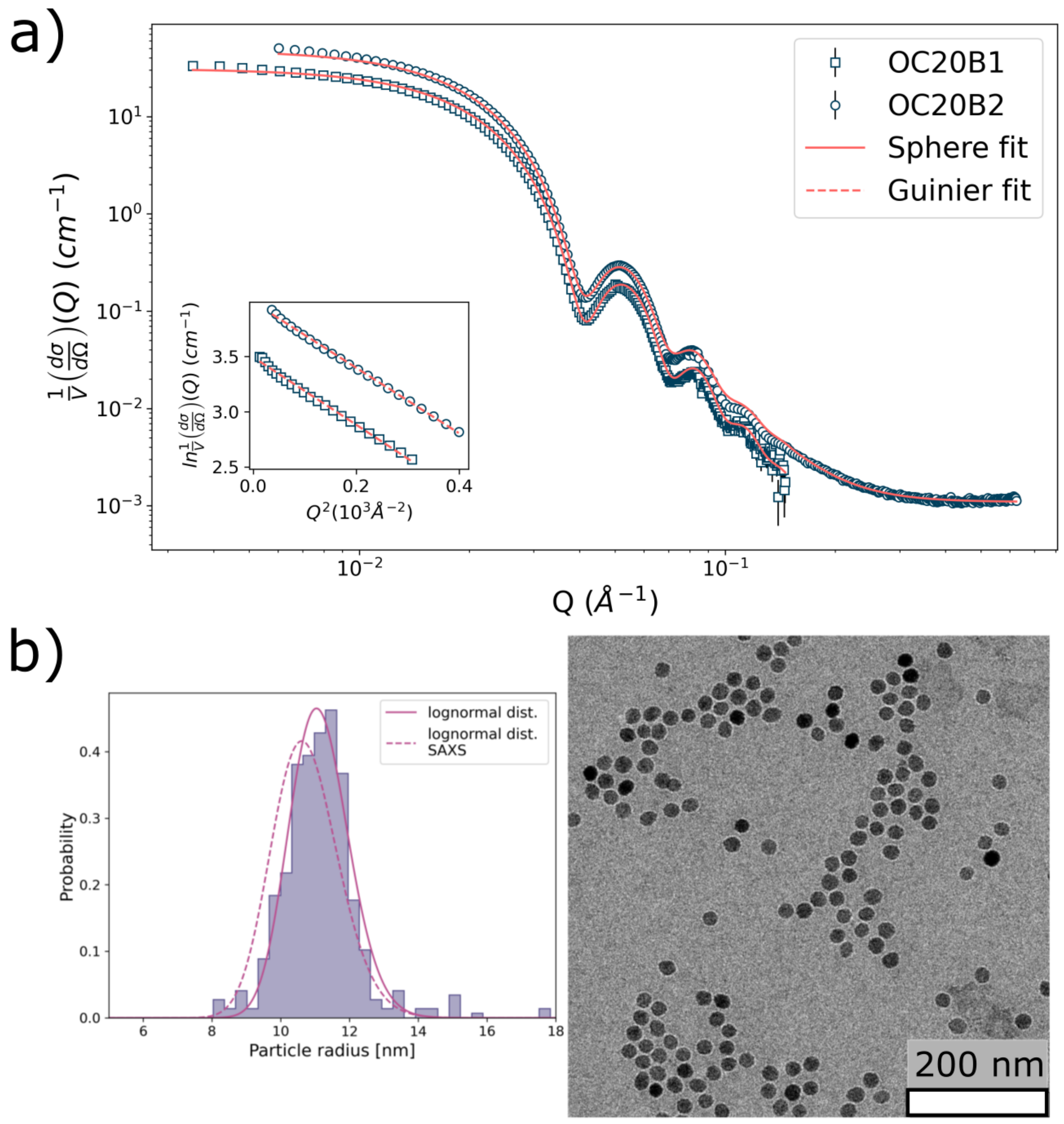


Fig. 4.22 a) OC20B1 and OC20B2 SAXS fit. Both samples result in the same core radius of $10.7(1)\,\mathrm{nm}$ with log-normal $\sigma = 0.085$. The difference in intensity of the curves is due to a different amount of particles in the sample, either because of a difference in concentration between the bottles or the diluted samples. b) Histogram of the particle sizes in sample OC20 determined from 450 particles. On the right hand side an image of the particles is shown. The solid line is a fit to the data and the dashed line corresponds to the results from SAXS.

4.3.2.6 Summary

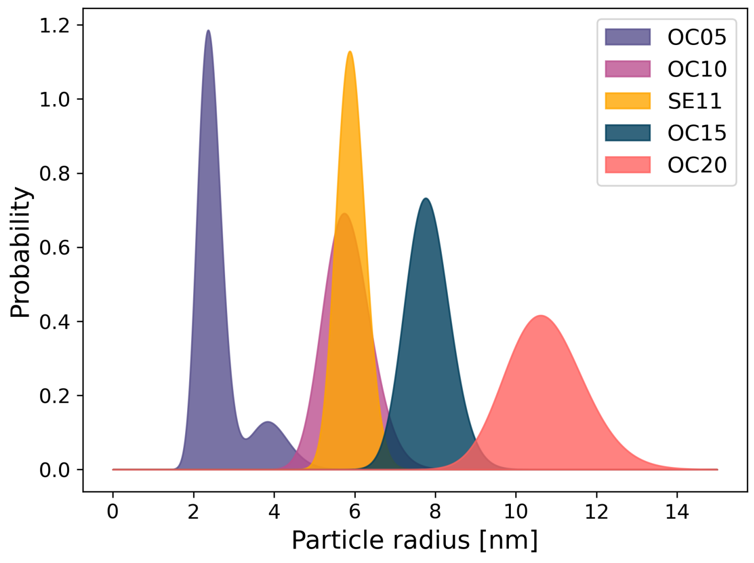


Fig. 4.23 Distribution of the particle sizes for the samples used in this thesis based on SAXS. The parameters of the distributions are given in tab. 4.3. The area beneath each curve is equal to 1.

The particle sizes and size distributions were determined by the use of SAXS and TEM, with consistent results. However, in the following the parameters as determined from SAXS are used since the statistics are superior compared to the analysis by TEM. The particle size distributions obtained from SAXS are shown in fig. 4.23 and listed together with the parameters obtained from TEM in tab. 4.3. For OC05 a bi-modal size distribution was found. The sample with the narrowest distribution of particle sizes is SE11. The broadest distribution is detected for OC20. However, all samples except OC05 can be considered reasonably monodisperse. For sample OC10 some particle aggregation is present even in the diluted sample used for SAXS. These results serve as the basis for the subsequent chapters.

Tab. 4.3 Particle sizes and lognormal size distribution parameter σ as determined from SAXS and TEM for the samples of this work.

Sample	R (TEM) [nm]	σ_{TEM}	R (SAXS) [nm]	σ_{SAXS}
OC05	55 % 2.5(1), 45 % 4.1(1)	0.13(1)	85 % 2.4(1), 15 % 3.9(1)	0.12(1)
OC10	5.5(1)	0.10(1)	5.8(1)	0.10(1)
SE11	6.0(1)	0.07(1)	5.9(1)	0.06(1)
SEs12	6.6(1)	0.05(1)	-	-
OC15	7.7(1)	0.05(1)	7.8(1)	0.07(1)
OC20	11.1(1)	0.08(1)	10.7(1)	0.09(1)

4.3.3 Particle composition

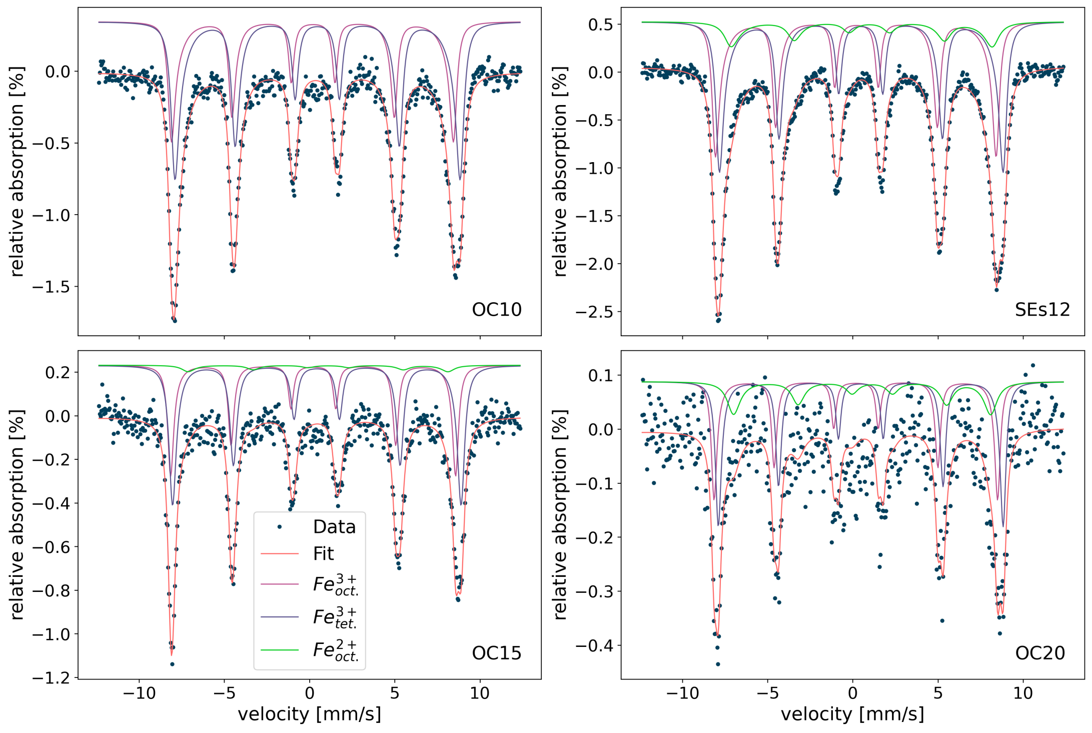


Fig. 4.24 Mössbauer spectra recorded for samples OC10, SEs12, OC15 and OC20 at $4.3\,\mathrm{K}$. For all samples the sextet splitting due to the hyperfine field can be observed. An Fe^{2+} contribution (green line) is only visible for samples SEs12, OC15 and OC20. The individual sextet contributions corresponding to the different iron cations in different lattice positions are offset for better visibility.

In addition to the particle sizes and the size distribution another important property to be considered is the chemical composition of the particles, especially the relative amounts of the various iron oxide phases presented in section 2.6. As established in section 2.5, Mössbauer spectroscopy is particularly well suited to distinguish the different phases. The presented data was recorded and analysed with the help of Dr. Joachim Landers (University of Duisburg-Essen).

Low temperature ($4.3\,\mathrm{K}$) Mössbauer spectra for samples OC10, SEs12, OC15 and OC20 are shown in fig. 4.24. For samples OC05 and SE11 no data is available. The spectrum of OC10 (upper left) can be modelled by two sextets corresponding to Fe^{3+} ions on octahedral and tetrahedral lattice sites. This suggests that no Fe^{2+} ions are present indicating that these particles consist entirely of maghemite. For SEs12 a relatively strong contribution of the Fe^{2+} sextet is detected, corresponding to a magnetite contribution of about $40\,\%$. As shown in the previous sections the particles of sample SE11 produced by the same method as SEs12 are slightly smaller. Thus for this sample the magnetite contribution is likely to be smaller, but can still be expected to be much larger than what was observed for OC10. The spectrum of OC15 can be reproduced again by a superposition of three symmetric sextets. The Fe^{2+} contribution yields an estimate of the magnetite content in the particle of $15\,\%$. For OC20 the highest

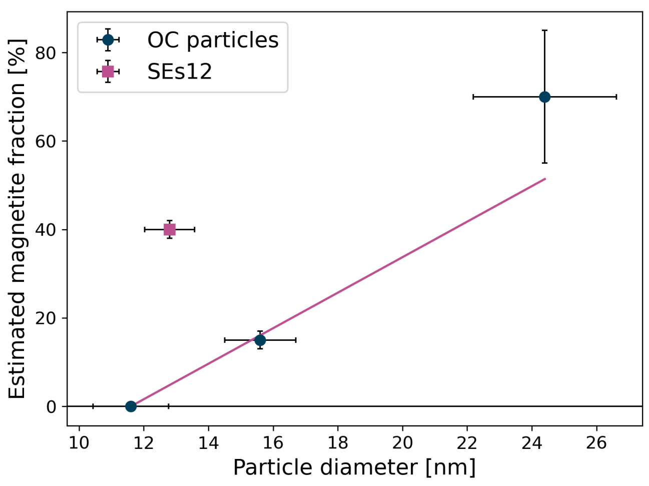


Fig. 4.25 Determined magnetite fractions drawn as a function of the particle diameters. The line illustrates the trend towards increasing magnetite content with increasing particle size for the samples obtained from Ocean NanoTech LLC.

magnetite fraction with 70 vol. % is obtained. However, the statistics for this sample is not as good as for the previous samples and the magnetite fraction is thus only a rough estimate. Still, it can be concluded that for these particles, which are the largest ones considered for this work the magnetite contribution is the highest compared to the other samples. This is also consistent with the model of nanoparticles oxidizing from the surface to the core by transforming magnetite to maghemite during the synthesis process. If equal oxidation rates are assumed for all particles more magnetite will remain in larger particles. The oleic acid coating seems to prevent further fast oxidation[225], however slower long-term oxidation has been observed for oleic acid coated iron oxide nanoparticles.[226] A wüstite contribution is not detected in any of the samples. A trend towards an increased magnetite content with increasing particle size can be concluded. However, it is apparent that the synthesis conditions have a strong influence on the degree of particle oxidation as the sample with different synthesis conditions (SEs12) shows a much higher magnetite contribution as expected from the trend observed for the particle obtained from Ocean NanoTech (fig. 4.25). For sample OC05 a composition of purely maghemite can be expected considering that this is reached already for particles of roughly 12 nm in diameter.

4.3.4 Structural properties

In this section the structural and crystallographic properties of the particles are addressed. X-ray powder diffraction is used to determine the lattice parameters and to draw conclusions on the vacancy distribution on a larger scale and the presence of antiphase boundaries. Where available HRTEM is used to support these findings. PDF analysis is applied to check for local disorder and determine the iron site occupancy. The analysis starts again with the smallest particle size and subsequently the larger particles are considered.

4.3.4.1 OC05

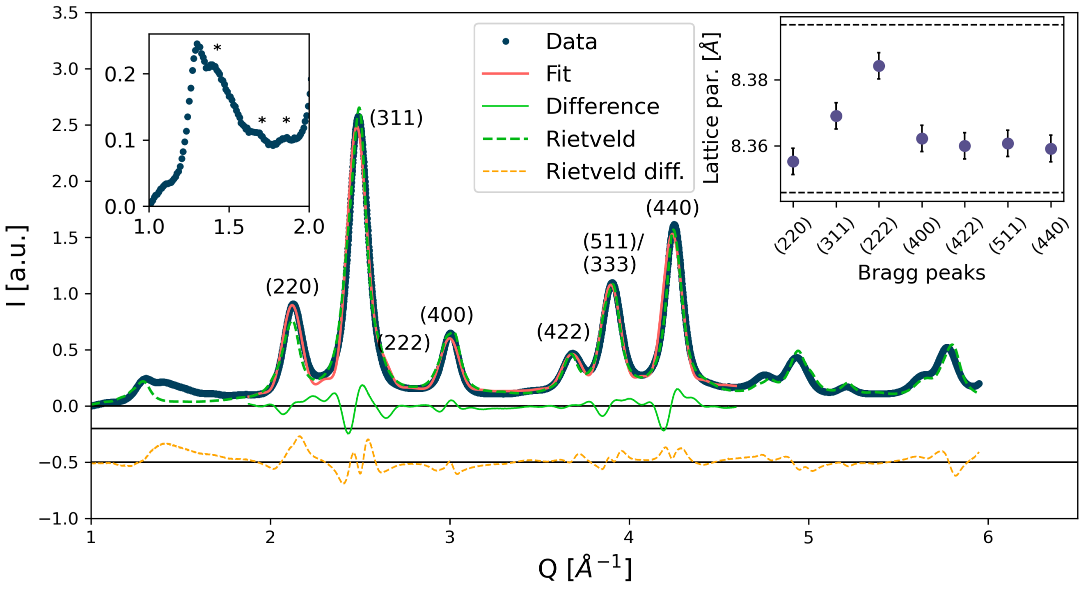


Fig. 4.26 X-ray powder diffraction data of OC05. The profile was fit with a superposition of peak profiles according to eq. 4.3. The inset on the right hand side shows the calculated cubic lattice parameter from each Bragg peak. The upper and lower dashed lines correspond to bulk magnetite and maghemite, respectively. The inset on the left hand side gives an enlarged view of the small Q region where the vacancy ordering superstructure peaks are marked with asterisks. A fit with the Rietveld method, performed with GSAS-II, is shown with the green dashed line. The corresponding difference between data and fit is given with the yellow dashed line.

In the powder diffraction pattern of OC05 (fig. 4.26) very small superstructure peaks very likely related to vacancy ordering are visible in the Q-range of 1 to 2 Å^{-1} (inset on the left hand side of fig. 4.26). The main peak positions translate into cubic lattice parameters close to the values expected for maghemite, albeit slightly larger. Deviations of the peak positions at small Q from the average structure indicate a strained lattice. No additional peaks relating to wüstite or other phases can be seen. The profile can be reproduced with a superposition of peak profiles according to eq. 4.3, with $\delta = 0.0$, i.e. no APBs are present, and a particle radius of 2.3 nm. The determined radius is slightly

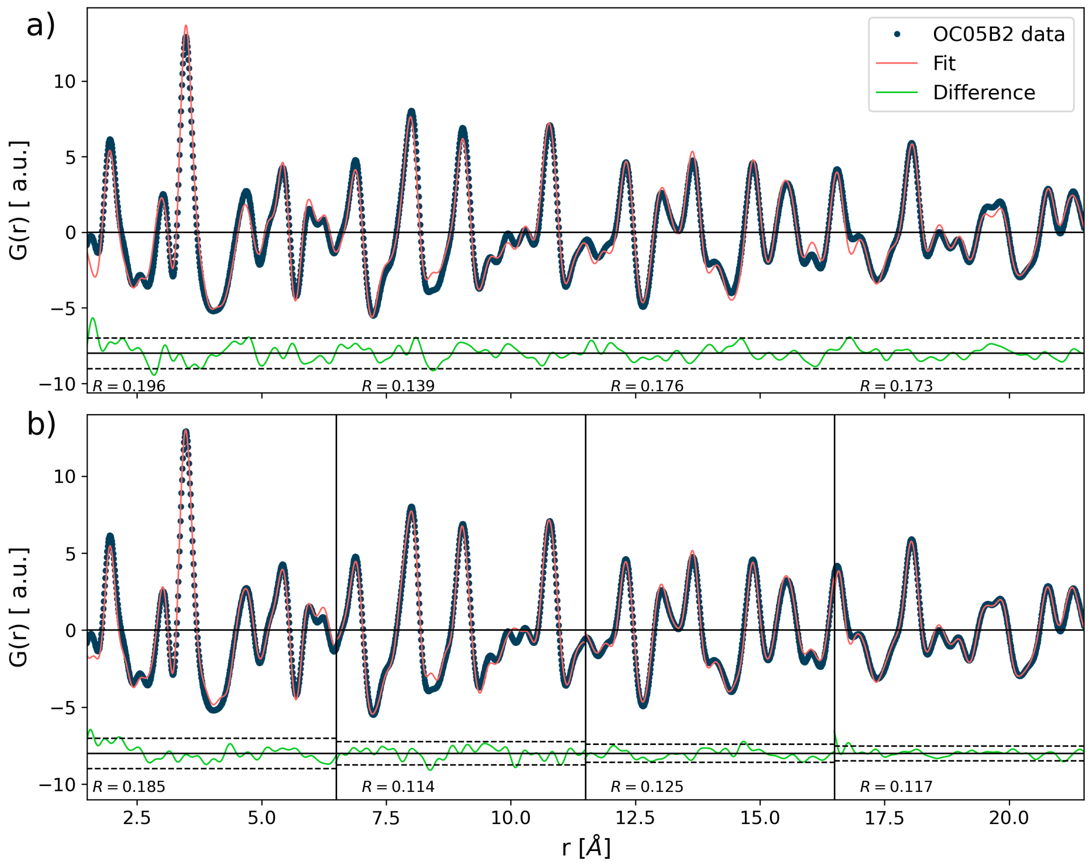


Fig. 4.27 a) PDF fit for sample OC05 using the tetragonal unit cell with space group symmetry $P4_32_12$ (see tab. 2.5). The difference between fit and data is given below, where dashed lines correspond to two standard deviations. The fit parameters are shown in tab. A.1. b) Fits to Q-ranges of $5\,\text{Å}$. The fit quality is given by the R-values in each section. The R-values corresponding to each section are also given for the total range fit in a).

smaller than that obtained from SAXS and TEM. This might be related to a residual oleic acid background and the presence of lattice strains.

A Rietveld fit to the Q-range of 1.6 to $6.0\,\text{Å}^{-1}$ (dashed green line in fig. 4.26 results in a strongly tetragonally distorted unit cell with lattice parameters $a = b = 8.256\,\text{Å}$ and $c = 8.386\,\text{Å}$, which is likely the cause for the deviations in small Q observed for the cubic lattice parameter determined from the peak positions (inset of fig. 4.26). The Fe-site occupancies were constrained to reduce the number of refinement parameters, giving a total iron site occupancy of 0.82. To study the lattice distortions and iron site occupancies on a more local scale PDF analysis is performed in the following.

The best fit to the PDF data is achieved with the tetragonal unit cell $P4_32_12$ (fig. 4.27a). The fit parameters are given in tab. A.1. Different occupancies on the iron sites were considered, showing a preference of vacancies on the $Fe2$ and $Fe4$ sites with occupancy factors of 0.75(3) and 0.60(2), respectively. The occupancy factors for $Fe1$ and $Fe3$ are 0.98(2) and 0.95(2), respectively. This results in a total iron occupancy of 0.81(2), which is reduced compared to the theoretical value of 0.89 for maghemite but within errors of the value obtained from the Rietveld refinement. The refined unit cell parameters indic-

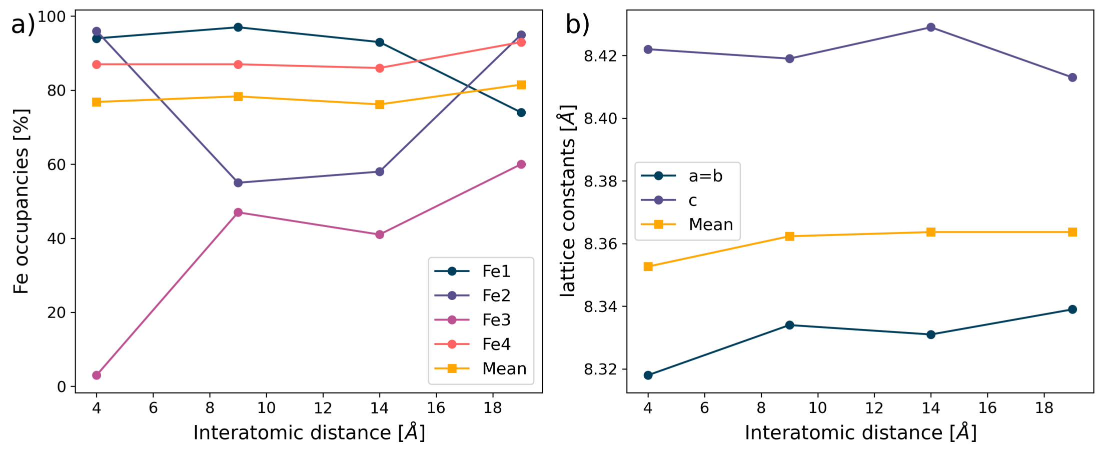


Fig. 4.28 OC05 Fe site occupancies (a) and lattice parameters (b) as determined from a PDF fit to intervals corresponding to different inter-atomic distances. The data points are drawn at the mean inter-atomic distance considered in each interval.

ate a stretched unit cell along the c-direction, resulting in a strained crystal lattice as was concluded from the analysis of the peak positions and was also seen in the Rietveld refinement. The fit does not capture all features perfectly, as is typical for a disordered structure. Thus, to gain a better understanding of the crystal structure a length-scale dependent fit was performed over intervals of 5 Å (fig. 4.27b). The largest mismatch between the model structure and the data is found for inter-atomic distances below 6.5 Å, indicated by a larger R-value. This suggests disorder in the first coordination spheres that averages out over larger inter-atomic distances. This structural disorder may be related to surface effects as well as lattice rearrangements around vacancies. The resulting changes in the bond lengths and angles may also affect the spin structure and subsequently reduce the magnetization.

Additionally, a refinement of the Fe-site occupancy factors shows that at this length scale strong vacancy ordering can be observed (fig. 4.28a), where vacancies are preferentially placed on octahedral $Fe3$ sites. Upon increasing the length scale, i.e. performing the fits at larger r a decrease in this vacancy ordering can be seen. This suggests that within the particles there exist spatially confined regions with high vacancy ordering while for the entire particle on average this ordering is less pronounced. The total iron site occupancy remains constant over all considered length scales. The lattice constants remain relatively unchanged over the considered r-ranges with a slight trend towards a less stretched unit cell (fig. 4.28b).

4.3.4.2 OC10

Similar to the previously considered sample OC05 also for OC10 only very small superstructure peaks related to vacancy ordering and subsequent symmetry reduction are visible in the powder diffraction pattern (inset on the left hand side of fig. 4.29). Analysis of the peak positions shows a high degree of variance in the associated cubic lattice constants indicating a strained crystal lattice, while the mean value would suggest the presence of some magnetite. However, no magnetite contribution was observed in both Mössbauer spectroscopy and SQUID-magnetometry. Taking also into account that even

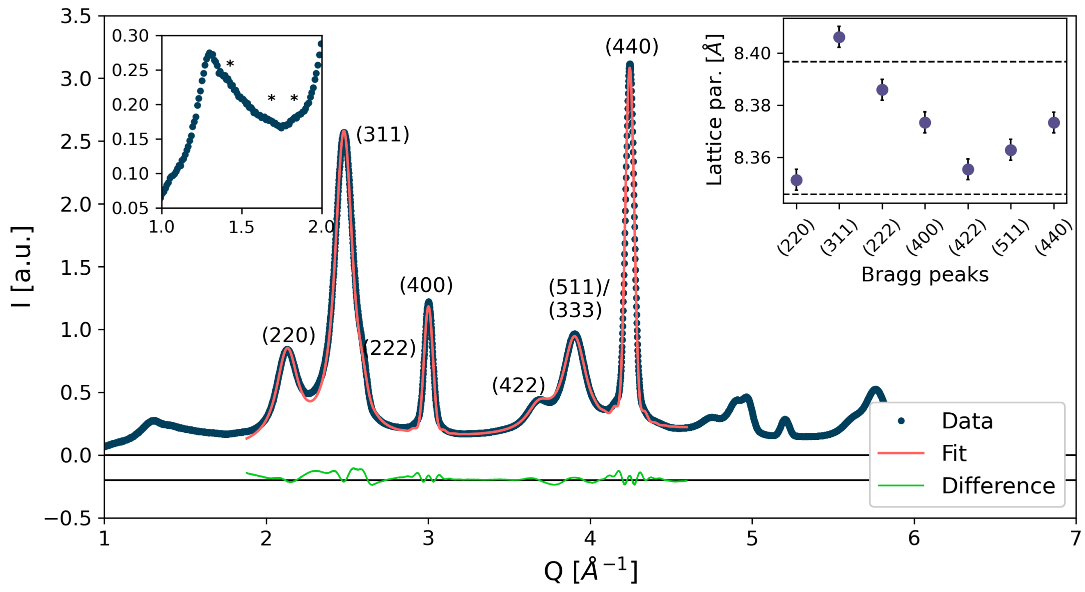


Fig. 4.29 X-ray powder diffraction data of OC10. The profile was fit with a superposition of peak profiles according to eq. 4.3. The inset on the right-hand side shows the calculated cubic lattice parameter for each Bragg peak. The upper and lower dashed lines correspond to bulk magnetite and maghemite, respectively. The inset on the left-hand side gives an enlarged view of the low-Q region, where very small superstructure peaks are marked with asterisks.

for the OC05 particles a lattice parameter larger than the bulk value was observed, the enlarged unit cell for the particles of OC10 can be attributed to lattice strain. Furthermore, no peaks related to the presence of wüstite are present. The best fit to the peak profile using eq. 4.3 is obtained with an APB probability parameter δ of 1.017. With eq. 4.22 this relates to 2.9(1) APBs. Some deviations are still present, which are likely related to additional lattice strains that have not been modeled.

The best fit to the PDF data is achieved with the tetragonal unit cell $P4_32_12$ (fig. 4.30). The fit parameters for a fit over the inter-atomic distance range of 1.5 to 21.5 Å are given in tab. A.2. Vacancy ordering persists up to the maximum inter-atomic distance considered of 21.5 Å, however, as mentioned before no significant superstructure peaks are visible in the powder diffraction suggesting that vacancy ordering is not present in the whole particle but confined to smaller regions on the particle surface or in the core separated by regions with random vacancy placement. In the ordered regions the vacancies are placed mainly on the octahedral $Fe2$ positions, while the other iron positions also show a reduced occupancy. The total iron occupancy evaluates to 0.69, which is strongly reduced compared to the theoretical value of 0.89 for maghemite. In addition the unit cell is elongated along the c-direction. A fit to sections in different r-ranges is shown in fig. 4.30b). Contrary to what was observed for the smaller particle size considered in the previous section here the fit quality gets worse with increasing inter-atomic distance. This can be attributed to the deviations introduced by the APBs. As seen from fig. 4.15 differences between the ideal PDF and the one obtained from a particle with an APB through the center become strong at atomic distances of approx.

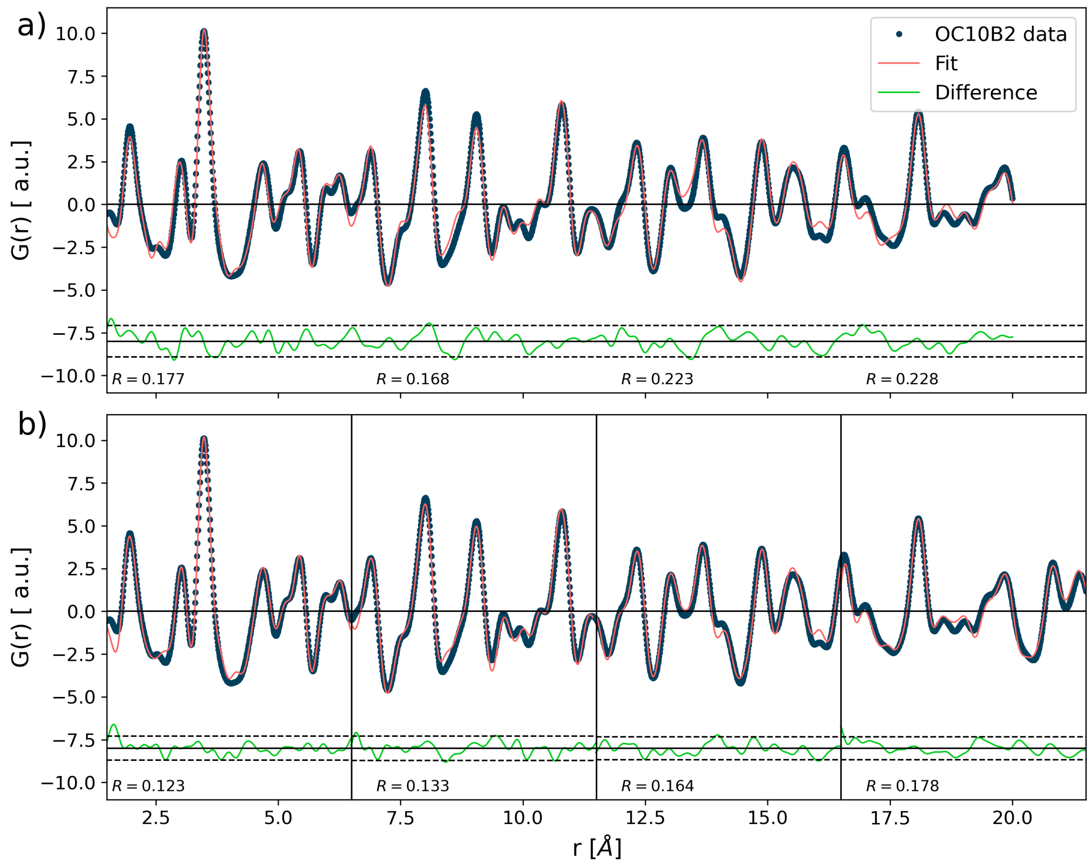


Fig. 4.30 a) PDF fit for sample OC10 using the tetragonal unit cell with space group symmetry $P4_32_12$ (see tab. 2.5). The difference between fit and data is shown below, where dashed lines correspond to two standard deviations. The fit parameters are given in tab. A.2. b) Fits to Q-ranges of 5 Å. The fit quality is given by the R-values in each section. The R-values corresponding to each section are also given for the total fit in fig. a).

8 Å, i.e. starting to become significant in the second r-section considered. Additionally, some local disorder can also be observed in the first r-section, which is slightly less pronounced than for the smaller particles.

4.3.4.3 SE11

In the powder diffraction pattern (fig. 4.31) no superstructure peaks related to vacancy ordering are visible. Very similar to sample OC10 the cubic lattice parameters obtained from the low angle peaks show a large variance indicating a strained or stretched lattice. Here the mean lattice parameter is closer to the theoretical value of magnetite, consistent with the previous considerations on the magnetite content. Again, no peaks related to the presence of wüstite can be found. Analysing the pattern with eq. 4.3 results in the best fit by using an APB probability parameter δ of 1.017, which using eq. 4.22 leads to a number of APBs of 3.0(1) that is comparable to the value obtained for sample OC10. To fit the PDF again the tetragonal unit cell with space group $P4_32_12$ is chosen (fig. 4.32a). The fit parameters for a fit over the inter-atomic distance range of 1.5

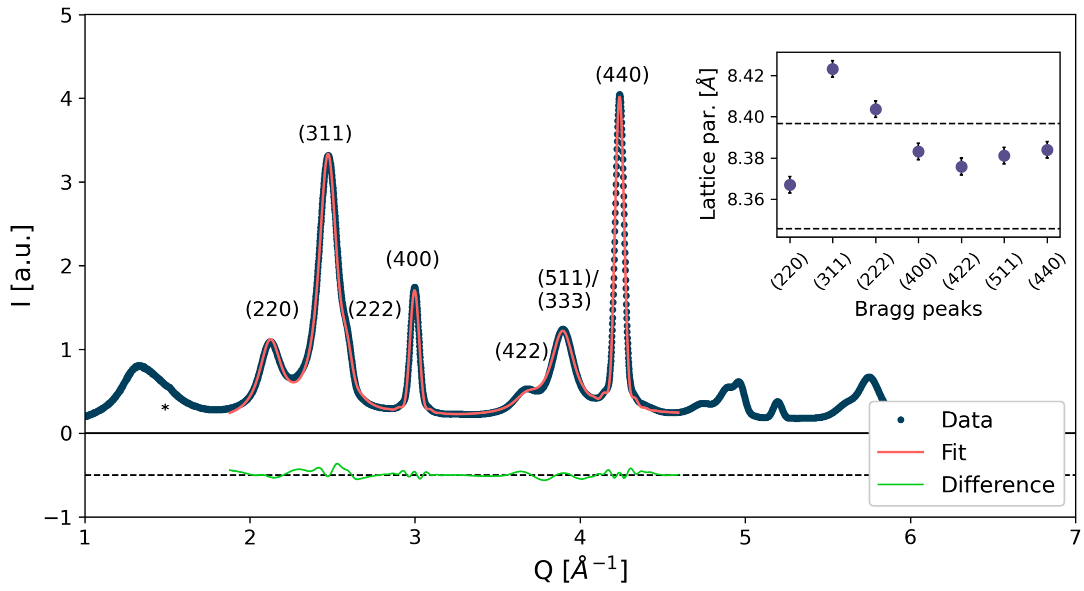


Fig. 4.31 X-ray powder diffraction data of SE11. The profile was fit with a superposition of peak profiles according to eq. 4.3. The inset shows the calculated cubic lattice parameter for each Bragg peak. The upper and lower dashed lines correspond to bulk magnetite and maghemite, respectively.

to 21.5 Å are given in tab. A.3. Vacancy ordering can be seen up to the maximum inter-atomic distance considered of 21.5 Å, however, as mentioned before, no significant superstructure peaks are visible in the powder diffraction suggesting that, very similar to sample OC10, vacancy ordering is not present in the whole particle but smaller regions on the particle surface or in the core are separated by regions with random vacancy placement. The vacancies in the ordered regions are placed mainly on the octahedral $Fe3$ positions, while the other iron positions also show a reduced occupancy. The total iron occupancy evaluates to 0.78, which is strongly reduced compared to the theoretical value of 0.89 for maghemite but larger than what was observed for OC10. In addition the unit cell is elongated along the c-direction. Again the influence of the APBs on the PDF can be seen from a smaller refined particle size, as well as an increase in the R-value for increasing inter-atomic distances (fig. 4.32b).

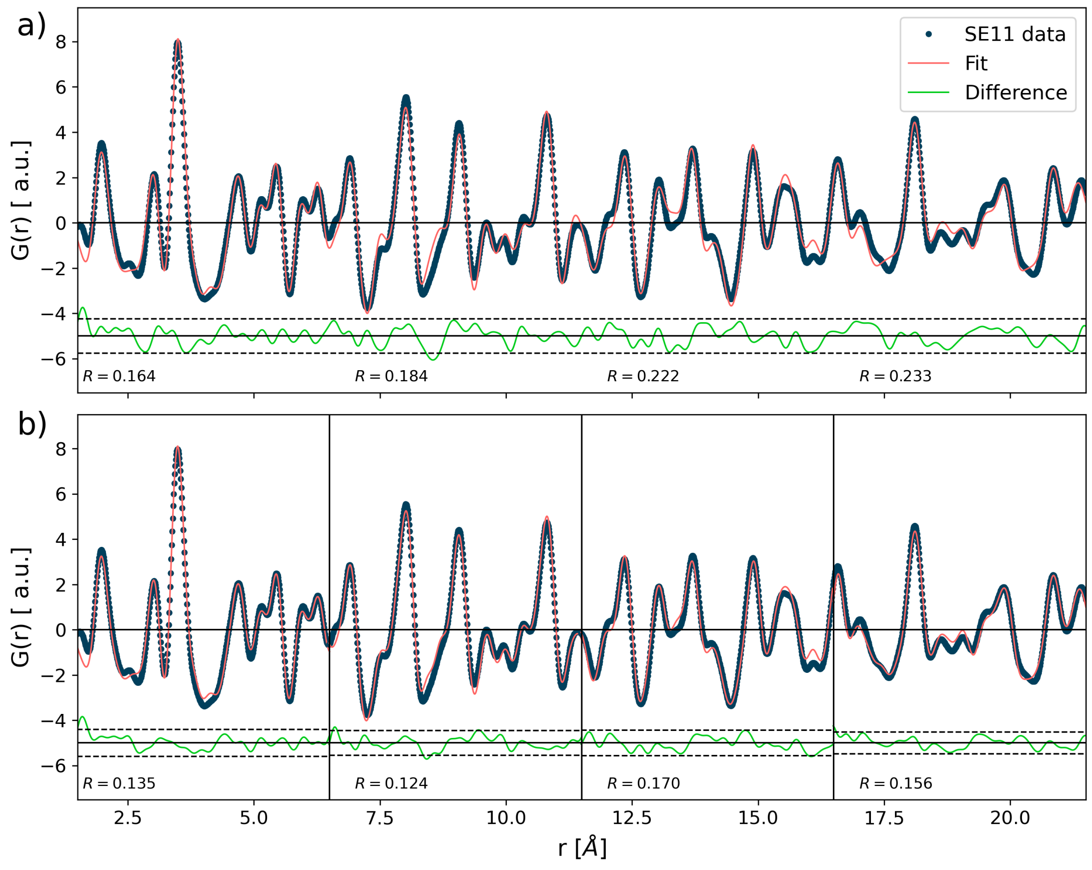


Fig. 4.32 a) PDF fit for sample SE11 using the tetragonal unit cell with space group symmetry $P4_32_12$ (see tab. 2.5). The difference between fit and data is shown below, where dashed lines correspond to two standard deviations. The fit parameters are shown in tab. A.3. b) Fits to Q-ranges of $5\,\text{Å}$. The fit quality is given by the R-values in each section. The R-values corresponding to each section are also given for the total fit in fig. a).

4.3.4.4 OC15

HRTEM data for these samples allow an additional complementary confirmation of the presence of APBs (fig. 4.33). The resulting iron sub-lattice shift can be seen in fig. 4.34. It is important to note that although it appears like bright spots correspond to atomic columns, image simulations show that for this particle thickness and the electron microscope settings dark spots relate to the atomic columns. The model structure with the APB then coincides reasonably well with the observed data. The APBs can be more easily identified by masking the Bragg peak of the lattice planes that are affected by this translation (lower row of fig. 4.33). E.g. for the left most image of fig. 4.33 the (220) reflection can be chosen (marked with the red circle in the inset). Masking instead the (440) peak leads to no observable lattice shifts, in agreement with the simulations and theoretical considerations on the peak broadening in the XRD pattern (section 4.2). Here, the peak (440) is not affected by the APB while the peak (220) is broadened. A drawback of relying only on HRTEM is the limited statistical information that can be

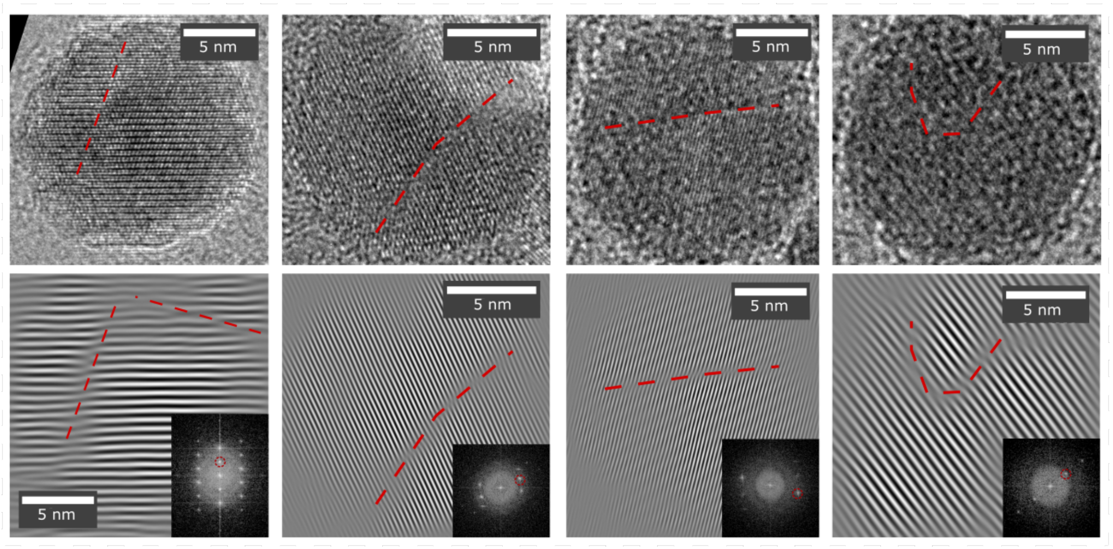


Fig. 4.33 HRTEM micrographs of nanoparticles of sample OC15 containing APBs as indicated with the red dashed lines. To make the lattice shifts more apparent below the inverse Fast Fourier Transformed (iFFT) images are shown. These are obtained by masking a certain peak (marked with the red circle) in the fast Fourier transformed image and performing the iFFT.

obtained. Such statistical information can be obtained from the analysis of the peak broadening in XRD as was done for the previous samples as well.

A part of the powder diffraction pattern measured at SLS on the beamline X04SA (section 3.2.1) is shown in fig. 4.35. Clearly superstructure peaks related to vacancy ordering are present, indicated with the asterisks. The cubic lattice parameters obtained from the major peak positions are more uniform than for the previously observed samples, however some variance in the parameters form the low-Q peaks remains. Again no wüstite peaks are present, in agreement with the Mössbauer results. A good fit to the experimental data is obtained by eq. 4.3, giving an APB probability parameter $\delta = 0.506$. This evaluates to 2.0(1) APBs using eq. 4.22, which is lower than what was observed for samples OC10 and SE11.

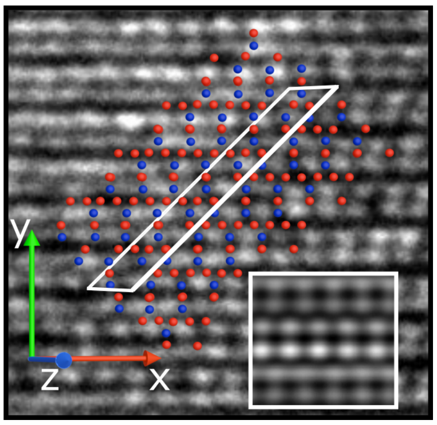


Fig. 4.34 Closeup of the APB structure in the particle shown in fig. 4.33 on the left. A model of the crystal structure is overlayed. The atom positions were verified by multislice TEM image simulations shown in the inset. The APB-plane is indicated with the white rectangle. Blue and red dots represent the tetrahedral and octahedral iron positions, respectively.

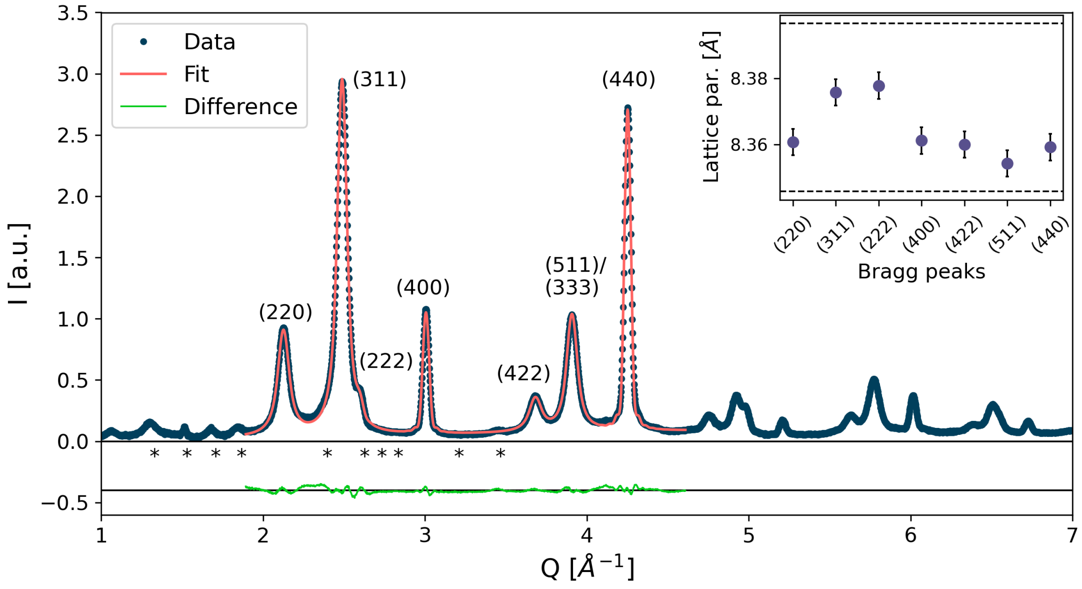


Fig. 4.35 X-ray powder diffraction data of OC15. The profile was fit with a superposition of peak profiles according to eq. 4.3. The inset shows the calculated cubic lattice parameter for each Bragg peak. The upper and lower dashed lines correspond to bulk magnetite and maghemite, respectively. Asterisks correspond to peak positions of superstructure peaks originating from vacancy ordering in the $P4_32_12$ unit cell.

As for the previous samples, the PDF is best reproduced using a tetragonal unit cell with space group $P4_32_12$ (fig. 4.36a). The parameters for a fit over a r-range of 1.5 to 21.5 Å are given in tab. A.4. The obtained lattice parameters are almost equal resulting in a pseudocubic unit cell. Vacancy ordering is observed with preferred placement on octahedral $Fe3$ sites. Some vacancies are also detected on the $Fe2$ position, while the tetrahedral $Fe1$ and octahedral $Fe4$ sites seem to be fully occupied. This results in a total iron site occupancy of 0.87(2), which is only slightly lower than the theoretical maghemite Fe-occupancy of 0.89. The presence of clear vacancy ordering superstructure peaks in the XRD pattern indicates that this vacancy ordering persists over a long range within the nanoparticles, i.e. the regions with vacancy ordering are much larger than for the previous samples. More detailed insights into the vacancy ordering are provided from the fit to sections of different inter-atomic distances shown in fig. 4.36b). From the R-values of the fit to the whole range it can be seen that the low r-section from 1.5 to 6.5 Å is not well described by the model. However, this is most likely due to a larger degree of vacancy ordering on this length scale and not due to structural distortions. Refining the Fe-site occupancies in this atomic distance range reveals a completely empty $Fe3$-site, while the other iron positions are almost fully occupied (fig. 4.37a). This is accompanied by an elongated unit cell along the c-direction (fig. 4.37b). It should be noted that this is a distortion on a local scale, while for the particle as a whole the difference in lattice parameters is very small as seen from the powder diffraction pattern. Upon increasing the inter-atomic distance r the $Fe3$ position gets gradually filled while simultaneously the occupancies of the other iron positions decrease slightly. Thereby the total iron occupancy remains constant. Correspondingly the lattice distortion is decreased showing

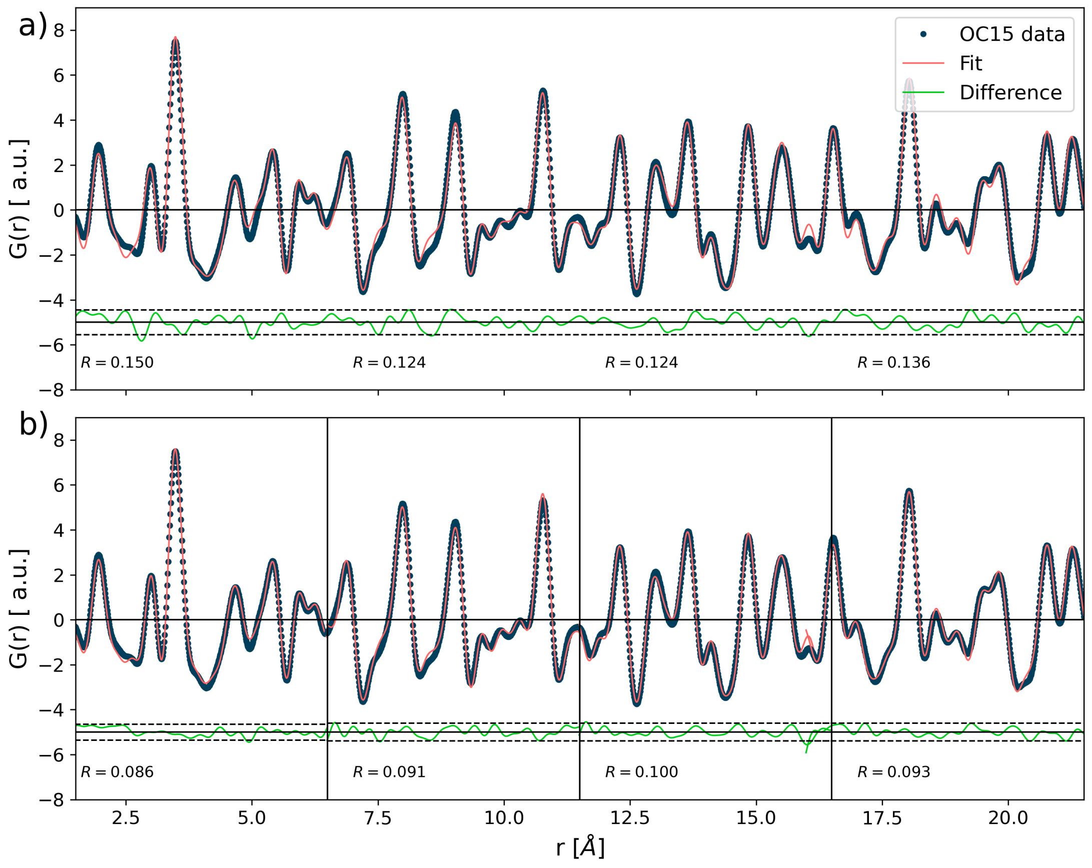


Fig. 4.36 a) PDF fit for sample OC15 using the tetragonal unit cell with space group symmetry $P4_32_12$ (see tab. 2.5). The difference between fit and data is shown below, where dashed lines correspond to two standard deviations. The fit parameters are shown in tab. A.4. For the generation of the PDF a maximum Q of $18\,\text{Å}^{-1}$ and $r_{poly} = 1.3\,\text{Å}$ were chosen. b) Fits to Q-ranges of $5\,\text{Å}$ with the R values for each section indicated at the bottom.

only a slight difference at the largest considered distances. It can clearly be seen that still at the largest considered inter-atomic distance vacancy ordering can be detected. Compared to the other samples the OC15 particles exhibit a much less distorted crystal structure, which is reflected in the better fit quality. As mentioned in section 4.3.1, these particles were already stored for several years, while the other samples were fresh. This might have allowed the crystal structure to relax more compared to the other samples.

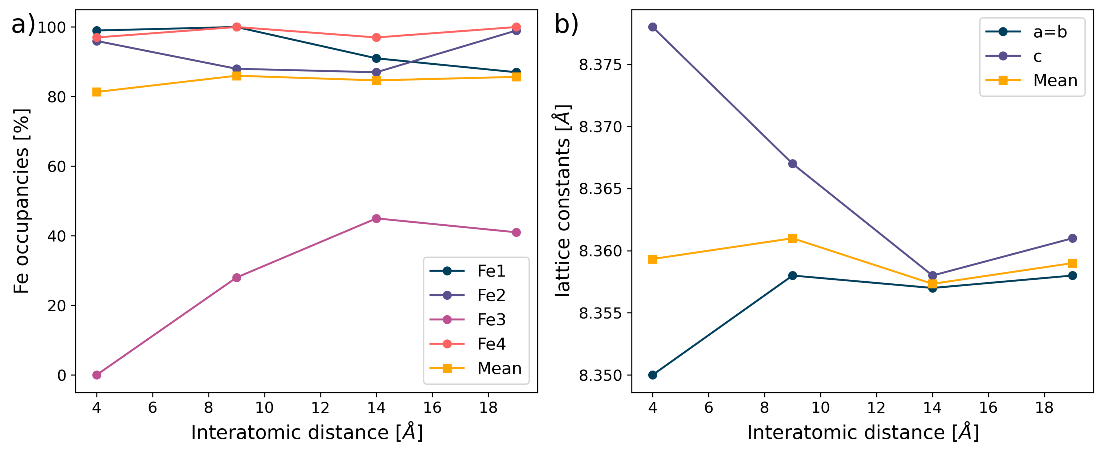


Fig. 4.37 a)Fe site occupancy of OC15 as determined from a PDF fit to sections corresponding to different inter-atomic distances (fig. 4.36b). b) Development of the lattice parameters with increasing inter-atomic distance.

4.3.4.5 OC20

The powder diffraction pattern is shown in fig. 4.38. Very strong vacancy-ordering superstructure peaks are visible at positions close to the theoretically expected positions for vacancy ordered maghemite/magnetite structures with space group $P4_32_12$. Regarding the presence of wüstite, only the small peak between peaks (222) and (400) in fig. 4.38 may be attributed to this phase. However, it is more likely that this is a superstructure peak related to vacancy ordering, since the peak is shifted to the left compared to the theoretical wüstite position and the new position could only be achieved by a larger wüstite unit cell. This is unlikely as the mismatch between the lattice parameters of magnetite and wüstite would be quite large. Additionally, if wüstite was present another peak should appear between peaks (511) and (440), which is not observed. The cubic lattice parameters obtained from the positions of the major peaks are shown in the inset. Again the deviation for the low angle peaks can be seen, hinting at a stretched unit cell. The average lattice parameter is closer to the theoretical value expected for magnetite, in agreement with the composition estimate from Mössbauer spectroscopy. From a fit to the major Bragg peaks clearly the hkl-dependent peak broadening is visible, where only some peaks are significantly broadened, while others remain relatively unaffected. By fitting the profile with eq. 4.3 the parameter $\delta = 0.664$ was determined, giving a number of APBs of 3.6(1) via eq. 4.22. This is the highest number of APBs of all the samples considered.

The best fit to the PDF data is achieved with the tetragonal unit cell $P4_32_12$ (fig. 4.39a), however the overall fit quality is not very good ($R = 0.198$), which is likely related to the large amount of APBs deduced from the significant peak broadening. Although superstructure peaks are strong in the XRD-pattern, the use of the $P4_12_12$ unit cell did not lead to a better fit. The fit parameters for space group $P4_32_12$ are given in tab. A.2. For the fit spanning an inter-atomic distance range of 1.5 to 21.5 Å the determined lattice parameters show a slight elongation of the unit cell along the c-direction, similar to the previous samples. The total iron site occupancy evaluates to 0.75(3), which is strongly reduced compared to the theoretical value of maghemite and even stronger reduced con-

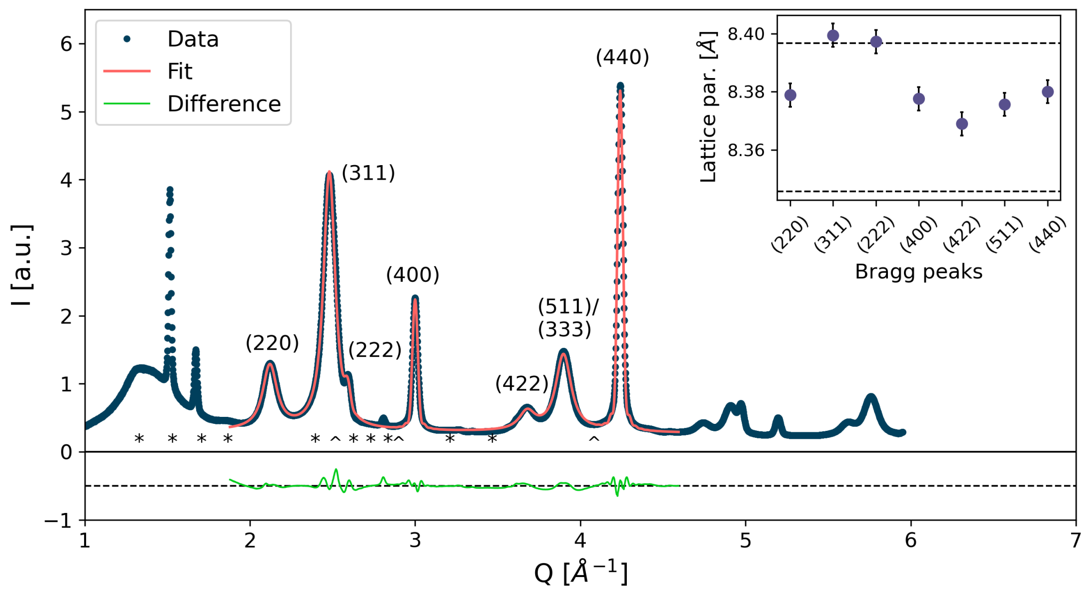


Fig. 4.38 X-ray powder diffraction data of OC20. The profile was fit with a superposition of peak profiles according to eq. 4.3. The inset shows the calculated cubic lattice parameter for each Bragg peak. The upper and lower dashed lines correspond to bulk magnetite and maghemite, respectively. Asterisks correspond to peak positions of superstructure peaks originating from vacancy ordering in the $P4_32_12$ unit cell. The hats show the theoretical peak positions of wüstite.

sidering a mixture of magnetite and maghemite. The site occupancies for all positions are reduced but a preference of vacancies on the $Fe3$-site can be seen. From a fit to sections of inter-atomic distances shown in fig. 4.39b) a trend of decreasing vacancy ordering with increasing inter-atomic distance can be observed (fig. 4.40 left). The difference in the lattice parameters decreases with increasing distance (fig. 4.40b), although not as uniformly as observed for sample OC15. This indicates that the crystal lattice is strained over a larger range. Additionally, large deviations between the model and the data are observed in the section corresponding to small inter-atomic distances (1.5 to 6.5 Å). This may indicate larger local disorder, likely resulting from stronger vacancy ordering than can be modelled with the space group $P4_32_12$. Locally the symmetry could be further reduced to the tetragonal space group $P4_12_12$.

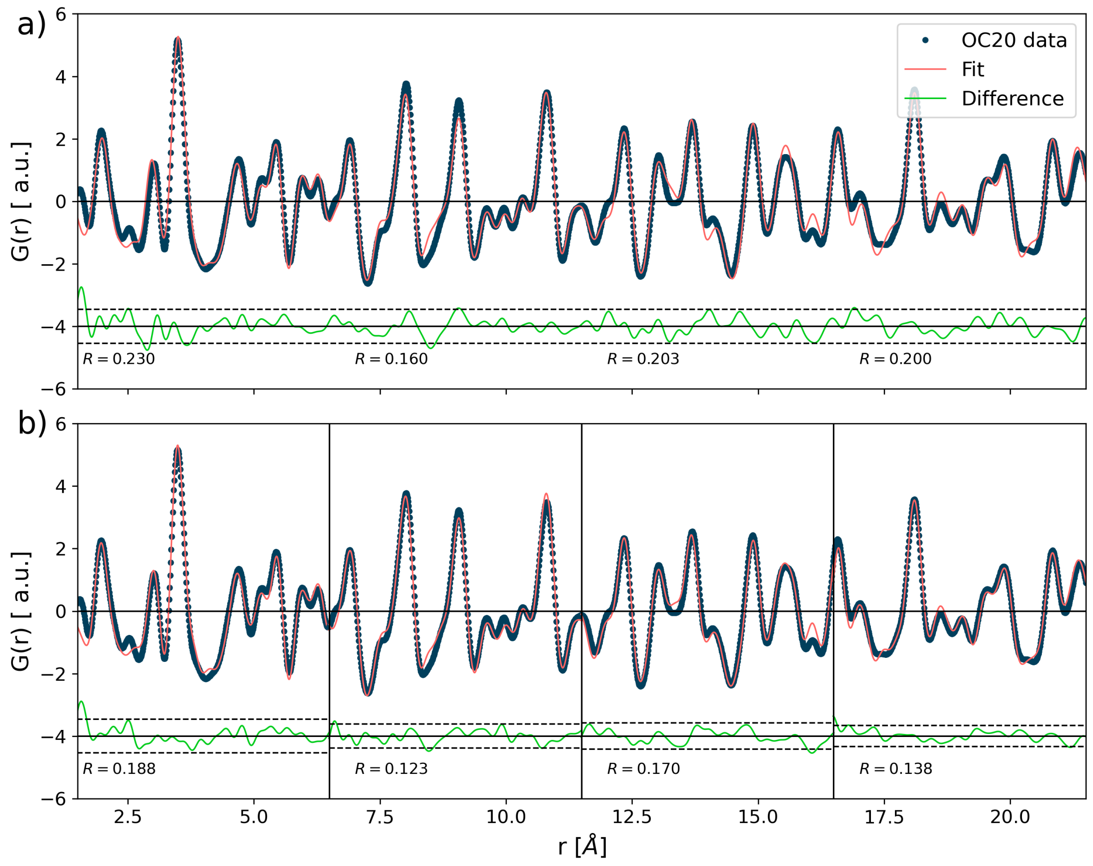


Fig. 4.39 PDF fit for sample OC20 using the tetragonal unit cell with space group symmetry $P4_32_12$ (see tab. 2.5). The difference between fit and data is shown below, where dashed lines correspond to two standard deviations. The fit parameters are shown in tab. A.5.

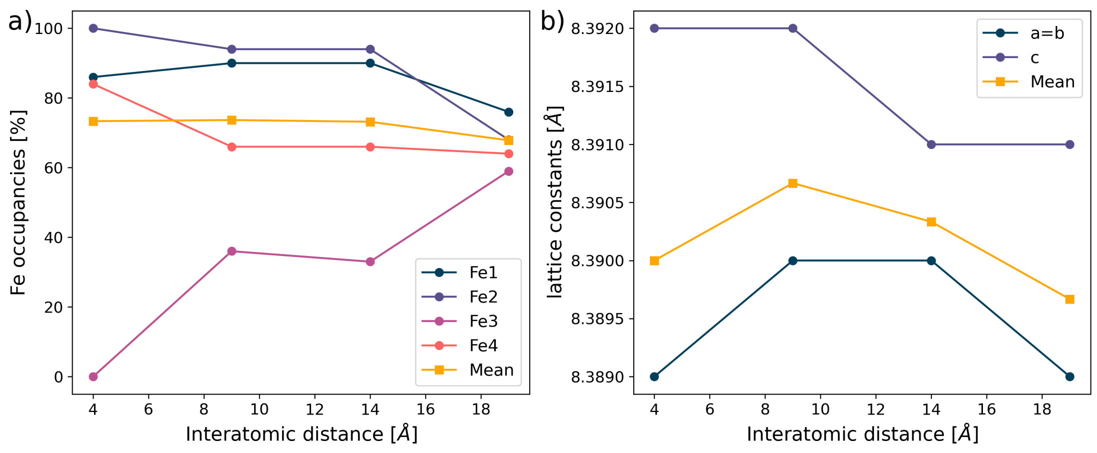


Fig. 4.40 a) Fe site occupancy of OC20 as determined from a PDF fit to sections corresponding to different inter-atomic distances (fig. 4.39b). b) Lattice parameters as a function of the inter-atomic distance.

4.3.4.6 Summary

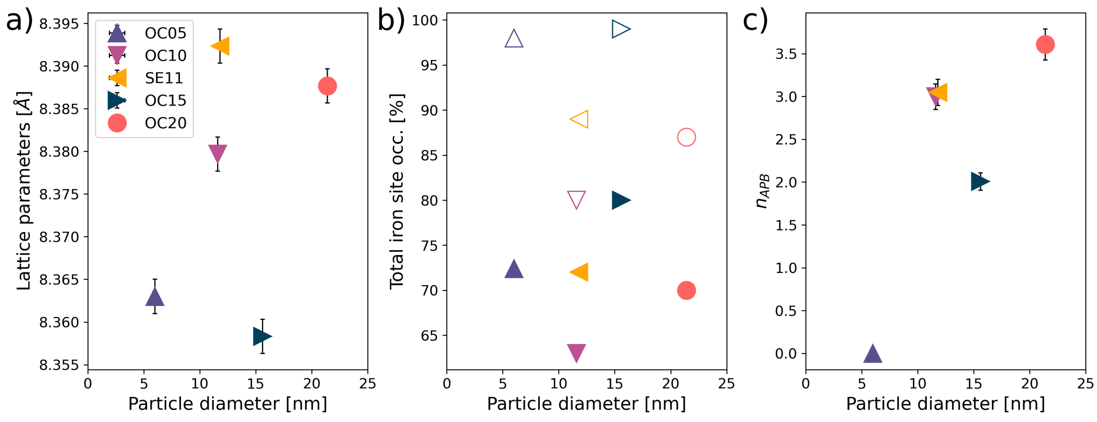


Fig. 4.41 Comparison of the structural properties. a) Cubic lattice parameters determined from PDF fits. b) Iron site occupancies determined from PDF fits. Open symbols correspond to tetrahedral iron positions, filled symbols - to octahedral positions. c) Number of APBs within the particles determined from fits to the powder diffraction data using eq. 4.22.

Consideration of the mean lattice parameters determined from the PDF analysis and the magnetite content estimated from the Mössbauer spectroscopy shows that there is some correlation between magnetite content and the lattice parameter (fig. 4.41a). However, sample OC10 shows an enlarged cubic lattice parameter, while no evidence for magnetite was found in the Mössbauer spectrum. A reason for this could be the significant local tetragonal distortions that were determined from PDF analysis and are in agreement with literature results of similar sized particles.[25] Such a large apparent unit cell is thus more likely a result of lattice strains. This would be supported by considering the particles of OC15, which show a smaller cubic lattice parameter than the other samples. In this case also the determined magnetite content from the lattice parameter matches that of the Mössbauer spectroscopy.

Also for sample OC20 the agreement between both methods is quite good, although the estimate from Mössbauer spectroscopy is rough due to the low signal-to-noise ratio. It can be concluded that the magnetite content increases with increasing particle size, however the lattice parameters alone are not sufficient to estimate the amount if significant lattice strains are present. The total iron site occupancy is determined via PDF analysis (fig. 4.41b). The lowest value is observed for sample OC10. All samples show a certain number of vacancies also on the tetrahedral lattice sites, but the majority is placed on octahedral sites. Different degrees of vacancy ordering are also detected with varying spatial extents. In the X-ray powder diffraction patterns the vacancy ordering superstructure peaks are strongly visible for samples OC15 and OC20 and to a much lesser degree also in the other samples. The inter-atomic distance dependent PDF refinements suggest the presence of vacancy ordered regions within disordered structures for the smaller particles of samples OC05, OC10 and SE11. For samples OC15 and OC20 these regions seem to be much larger possibly spanning the whole particle for sample OC15.

For the smallest particles no significant APB peak broadening is detected thus suggesting the lack of these defects for this particle size. Samples OC10 and SE11 show almost identical values, the determined value for OC20 is even larger. For OC15 the second lowest number of APBs is detected. For all samples with the exception of OC05 more than one APB is present in all particles (fig. 4.41c). These findings are similar to the ones reported by Levy et al.[158] Here, the presented XRD pattern clearly show the simulated APB effect (section 4.2). The authors focused, however only on the major peak (311) and mention that for particles ranging in diameter from 10 to 18 nm the coherent structure size is significantly reduced, while for smaller particles the determined structure size roughly matches that of TEM images. As shown in section 4.2 the peak (311) is affected by the APB broadening. Thus, this would indicate the onset of a significant APB contribution at a particle size of about 10 nm, in agreement with the data of this work. It has to be noted, however, that for the cited work a seeded growth method was used to synthesize the particles whereas for the particles of this work a thermal decomposition route was used.

4.3.5 SANS results

In this section contrast variation in SANS with polarized neutrons is used to accurately determine the organic shell thickness and the magnetic particle radius of samples SE11 and OC15, allowing conclusions on the presence and thickness of a magnetically dead surface layer. In the following a description of the fitting procedure is given that is based on the explanation given in the supplementary material to Köhler et al.,[84] which applies to both investigated samples.

To enable the determination of the thickness of a non-magnetic surface layer first the organic shell thickness has to be determined precisely. This is due to the small influence a non-magnetic surface layer has on the scattering intensity. The best contrast for the nuclear-magnetic interference term is achieved by a high deuteration of the organic solvent. This compensates the nuclear scattering length density of the core thus the sensitivity for the magnetic signal in the q-range of interest is enhanced. The largest contrast used in this work is 80 % deuterated toluene. A higher contrast would require the concentration of the original dispersion by evaporation of the non-deuterated solvent. However, this might introduce particle aggregation and was thus not chosen for this work. At this deuteration level also the contrast between the organic shell and the solvent is large. Therefore the organic shell thickness needs to be determined accurately prior to investigating the non-magnetic surface layer. The use of different solvent deuterations in a simultaneous fit finally gives more reliable parameters. To determine the organic shell thickness the parallel sector is used, i.e. a sector of 10° in the direction parallel to the applied saturation magnetic field (1.3 T for SE11 and 0.5 T for OC15). As shown in section 2.3.4.6, if $\boldsymbol{Q}$ is parallel to the applied field, i.e. $\alpha = 0°$ or 180°, only nuclear scattering contributes to the observed intensity. For sectors of 10° around this parallel orientation the magnetic contribution is still small enough to be neglected. The data for both spin channels is averaged to improve the statistics. A core-shell sphere model (eq. 2.133) was used in combination with a sticky hard sphere structure factor accounting for weak interactions between the nanoparticles induced by the magnetic field. The sticky hard sphere structure factor introduces three additional fitting parameters, which are the effective particle radius obtained by the addition of a small radius r_{shs} to the total particle radius, i.e. the inorganic core radius plus the organic shell thickness. Additionally, a volume faction and a stickiness parameter are used.

From simulations of the effect of these parameters it can be seen that for the 80 % scattering curve the volume fraction and the stickiness have no effect on the position of the first minimum (see appendix A). The radius r_{shs} has a small influence, however a change of this parameter changes the first maximum shape significantly. These observations are important as from this it can be concluded that the first minimum of the 80 % scattering curve provides a first estimate of the organic shell thickness that is not modified by the presence of the sticky hard sphere structure factor. The organic shell thickness thus obtained can be fixed at first and a simultaneous refinement of all scattering curves gives the parameters for the structure factor. The next parameters that are needed for an accurate determination of the non-magnetic surface layer thickness (t_{surf}) are the scaling factors for each curve. With these parameters the perpendicular sectors are finally used to obtain the nuclear-magnetic interference terms by subtracting the spin-down data from the spin-up data (eq. 2.113). Again the 80 % scattering curve is used at first, where only t_{surf} is refined in a core-shell sphere model used for the magnetic form factor. The subsequent incorporation of the other data yields the magnetic scattering length density (SLD_m). In the final step the surface layer thickness is refined simultaneously for all contrasts.

4.3.5.1 SE11

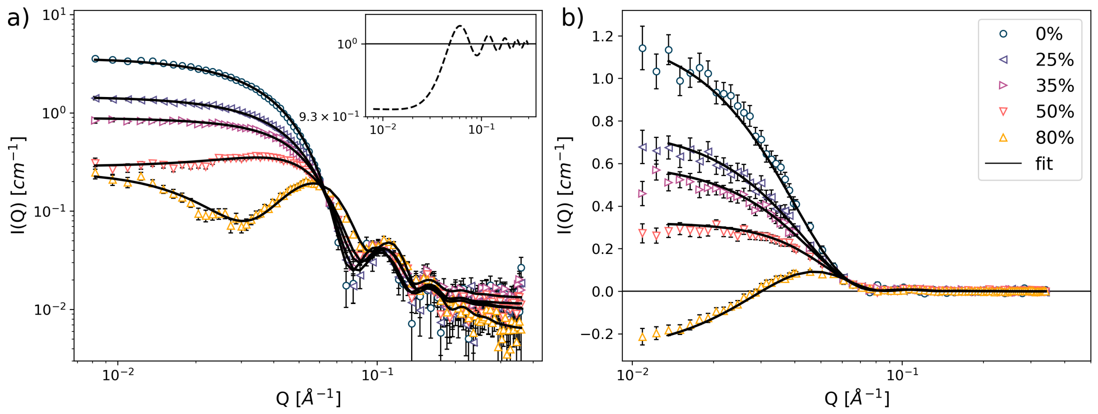


Fig. 4.42 Sector analysis of the SANS data on SE11 samples with varying solvent isotope composition given in volume percentage of deuterated toluene. a) Purely nuclear scattering data obtained from the parallel sectors. A small contribution of a sticky hard sphere structure factor was included for the fits to the nuclear scattering curves (shown in the inset). A contribution from oleic acid micelles in the scattering was necessary to take into account in order to properly fit the data, especially at higher contrasts. b) Fits to the nuclear-magnetic interference terms obtained from the perpendicular sectors are shown. The fit parameters are given in the text.

From the extrapolated intensities at $Q = 0\,\text{Å}^{-1}$ obtained from Guinier fits to the low-Q regions after correction for the sticky hard sphere structure factor contribution the match point can be determined. For this sample the solvent SLD approximately matches

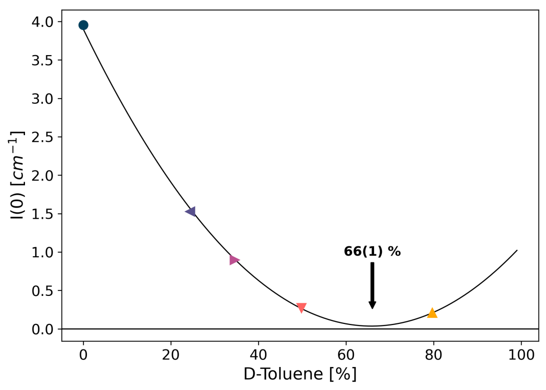


Fig. 4.43 Contrast matching for sample SE11. Extrapolated intensities at $Q = 0$ are obtained from Guinier fits to the SANS measurements with different SLD contrasts after correction for the sticky hard sphere structure factor. The minimum of a parabolic fit corresponds to the solvent contrast where the average particle SLD is almost compensated.

Tab. 4.4 Contrasts in the samples of SE11 prepared for SANS. The last columns gives the calculated SLD contrast for the entire particle using a volume weighted average of the particle core SLD and the organic shell SLD, assuming $R_{core} = 5.9\,\mathrm{nm}$ and $t_{shell} = 1.1\,\mathrm{nm}$.

deuteration (vol. percent)	$\mathrm{SLD}_{solvent}$ (10^{-6} Å^{-2})	$\Delta\mathrm{SLD}_{core}$ (10^{-6}Å^{-2})	$\Delta\mathrm{SLD}_{shell}$ (10^{-6} Å^{-2})	$\Delta\mathrm{SLD}^2_{particle}$ (10^{-6} Å^{-2})^2
0.0	0.939	5.861	0.861	9.382
24.41(1)	2.092	4.708	2.014	3.648
34.60(1)	2.574	4.226	2.496	2.039
49.77(1)	3.291	3.509	3.213	0.506
79.72(1)	4.706	2.094	4.628	0.496

$\mathrm{SLD}_{C_7H_8} = 0.939 \cdot 10^{-6}$ Å^{-2}, $\mathrm{SLD}_{C_7D_8} = 5.664 \cdot 10^{-6}$ Å^{-2}
$\mathrm{SLD}_{shell} = 0.078 \cdot 10^{-6}$ Å^{-2}, $\mathrm{SLD}^a_{core} = 6.8 \cdot 10^{-6}$Å^{-2}

the volume weighted average particle SLD at 66(1) % deuterated toluene (fig. 4.43). Full matching is not possible due to the particle size distribution leading to different ratios of organic shell SLD to inorganic core SLD. To calculate the core SLD from the match point the thickness of the organic shell has to be known, which is determined in the following.

The parallel sector fit to the data of SE11 is shown in fig. 4.42 on the left. To properly describe the data especially at higher deuteration a contribution of oleic acid micelles had to be included via an additional solid sphere form factor. Due to the small size of the micelles a Guinier contribution would also be sufficient. As seen from tab. 4.4 the contrast between oleic acid and solvent is largest for high deuteration, thus a micelle contribution affects the samples with higher deuterated solvents more strongly. With the solvent contrasts and the SLDs for the particle core estimated from the approximate composition obtained from XRD/PDF and Mössbauer spectroscopy and the organic shell given in tab. 4.4 the parameters for the organic shell, the sticky hard sphere structure factor and the micelles could be obtained. The particle core radius and size distribution were fixed to the values obtained from SAXS. This gives an oleic acid surface layer thickness of 1.2(1) nm, which is slightly smaller than the values previously reported[43,125,227]. The micelle radius was found to be 2.4 nm, which is close to the reported values for oleic acid micelles.[228] The sticky hard sphere structure factor parameters are evaluated to 0.021 95 for the volume fraction and 0.386 85 for the stickiness. An addition to the total particle radius was not needed. With these parameters the core SLD via eq. 2.121 evaluates to $6.8(1) \times 10^{-6}\,\text{Å}^{-2}$, which would indicate the presence of some magnetite as was also concluded from Mössbauer data.

It is also possible to determine a non-magnetic surface thickness as described previously. From the fit shown on the right of fig. 4.42 a non-magnetic shell thickness of 0.48(10) nm is obtained, with a magnetic scattering length density in the particle core of $5.8(1) \times 10^{-7}\,\text{Å}^{-2}$. This corresponds to a magnetic particle moment of $14\,970\mu_B$ using the particle size as determined from SAXS reduced by the dead layer thickness. The magnetic moment per iron atom in the magnetic core can be determined using the relation

$$\mu_{Fe} = \frac{\text{SLD}_m V_m}{C n_{Fe}}, \tag{4.26}$$

where $C = \frac{e^2\gamma}{2mc^2} = 2.7 \times 10^{-5}\,\text{Å}$ is the scattering length of $1\mu_B$[229] and V_m is the magnetic volume. The number of Fe-atoms n_{Fe} in the volume V can be estimated with

$$n_{Fe} = \frac{V}{V_{\text{unit cell}}} n_{\text{UC}}. \tag{4.27}$$

With the number of Fe atoms per unit cell $n_{\text{UC}} = n_{\text{FU}}Z = 2.34 \cdot 8$, where Z is the number of formula units per unit cell and n_{FU} is the number of Fe atoms per formula unit obtained from PDF analysis. The mean cubic lattice parameter $a = 8.393\,\text{Å}$ is also taken from the PDF results (section 4.3.4). With these parameters a magnetic moment per iron atom in the magnetic particle core of $0.68\mu_B$ was calculated, which is significantly lower than the estimated bulk maghemite and magnetite room temperature iron atom moment of $1.10\mu_B$ and $1.17\mu_B$, respectively (estimated from the room temperature saturation magnetization values given in tab. 2.3 by $M_{sat.}\rho V_{uc}/(n_{Fe,uc}\mu_B)$, where ρ is the density, V_{uc} is the unit cell volume and $n_{Fe,uc}$ is the number of Fe atoms in the unit cell). The determined iron magnetic moment indicates the presence of significant spin disorder in the particle core. For the entire particle, i.e. including the magnetically dead surface, a net magnetic iron moment of $0.55\mu_B$ is obtained, which would relate to a saturation magnetization of $39(2)\,\text{Am}^2/\text{kg}_{\text{Ferrite}}$ assuming the same density, lattice parameter and iron site occupancy as the bulk material as was implicitly done by the normalization of the SQUID-magnetometry data to the weight of the ferrite.

4.3.5.2 OC15

The contrasts for sample OC15 are given in tab. 4.5. For this sample the solvent SLD approximately matches the volume weighted average particle SLD at 69(2) % deuterated toluene (fig. 4.45). Again full matching is not possible due to the presence of a particle size distribution. With the particle core radius and the lognormal size distribution parameters from SAXS the organic shell thickness was determined to 1.4(1) nm (fig. 4.44 left). This is in agreement with previously reported values[43,125,227] and slightly larger than the value found for sample SE11. With these parameters it is possible to calculate the core SLD via eq. 2.121, giving $6.8(2) \times 10^{-6}\,\text{Å}^{-2}$, which again indicates the presence of some magnetite and is in the range of the expected value using the composition determined from Mössbauer spectroscopy. The parameters of the sticky hard sphere

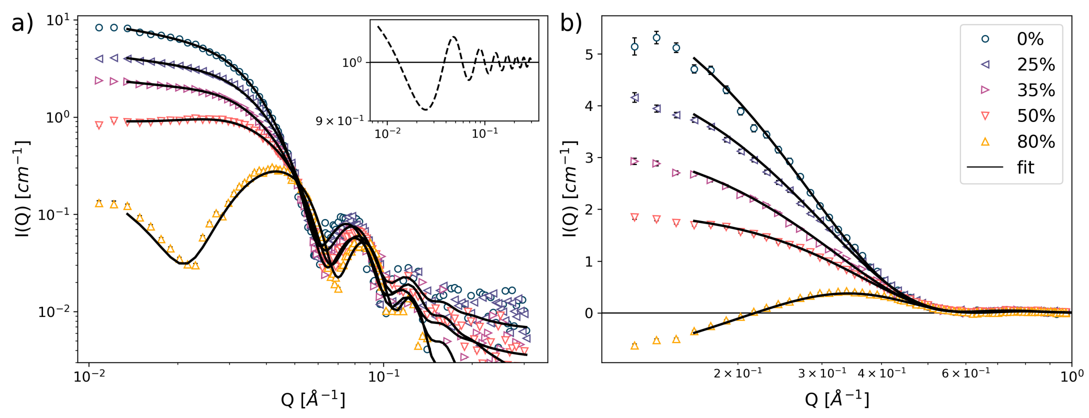


Fig. 4.44 Sector analysis of the SANS data on OC15 samples with varying solvent isotope composition given in volume percentage of deuterated toluene. a) Parallel sector yielding the purely nuclear scattering. The fit gives the organic shell thickness to $1.4(1)\,\mathrm{nm}$. A nuclear core SLD of $6.8 \times 10^{-6}\,\mathrm{\AA}^{-2}$ was used. A small contribution of a sticky hard sphere structure factor (inset) was also included for the parallel sector with a volume fraction of 0.0284, a stickiness of 0.1623 . An addition to the particle radius was not needed. b) Nuclear-magnetic interference term fits obtained from the perpendicular sectors are shown. From this a non-magnetic shell thickness of $0.3(1)\,\mathrm{nm}$ is obtained with a magnetic scattering length density in the particle core of $9.5(2) \times 10^{-7}\,\mathrm{\AA}^{-2}$.

structure factor were also determined with stickiness of $0.344\,56$ and volume fraction of $0.059\,44$. Again an addition to the total particle size was not needed.

From the difference of the integrated data in sectors perpendicular to the applied field of $0.5\,\mathrm{T}$ the nuclear-magnetic interference terms are obtained. As mentioned previously this data can be used to determine the thickness of a non-magnetic surface layer. From the magnetic form factor using a core-shell sphere model the non-magnetic shell thickness t_{surf} was determined to $0.3(1)\,\mathrm{nm}$ with a magnetic SLD of $9.3(2) \times 10^{-7}\,\mathrm{\AA}^{-2}$. This corresponds to a magnetic particle moment of $59\,363\mu_B$ using the particle size as determined from SAXS reduced by the dead layer thickness. With eq. 4.26 the magnetic moment per iron atom in the particle core is determined to $0.96(2)\mu_B$, which is still reduced compared to the estimated bulk value of $1.1\mu_B$, although not as strongly as for sample SE11. Again, the reduced magnetic moment in the particle core is an indication

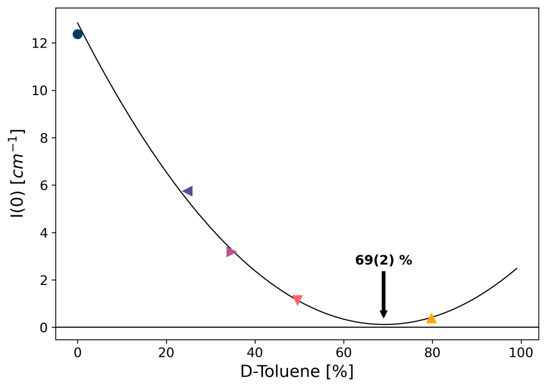


Fig. 4.45 Contrast matching for sample OC15. Extrapolated intensities at $Q = 0$ are obtained from Guinier fits to the SANS measurements with different SLD contrasts after correction for the sticky hard sphere structure factor. The minimum of a parabolic fit corresponds to the solvent contrast where the average particle SLD is almost compensated.

Tab. 4.5 Contrasts in the samples of OC15 prepared for SANS. The last column gives the calculated SLD contrast for the entire particle using a volume weighted average of the particle core SLD and the organic shell SLD, assuming $R_{core} = 7.8\,\text{nm}$ and $t_{shell} = 1.4\,\text{nm}$.

deuteration (vol. percent)	$\text{SLD}_{solvent}$ $(10^{-6}\ \text{Å}^{-2})$	ΔSLD_{core} (10^{-6}Å^{-2})	ΔSLD_{shell} $(10^{-6}\ \text{Å}^{-2})$	$\Delta\text{SLD}^2_{particle}$ $(10^{-6}\ \text{Å}^{-2})^2$
0.0	0.939	5.717	0.861	10.469
24.8(1)	2.111	4.545	2.033	4.258
34.7(1)	2.579	4.077	2.501	2.548
49.6(1)	3.283	3.373	3.205	0.795
79.8(1)	4.710	1.946	4.632	0.287

$\text{SLD}_{C_7H_8} = 0.939 \cdot 10^{-6}\ \text{Å}^{-2}$, $\text{SLD}_{C_7D_8} = 5.664 \cdot 10^{-6}\ \text{Å}^{-2}$
$\text{SLD}_{shell} = 0.078 \cdot 10^{-6}\ \text{Å}^{-2}$, $\text{SLD}^{a}_{core} = 6.8 \cdot 10^{-6}\text{Å}^{-2}$

for spin canting and disorder. For the entire inorganic particle, i.e. the inorganic core size determined from SAXS, an average magnetic iron moment of $0.86(2)\mu_B$ is obtained, which relates to a saturation magnetization of $61(2)\ \text{Am}^2/\text{kg}_{\text{Ferrite}}$.

4.3.5.3 Summary and discussion

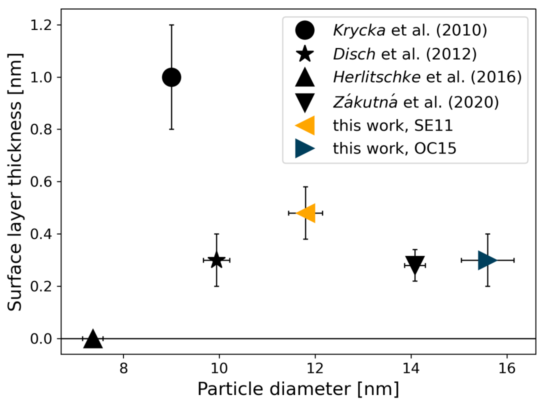


Fig. 4.46 Non-magnetic surface layer thicknesses given in the literature and determined in this work as a function of the particle diameter. Data from Zákutná et al.[44] was recorded with cobalt ferrite nanoparticles. All other cited works refer to iron oxide nanoparticles.

The organic shell thickness and the non-magnetic surface layer thickness could be determined for both samples with the help of contrast variation in SANSPOL. For the smaller particles of sample SE11 a slightly larger dead layer thickness of 0.5(1) nm was found than for sample OC15 with 0.3(1) nm, suggesting an increase of the dead layer with decreasing particle size. Comparison with literature values may provide a more complete picture of the size dependence of this surface layer (fig. 4.46). For monodisperse spherical iron oxide nanoparticles with radius of 4.97 nm and oleic acid coating, i.e. very similar to the particles of this work, Disch et al.[43] found a surface layer thickness of 0.3(1) nm, in agreement with the results obtained for sample OC15. Additionally, in the cited work a similar reduced core iron magnetic moment of $0.67\mu_B$ was found. Considering the results for these three particle sizes no significant size dependence of the surface layer thickness can be seen. Herlitschke et al.[51] studied iron oxide nanoparticles with a radius of 3.7 nm and found no evidence for a non-magnetic surface layer. For cobalt ferrite nanoparticles with mean radius of 7.04 nm a surface layer with spin disorder of 0.28(6) nm was found by Zákutná et al.[44] at an applied field of 1.2 T. While this value is also in agreement with the results of this work due to the different composition these particles may not be comparable to the samples used for this work. Additionally, in the cited work a field dependence of the thickness of the non-magnetic surface layer was found, where it increases with decreasing field. Krycka et al.[37] report surface layer of 1.0(2) nm thickness, where the net magnetization of this surface region is rotated compared to the particle core for iron oxide nanoparticles with radii of 4.5 nm. However, again comparability to the samples presented in this work is limited due to the presence of significant particle interactions as the experiments in the cited work were performed on self assembled particles. Additionally, as mentioned in the introduction of this work, their results are strongly debated in the literature and it is argued that the reported results are not supported by the presented data.[40] Nevertheless, the comparison of all available data suggests a relatively constant surface layer thickness independent of particle size around 0.3 nm for particles in saturation fields.

4.3.6 Magnetometry results

In this section the SQUID-magnetometry data obtained for the different samples is presented and discussed in light of the previously shown results. Starting with the smallest particle sizes the same order as in the previous sections is followed. At the end of this section a direct comparison of all samples is given.

4.3.6.1 OC05

Tab. 4.6 Results of the inductively coupled plasma optical emission spectroscopy (ICP-OES) for the same sample of OC05 that was used for the magnetometry measurements. The total weight is the weight of the sample including the paraffin matrix.

Sample		Mean	standard deviation
Total	mg	33.35	-
Co	µg	<0.01	-
Fe	µg	2.79	0.02
Ni	µg	<0.03	-
Gd	µg	<0.03	-
Cr	µg	<0.03	-
Al	µg	0.984	0.018

Magnetization curves, $M(H)$ and $M(T)$, normalized to the amount of iron present in sample of OC05 as determined by ICP-OES (tab. 4.6), are shown in fig. 4.47. Other possible impurity elements were also considered in the elemental analysis but with the exception of Al their contribution was found to be very small. The Al content possibly originates from impurities in the used paraffin or the scotch tape, from the used ceramic tools or from the sample bottle cap. It is likely present in the oxidized form of Al_2O_3 in the sample, which only leads to a diamagnetic background for larger aggregates or a

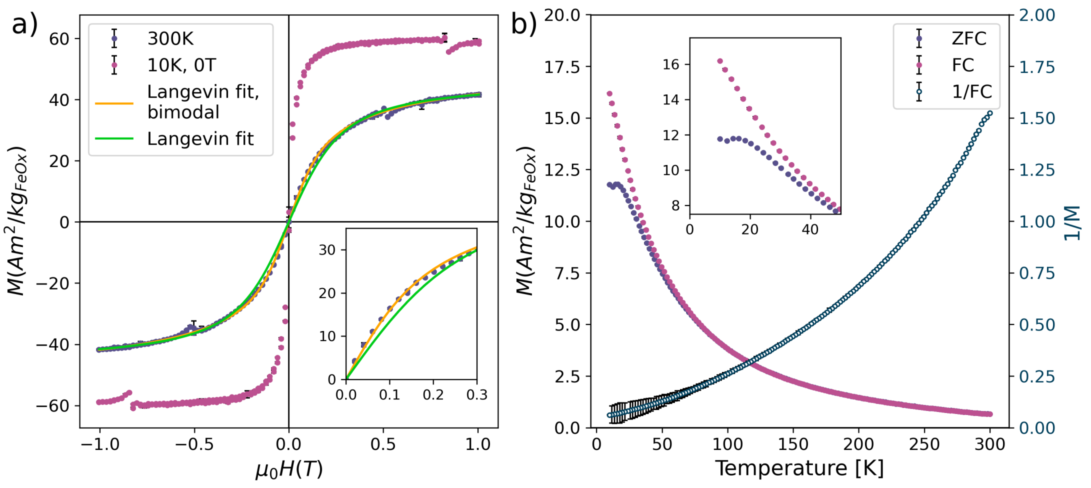


Fig. 4.47 Magnetometry results of sample OC05. a) Magnetization vs. field curves. The cooling to $10\,\mathrm{K}$ was performed without external field. Due to the small particle size even at low temperatures no significant loop opening corresponding to the blocked state is observed. As seen from figure b) the blocking temperature is at about $15\,\mathrm{K}$. The discontinuity in the data for the $10\,\mathrm{K}$ $M(H)$ curve is due to the zero crossing of the data prior to the subtraction of the diamagnetic background. A $1/M_{FC}(T)$ curve is also shown where a non-linear behaviour with increasing temperature can be seen. The inset in fig. b) shows a close-up of the temperature range between $0\,\mathrm{K}$ to $50\,\mathrm{K}$.

four orders of magnitude smaller magnetization than iron oxide nanoparticles[230] and therefore does not influence the magnetometry results. From the $M(T)$ measurements the blocking temperature can be obtained as the temperature where the ZFC curve has its maximum (section 2.1.3.1). For the applied measurement field of $5\,\mathrm{mT}$ the blocking temperature T_B lies at $16\,\mathrm{K}$. Together with the particle volume V, the measurement time τ_m (taken as ca. $30\,\mathrm{s}$) and the elementary spin flip time τ_0, which is typically on the order of $10^{-9}\,\mathrm{s}$ for superparamagnetic systems in ZFC/FC measurements, the magnetic anisotropy energy constant K can be determined according to eq. 2.50. This results in a magnetic anisotropy energy of $92(1)\,\mathrm{kJ/m^3}$ assuming a particle radius of $2.4\,\mathrm{nm}$ and $21(1)\,\mathrm{kJ/m^3}$ assuming a particle radius of $3.9\,\mathrm{nm}$. These two particle sizes are the mean sizes of the bi-modal size distribution determined in the previous section. Thus, using the volume fractions of both particle sizes an effective anisotropy constant of $82(1)\,\mathrm{kJ/m^3}$ is obtained, which is comparable to previously reported values for similar particles[150,231] and significantly larger than the maghemite magnetocrystalline anisotropy of $K = 4.7\,\mathrm{kJ/m^3}$ obtained from measurements on single crystal films and powders[232]. From the inverse of the field cooled magnetization vs. temperature curve a deviation from the ideal linear temperature dependence can be observed. This deviation may be due to a measurement field that is too large for this particle size. In addition there might be an internal phase transition from ferrimagnetic to paramagnetic order with increasing temperature due to the small size of the particles.

At $300\,\mathrm{K}$ the saturation magnetization is determined to $67(1)\,\mathrm{Am^2/kg_{Fe}}$ (using the amount of iron of $2.79(2)\,\mathrm{\mu g}$ in the sample as obtained by ICP-OES (tab. 4.6)). Assuming pure γ-Fe_2O_3, which is reasonable for these particle sizes as shown in sec-

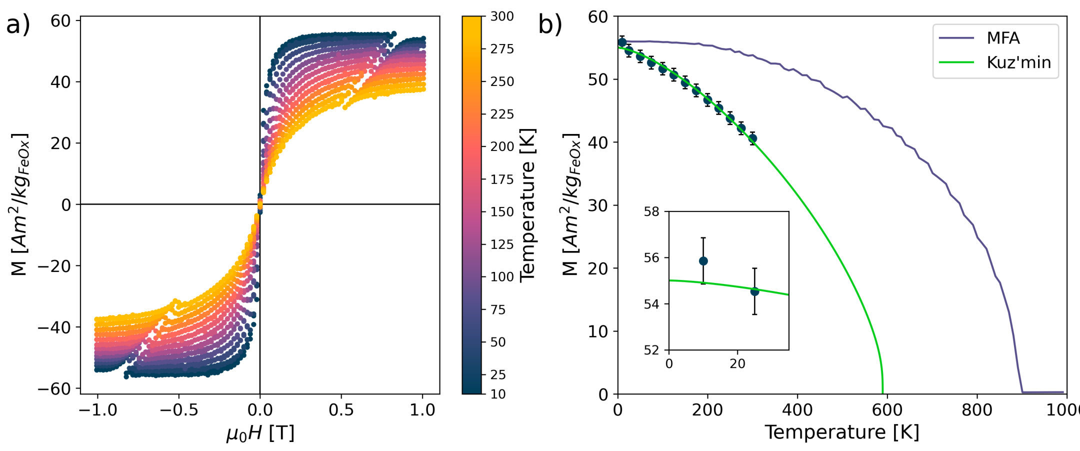


Fig. 4.48 a) Magnetization curves measured at different temperatures as indicated by the colors. The discontinuities correspond to zero crossings prior to the subtraction of the linear diamagnetic background contribution. The data was normalized to the iron oxide weight as concluded from the elemental analysis. b) Saturation magnetization values as determined from the $M(H)$ curves depicted in a) are plotted against the measurement temperature. The lines correspond to theoretical models used to describe the temperature dependence. The approximation from Kuz'min et al.[233] is given in eq. 4.28.

tion 4.3.3, this yields 47(1) $\mathrm{Am^2/kg_{\gamma\text{-}Fe2O3}}$. This value is significantly lower than the room temperature bulk values for γ-Fe_2O_3 (76 $\mathrm{Am^2/kg}$[164]) suggesting a decrease in saturation magnetization of about 38 %. At 10 K the saturation magnetization is larger with 60(1) $\mathrm{Am^2/kg_{\gamma\text{-}Fe2O3}}$, but still reduced by 25 % compared to the bulk. The decrease in saturation magnetization with increasing temperature does not follow the expected path for the bulk material estimated by a mean field approximation (MFA) (fig. 4.48b)). Instead the curvature can be better described by a semi-empirical expression given by Kuz'min et al.[233] according to

$$M(T) = M(0)\left[1 - s\left(\frac{T}{T_C}\right)^{3/2} - (1-s)\left(\frac{T}{T_C}\right)^{5/2}\right]^{1/3}, \tag{4.28}$$

where T_C is the Curie temperature and $0 < s < 5/2$ is a shape parameter to adjust the curve. For sample OC05 the parameter s was determined to 2.4 leading to a Curie temperature of 590 K, which is lower than the expected bulk T_C of approx. 900 K (tab. 2.3), but in accordance with previous observations on similar sized particles[234]. Although the curve appears to have a non-vanishing slope as the temperature approaches 0 K in the closeup of fig. 4.48b it can be seen that the slope decreases with decreasing temperature and vanishes at temperatures around 0 K as expected from the laws of equilibrium thermodynamics.

From a Langevin fit including the lognormal size distribution the superspin moment was determined to 4034(5)μ_B corresponding to a mean magnetic radius of 3.5(1) nm. A superposition of two Langevin curves corresponding to superspins with moments of 1416(5)μ_B and 6456(5)μ_B leads to a better fit in the small field region, where the single Langevin fit deviates from the data. In the presence of two particle sizes or a large size

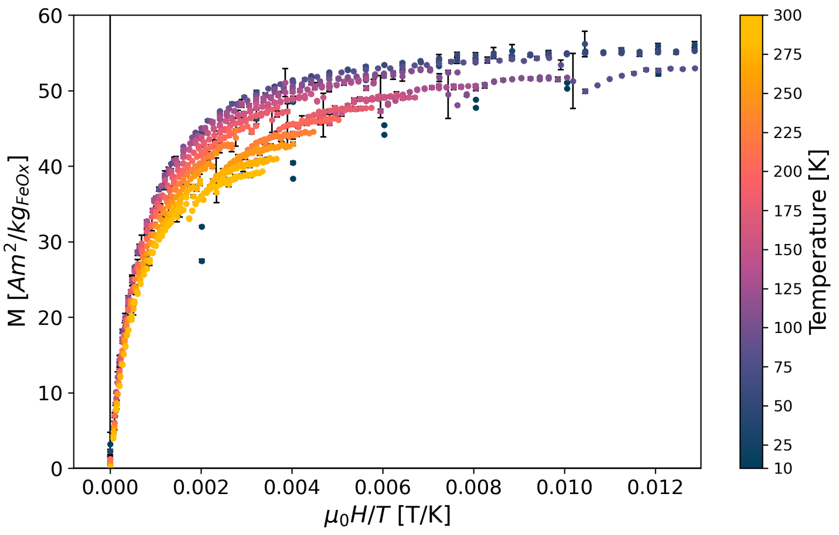


Fig. 4.49 Magnetization vs $\mu_0 H/T$ plot for the OC05 particles (approx. 5 nm in diameter). The curves obtained at different temperatures coincide well indicating superparamagnetic behaviour.[58,235] The deviation of some curves from ideal superposition suggests the presence of two particle sizes as confirmed by SAXS measurements on these particles.

distribution the initial slope of the magnetization curve is more strongly influenced by the particles with larger size, where the magnetic moments can be aligned more easily. In contrast the smaller particles have a stronger influence at the approach to saturation.[235] This observation is consistent with the two particle sizes observed in the SAXS curves and TEM images. The magnetic moments correspond to magnetic volumes of 52 and 256 nm^3, comparable to the particle volumes from SAXS constituting 58(5) and 248(20) nm^3. The presence of two particle sizes is also visible from a superposition of magnetization curves measured at different temperatures. For ideal monodisperse superparamagnetic samples it would be expected that all curves coincide.[58,235] However, clearly two slightly smeared out features are visible (fig. 4.49).

To achieve the measured saturation magnetization at room temperature a magnetically dead surface layer would have to be approx. 0.5 nm thick for the mean particle radius of 3 nm at room temperature, which reduces to 0.3 nm at 10 K. This estimate is, however, neglecting the bi-modal size distribution that was found with SAXS and TEM. With the assumption of an equally thick surface layer for particles of both sizes and under the consideration of the volume fractions a magnetically dead surface layer of 0.2 nm would be sufficient to give the measured saturation magnetization at 10 K and a thickness of 0.4 nm for the room temperature value.

As shown in section 4.3.4 for this sample APBs are not present. However, the total iron occupancy is reduced compared to the bulk material and vacancy ordering is only observed for small regions in the particles. Additionally, local structural disorder was also found. The observed reduced saturation magnetization is thus likely a combination of a magnetically depleted surface layer and the reduced iron site occupancy as well as the structural disorder. It has to be noted, however, that these effects likely do not occur individually but are linked. Meaning that the iron occupancy could be reduced in the particle surface leading to a magnetically depleted and disordered surface layer.

Likewise the structural disorder is probably linked to the presence of an increased amount of vacancies.

4.3.6.2 OC10

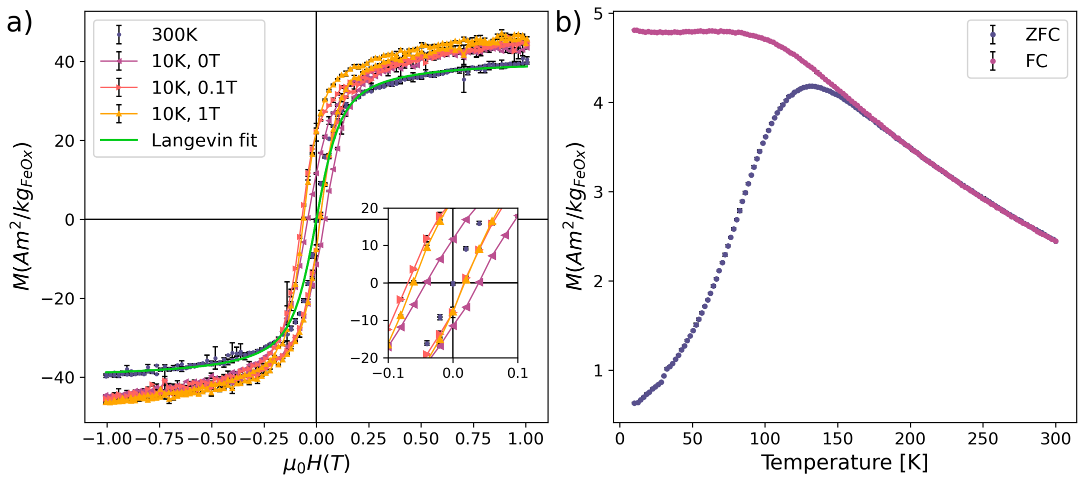


Fig. 4.50 Magnetometry data of sample OC10. a) Magnetization vs. field measurements. The low temperature curves were recorded after cooling in different fields as given in the legend. The inset shows the central region of the plot, where a loop shift of the low temperature magnetization curves after field cooling is clearly visible. b) Magnetization vs. temperature. At measurement fields of $5\,\mathrm{mT}$ the blocking temperature is at $132\,\mathrm{K}$. No sign of a Verwey transition is visible. The disturbance at around $30\,\mathrm{K}$ is due to the zero crossing prior to the diamagnetic background correction.

Tab. 4.7 Results of the inductively coupled plasma optical emission spectroscopy (ICP-OES) for the same sample of OC10 that was used for the magnetometry measurements. The total weight is the weight of the sample including the paraffin matrix.

Sample		Mean	standard deviation
Total	mg	38.51	-
Co	µg	<0.01	-
Fe	µg	1.04	0.02
Ni	µg	<0.06	-
Gd	µg	<0.07	-
Cr	µg	<0.08	-
Al	µg	0.64	0.02

Magnetization curves, $M(H)$ and $M(T)$, normalized to the amount of iron oxide present in the sample as determined from the iron content obtained by ICP-OES (tab. 4.7), are displayed in fig. 4.50. Again impurity contributions were also considered in the elemental analysis. Similar to sample OC05 with the exception of *Al* the relative weight is very small. As mentioned before the *Al* content likely originates from impurities in the used paraffin or the scotch tape, from the used ceramic tools or from the sample bottle cap. At an applied measurement field of 5 mT fig. 4.50b) the blocking temperature T_B lies at 132 K. The magnetic anisotropy energy constant K is determined according to eq. 2.50. With T_B from the SQUID measurement this results in a magnetic anisotropy energy of 54(1) kJ/m^3, which is comparable to previously reported values for similar particles[150] and again significantly larger than the maghemite magnetocrystalline anisotropy. It is also smaller than the value obtained for the OC05 particles consistent with previously reported trends of decreasing anisotropy energy with increasing particle size.[150,231] The close proximity of the splitting of the curves to the maximum of the ZFC curve points to a narrow particle size distribution[86], which is consistent with the results obtained from SAXS experiments. The constant, i.e. not decreasing part of the FC curve for temperatures smaller than the ZFC peak temperature indicates the absence of a significant interparticle interaction[86] and the absence of an additional paramagnetic signal from impurity atoms or clusters dispersed between the particles[87]. No Verwey transition is visible in the $M(T)$ curves. A small exchange bias field of $\mu_0 H_E = 51$ mT in the 0.1 T field cooled $M(H)$ curve and of $\mu_0 H_E = 42$ mT at 1 T was detected (inset in fig. 4.50a)). At 300 K the saturation magnetization was determined to 58(1) Am^2/kg$_{\mathrm{Fe}}$ (using the amount of iron of 1.04(2) µg per sample as obtained by ICP-OES (tab. 4.7). Assuming pure γ-Fe_2O_3, as is supported by Mössbauer spectroscopy, this yields 41(1) Am^2/kg$_{\gamma\text{-Fe2O3}}$. The value for the saturation magnetization is significantly lower than the room temperature bulk values for γ-Fe_2O_3 (76 Am^2/kg[164]) suggesting a decrease in saturation magnetization of about 46 %. At 10 K the saturation magnetization is slightly larger with 46(1) Am^2/kg$_{\gamma\text{-Fe2O3}}$. For this sample the temperature dependence is much less pronounced than observed for the smaller OC05 particles. From a Langevin fit to the room temperature data the superspin moment was determined to 9530μ_B. This would correspond to a magnetic radius of 4.7(1) nm, i.e. a dead surface layer thickness of 1.1(2) nm. However, it has to be emphasized that not all of the magnetically dead or depleted volume has to be on the particle surface. The SANS results for the similar sized particles of SE11 suggest a surface layer thickness of about 0.4 nm. The comparison with literature data also showed that in this size range the surface layer thickness remains rather constant. It is therefore possible to assume that for this sample the surface layer has a similar thickness and is smaller than from the magnetometry data alone. The remaining volume with reduced magnetization is thus distributed within the particle core. As seen in section 4.3.4 for this sample a high degree of APB-superstructure peak broadening was detected, that was attributed to an increased number of APBs in the particle core, which would introduce significant spin disorder in the particle core. Additionally, similar to sample OC05 a reduced iron site occupancy was found that is even more pronounced for this sample.

4.3.6.3 SE11

Magnetization curves, $M(H)$ and $M(T)$, normalized to the amount of iron oxide present in the sample as determined from the iron content obtained by ICP-OES (tab. 4.8), are displayed in fig. 4.51. The relative amounts of the different considered elements are comparable to the previous samples, again with an elevated contribution of Al. At an applied measurement field of 5 mT (fig. 4.51b) the blocking temperature T_B lies at 106 K, which is much lower than the value observed for OC10 despite the similar particle size. This points to a difference in the magnetic anisotropy energy constant determined

Tab. 4.8 Results of the inductively coupled plasma optical emission spectroscopy (ICP-OES) for the same sample of SE11 that was used for the magnetometry measurements. The total weight is the weight of the sample including the paraffin matrix.

Sample		Mean	standard deviation
Total	mg	36.33	-
Co	µg	<0.08	-
Fe	µg	1.8	0.2
Ni	µg	<0.03	-
Gd	µg	<0.1	-
Cr	µg	<0.04	-
Al	µg	2.6	1.4

according to eq. 2.50. With T_B from the SQUID measurement this results in a magnetic anisotropy energy of $41(1)\,\mathrm{kJ/m^3}$, which is smaller than the value obtained for the OC10 particles. The very narrow size distribution found from the SAXS data is again reflected in the close proximity of the splitting of the curves to the maximum of the ZFC curve.[86] As noted previously also for sample OC10 the constant, i.e. not decreasing part of the FC curve for temperatures smaller than the ZFC peak temperature indicates the absence of a significant interparticle interaction[86] and the absence of an additional paramagnetic signal from impurity atoms or clusters dispersed between the particles[87]. The Verwey transition indicating the presence of magnetite would be expected at 120 K, however it would be near or above the blocking temperature. It has been suggested that the transition temperature may be shifted towards smaller temperatures for non-stoichiometric magnetite.[169] In the 0.05 T zero field cooled curve a feature is visible at around 100 K, that is absent in the purely maghemite particles of sample OC10 (fig. 4.50b). Additionally, a small hump in the ZFC curve is detected at around 50 K visible at both fields. This has been attributed to a spin glass transition or spin reorientation.[236] However, it could also be attributed to the freezing of oxygen in the sample environment at around 50 K. Against this speaks the lack of this feature in the curves obtained for sample OC10 shown in fig. 4.50b) of the preceding section. Since for both samples data was recorded with the same instrument under the same conditions a sample environment effect should be visible in both. A small exchange bias field of $\mu_0 H_E = 35\,\mathrm{mT}$ in the 0.1 T field cooled $M(H)$ curve and of $\mu_0 H_E = 30\,\mathrm{mT}$ at 1 T was detected (inset in fig. 4.51a). As mentioned in section 2.1.2.7 this may be due to the presence of antiferromagnetic wüstite or of APBs. However, the wüstite contribution was excluded based on both X-ray diffraction and Mössbauer spectroscopy. These exchange bias fields are also slightly smaller than the ones observed for OC10.

At 300 K the saturation magnetization was determined to $62(6)\,\mathrm{Am^2/kg_{Fe}}$ (using the amount of iron of 1.8(2) µg in the sample as obtained by ICP-OES (tab. 4.7). With a particle composition of 70 % γ-Fe_2O_3 and 30 % Fe_3O_4 as estimated from Mössbauer spectroscopy of the slightly larger particles of SEs12 this yields
$44(5)\,\mathrm{Am^2/kg_{ferrite}}$. Within the errors this is in agreement with the result obtained from SANS. It has to be noted that the difference in saturation magnetization assuming different particle compositions is smaller than the uncertainty introduced by the measurement of the elemental analysis. The large standard deviation is probably due to a

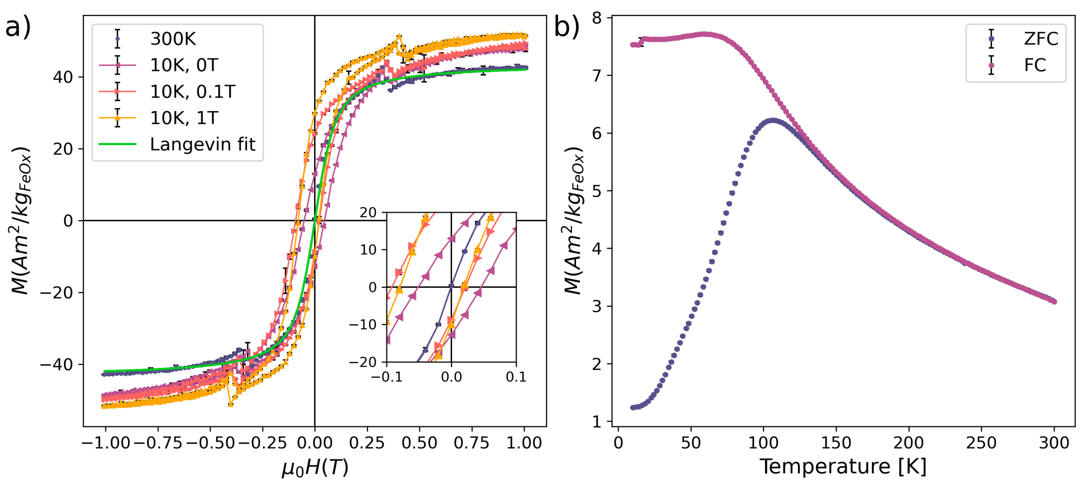


Fig. 4.51 Magnetometry data of sample SE11. a) Magnetization vs. field measurements. The low temperature curves were recorded after cooling in different fields as given in the legend. The inset shows the central region of the plot, where a loop shift of the low temperature magnetization curves after field cooling is clearly visible. b) Magnetization vs. temperature. At measurement fields of $5\,\mathrm{mT}$ the blocking temperature is at $106\,\mathrm{K}$. A small additional peak lies at approx. $50\,\mathrm{K}$.

stronger dilution of the as-prepared sample compared to the Ocean NanoTech particles leading to an overall weaker signal.

The value for the saturation magnetization is significantly lower than the room temperature bulk values for magnetite and maghemite, suggesting a decrease in saturation magnetization of about $42(6)\,\%$ compared to maghemite and $49(6)\,\%$ compared to magnetite. Assuming the mixture of ferrites gives an average reduction of $44(6)\,\%$. At $10\,\mathrm{K}$ the saturation magnetization is slightly larger with $52(5)\,\mathrm{Am}^2/\mathrm{kg}_{\mathrm{ferrite}}$. From a Langevin fit to the room temperature $M(H)$ curve the magnetic particle moment is determined to $11\,774\mu_B$, corresponding to a magnetic radius of $4.9\,\mathrm{nm}$, i.e. a magnetically dead surface layer thickness of about $1.0\,\mathrm{nm}$. The same considerations as for sample OC10 apply, namely that the magnetically depleted volume is not necessarily placed at the particle surface but can also correspond to spin disordered regions within the particle. For this sample also SANS data is available, that was presented in section 4.3.5. A magnetically dead surface layer of approx. $0.4\,\mathrm{nm}$ was found with SANS, which is less than half the value obtained from SQUID. As mentioned before the reason for this discrepancy may be the presence of APBs and disorder around defects. This is supported by the small iron magnetic moment in the particle core that was deduced from the SANS data. From the magnetic scattering length density of the particle core a moment of $14\,660\mu_B$ was calculated, which is larger than the obtained value from SQUID magnetometry. However, the SANS data was recorded at an applied field of $1.3\,\mathrm{T}$. A field dependence of the magnetic moment via a field dependence of the magnetic volume could account for this difference.[44] The reduced moment even at high fields indicates the prevalence of spin disorder at the surface and in the particle core.

Tab. 4.9 Results of the inductively coupled plasma optical emission spectroscopy (ICP-OES) for the same sample of OC15 that was used for the magnetometry measurements. The total weight is the weight of the sample including the paraffin matrix.

Sample		Mean	standard deviation
Total	mg	33.35	-
Co	µg	<0.01	-
Fe	µg	1.363	0.017
Ni	µg	<0.03	-
Gd	µg	<0.06	-
Cr	µg	<0.001	-
Al	µg	0.7	0.03

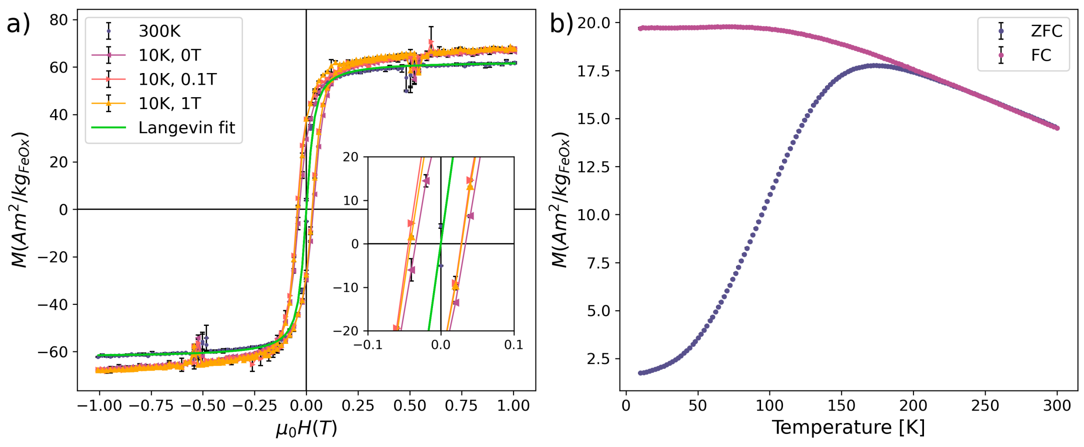


Fig. 4.52 Magnetometry data of sample OC15. a) Magnetization vs. field measurements. The low temperature curves were recorded after cooling in different fields as given in the legend. The inset shows the central region of the plot, where a loop shift of the low temperature magnetization curves after field cooling is clearly visible. b) Magnetization vs. temperature. At measurement fields of $5\,\mathrm{mT}$ the blocking temperature lies at $172\,\mathrm{K}$. No sign of a Verwey transition is visible.

4.3.6.4 OC15

Magnetization curves, $M(H)$ and $M(T)$, normalized to the amount of iron oxide present in sample OC15 as determined from the iron content obtained by ICP-OES (tab. 4.8), are displayed in fig. 4.52. The relative amounts of the different considered elements are again comparable to the previous samples. The elevated contribution of *Al* was also found for this sample. At an applied measurement field of $5\,\mathrm{mT}$ (fig. 4.52b) the blocking temperature T_B lies at $172\,\mathrm{K}$. The blocking temperature from the SQUID measurement corresponds to the magnetic anisotropy energy of $28.9(1)\,\mathrm{kJ/m^3}$, which is comparable to previously reported values for similar particles[150] and again significantly larger than the maghemite magnetocrystalline anisotropy. The very narrow size distribution found from the SAXS data is reflected again in a small difference between the ZFC peak and the FC/ZFC splitting temperature. As noted previously also for samples OC10 and SE11 the constant, i.e. not decreasing part of the FC curve for temperatures smaller

than the ZFC peak temperature indicates the absence of a significant interparticle interaction[86] and the absence of an additional paramagnetic signal from impurity atoms or clusters dispersed between the particles[87]. No Verwey transition can be observed in the ZFC curve and no other features are visible. From Mössbauer spectroscopy a magnetite content of about 15 % was estimated, however for this particle size the Verwey transition might be suppressed.[64] A small exchange bias field of $\mu_0 H_E = 8\,\mathrm{mT}$ in the 0.1 T field cooled $M(H)$ curve and of $\mu_0 H_E = 6\,\mathrm{mT}$ at 1 T was detected (inset in fig. 4.52). These values are much smaller than observed for the previous samples. Again, a wüstite contribution as an explanation for the presence of the exchange bias effect was ruled out on the basis of Mössbauer spectroscopy and X-ray diffraction.
At 300 K the saturation magnetization was determined to 88(2) $\mathrm{Am^2/kg_{Fe}}$ (using the amount of iron of 1.363(17) µg in the sample as obtained by ICP-OES (tab. 4.9). Assuming pure γ-Fe_2O_3 this yields 62(1) $\mathrm{Am^2/kg_{\gamma\text{-}Fe2O3}}$. For pure Fe_3O_4 a value of 64(1) $\mathrm{Am^2/kg_{Fe3O4}}$ is obtained. Using the composition obtained from Mössbauer spectroscopy gives 62(1) $\mathrm{Am^2/kg_{Ferrite}}$, in good agreement with the value determined by SANS (section 4.3.5). At 10 K the saturation magnetization is slightly larger with 68(1) $\mathrm{Am^2/kg_{Ferrite}}$.
As observed for the previous samples, also for this particle size the value for the saturation magnetization is significantly lower than the room temperature bulk values suggesting a decrease in saturation magnetization of about 20 %. The Langevin fit to the room temperature data results in a magnetic particle moment of $28\,458\mu_B$ corresponding to a particle magnetic radius of 5.9 nm, i.e. a magnetically dead surface layer thickness of 1.9 nm. However, SANS data (section 4.3.5) show a magnetically dead surface layer of only 0.3 nm thickness. Furthermore, the magnetic moment of the particle determined by SANS constituted $59\,430\mu_B$. Such a large particle moment would not be able to properly describe the curvature of the $M(H)$ curve for low fields. Thus, similar to sample SE11 a field dependence of the magnetic moment is proposed.
It is interesting that the reduction in saturation magnetization for this sample, while still significant, is much less severe than for the other samples. This is likely linked to the structural properties of sample OC15 determined in section 4.3.4. The degree of APB induced peak broadening is less pronounced for this sample compared to samples OC10, SE11 and OC20, suggesting the presence of fewer APBs. This would explain the larger saturation magnetization compared to these samples. Additionally, the highest iron site occupancy was found for this sample, leading to a higher saturation magnetization as well. Finally, the influence of the non-magnetic surface layer is smaller than for the particles of sample OC05 due to the smaller relative volume ratio in OC15, assuming a similar thickness of the magnetically dead surface layer for both samples.

4.3.6.5 OC20

Despite the large magnetite content concluded from Mössbauer spectroscopy no Verwey transition is visible in the ZFC curve (fig. 4.53b) of sample OC20. However, again a small feature is visible at about 50 K similar to sample SE11 where a larger magnetite content was also expected. At an applied measurement field of 5 mT (fig. 4.52b) the blocking temperature T_B lies above 300 K, i.e. outside the measurement range. An extrapolation suggests a blocking temperature of about 310 K. With this the magnetic anisotropy energy results in 20.1(1) $\mathrm{kJ/m^3}$. This value is still significantly larger than the maghemite manetocrystalline anisotropy, but smaller than the values observed for

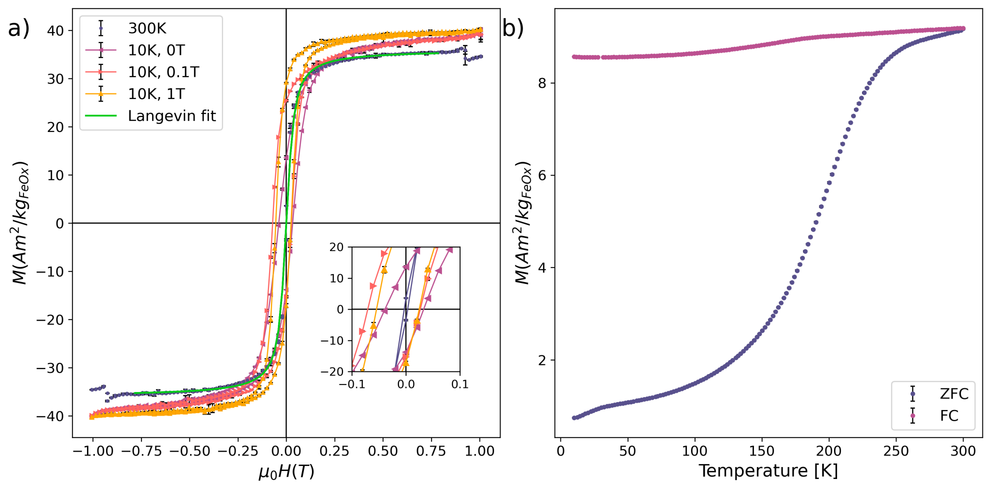


Fig. 4.53 Magnetometry data of sample OC20. a) Magnetization vs. field measurements. The low temperature curves were recorded after cooling in different fields as given in the legend. The inset shows the central region of the plot, where a loop shift of the low temperature magnetization curves after field cooling is clearly visible. b) Magnetization vs. temperature is shown. At measurement fields of $5\,\mathrm{mT}$ the blocking temperature is above $300\,\mathrm{K}$, i.e. outside the measurement range.

the smaller particle sizes. At 300 K the saturation magnetization was determined to $51(1)\,\mathrm{Am^2/kg_{Fe}}$, using the amount of iron of 4.68(5) µg in the sample as obtained by ICP-OES (tab. 4.7). As for the previous samples no significant impurities were found from the elemental analysis, with the exception of Al. Assuming pure γ-Fe_2O_3 or Fe_3O_4 this yields $35(1)\,\mathrm{Am^2/kg_{\gamma\text{-}Fe2O3}}$ or $37(1)\,\mathrm{Am^2/kg_{Fe3O4}}$, respectively. Assuming a composition of 70 % Fe_3O_4 and 30 % γ-Fe_2O_3 as indicated by Mössbauer spectroscopy a value of $36(1)\,\mathrm{Am^2/kg_{Ferrite}}$ is obtained. This is significantly lower than the room temperature bulk values for γ-Fe_2O_3 ($76\,\mathrm{Am^2/kg}$[164]) and Fe_3O_4 ($86\,\mathrm{Am^2/kg}$[163]) suggesting a decrease in saturation magnetization of about 57 %. A slightly larger saturation magnetization is recorded at 10 K with $41(1)\,\mathrm{Am^2/kg_{Ferrite}}$. The superspin moment was determined to $27\,960(5)\mu_B$. This would correspond to a magnetic radius of 7.0(1) nm, i.e. a dead surface layer thickness of 3.7(2) nm. As seen from the SANSPOL data on

Tab. 4.10 Results of the inductively coupled plasma optical emission spectroscopy (ICP-OES) for the same sample of OC20 that was used for the magnetometry measurements. The total weight is the weight of the sample including the paraffin matrix.

Sample		Mean	standard deviation
Total	mg	36.33	-
Co	µg	<0.01	-
Fe	µg	4.68	0.05
Ni	µg	<0.06	-
Gd	µg	<0.07	-
Cr	µg	<0.08	-
Al	µg	0.57	0.05

smaller particles this is unreasonably large. More likely is a smaller non-magnetic surface layer in combination with significant spin disorder due to the large amount of APBs concluded from the X-ray diffraction peak broadening.

Exchange bias fields of $\mu_0 H_E = 22\,\mathrm{mT}$ in the $0.1\,\mathrm{T}$ field cooled $M(H)$ curve and of $\mu_0 H_E = 15\,\mathrm{mT}$ at $1\,\mathrm{T}$ were detected (inset in fig. 4.52). Additionally, a small shift of $3\,\mathrm{mT}$ can be observed for the curve after cooling in zero field. The approach to saturation for the $10\,\mathrm{K}$ curves also differs significantly. While the curves recorded after cooling fields of $0\,\mathrm{T}$ and $0.1\,\mathrm{T}$ almost coincide the $1\,\mathrm{T}$ curve shows a more rectangular hysteresis loop. A similar feature was observed for sample SE11 that also exhibited an increased amount of magnetite. Sample OC15 that also contains magnetite does not show this feature. In the purely maghemite containing particles of OC10 a slightly less pronounced yet similar effect can also be observed.

4.3.6.6 Comparison of the samples

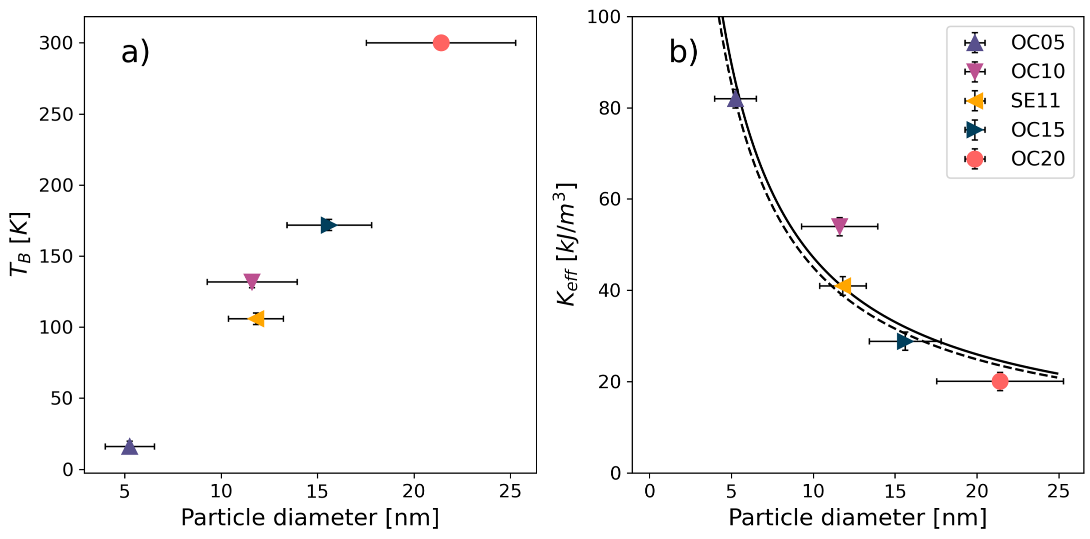


Fig. 4.54 a) Blocking temperature determined from ZFC $M(T)$-curves as a function of the particle diameter. b) Effective anisotropy constants for the samples as a function of the particle diameter. The lines correspond to fits according to eq. 4.29. The dashed line represents a fit excluding sample OC10. The error bars on the particle sizes correspond to the standard deviations calculated from the lognormal size distributions obtained from SAXS.

A size dependence of the effective magnetic anisotropy calculated from the blocking temperatures given in fig. 4.54a) is visible in the data shown in fig. 4.54b). The effective anisotropy constant can be phenomenologically expressed as a sum of a volume contribution K_V and a surface contribution K_S according to

$$K_{\text{eff}} = K_V + \frac{6}{d} K_S, \tag{4.29}$$

where d is the particle diameter.[64,237] A fit to the experimental data using the bulk maghemite magnetocrystalline anisotropy constant of $K_V = 4.7\,\mathrm{kJ/m^3}$ and the particle diameters as determined from SAXS results in a surface anisotropy of $K_S = 7.1(2) \times 10^{-8}\,\mathrm{kJ/m^2}$. This value is similar but slightly larger compared to previously published data ($2.9 \times 10^{-8}\,\mathrm{kJ/m^2}$,[238] $6.9 \times 10^{-8}\,\mathrm{kJ/m^2}$,[239] $5.8 \times 10^{-8}\,\mathrm{kJ/m^2}$[240]). Sample OC10 exhibits a larger effective anisotropy constant than would be expected for this particle size. This could either be related to a larger influence of the surface for this sample, or more likely it is related to the fact that the particles of sample OC10 contain more vacancies than the particles of similar size of sample SE11. Thus, spin canting around these defects might lead to the larger observed anisotropy.[241] Additionally, the observed aggregation for this sample might influence the magnetic anisotropy. Excluding this sample from the fit results in a surface anisotropy constant of $K_S = 6.7(1) \times 10^{-8}\,\mathrm{kJ/m^2}$, which is closer to the previously published data.

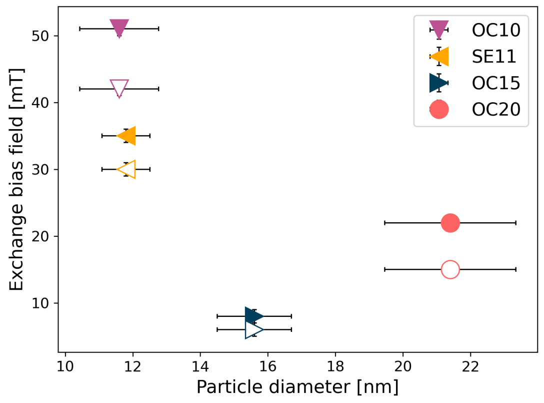


Fig. 4.55 Exchange bias (EB) fields determined from the $10\,\mathrm{K}$ $M(H)$ curve coercivities for the different particle sizes. For sample OC05 no exchange bias fields could be determined because at $10\,\mathrm{K}$ the particles were not yet in the blocked state. Filled and empty symbols relate to EB-fields determined for cooling fields of $0.1\,\mathrm{T}$ and $1.0\,\mathrm{T}$, respectively.

Exchange bias fields shown in fig. 4.55. Exchange bias in iron oxide compounds is commonly attributed to the presence of an interface of antiferromagnetic wüstite with ferrimagnetic magnetite/maghemite.[69,72] However, no indications for the presence of wüstite were found in the Mössbauer spectra as well as the SQUID-data. Furthermore, the development of the exchange bias fields with the particle size would suggest an increase in wüstite content with decreasing particle size, which is not plausible considering the oxidation sequence of the iron oxides (see section 2.6). An exchange bias effect originating from the interface of a magnetically disordered spin-glass like surface layer and the particle core has also been proposed.[74,75] However, Martínez-Boubeta et al.[74] observed an increasing exchange bias field with increasing particle size and increasing magnetite content, which is not observed here. Their largest particle size of 13 nm showed a saturation magnetization close to the bulk magnetization of magnetite, thus suggesting these particles lack the structural defects of the particles investigated in this thesis and the results may not be comparable. Increasing exchange bias fields have also been linked to increasing vacancy disorder[33] as well as to the presence of APBs[46]. In the work of Levy et al.[45] exchange bias observed for 18 nm particles was ascribed to lattice strain and antiphase boundaries. However, comparison of the size dependence of the APB peak broadening and the exchange bias fields shows no direct correspondence. Especially for samples OC10 and SE11 that showed almost identical values for the number of APBs (fig. 4.41c) different exchange bias fields are detected. Considering different degrees of vacancy ordering as the source of differing exchange bias fields as proposed by Li et al.[33] would explain the smaller exchange bias field for samples OC15 and OC20, as for these sample the vacancy ordering was found to persist over larger distances than for the other samples (section 4.3.4). For OC10 the onset of small superstructure peaks was found, whereas for SE11 this was not the case, which would suggest that for OC10 the exchange bias field should be smaller. However, the opposite is the case which could be related to the presence of aggregates in the sample of OC10 as found by SAXS (section 4.3.2) that were absent for sample SE11, leading to a larger exchange bias due to the interaction of particles in OC10. The observed difference in the exchange bias fields for the samples may thus be the result of the interplay of a non-magnetic surface layer, the presence of APBs, particle interactions and the degree of vacancy ordering. A study of the exchange bias field as a function of these individual parameters is however beyond the scope of this work.

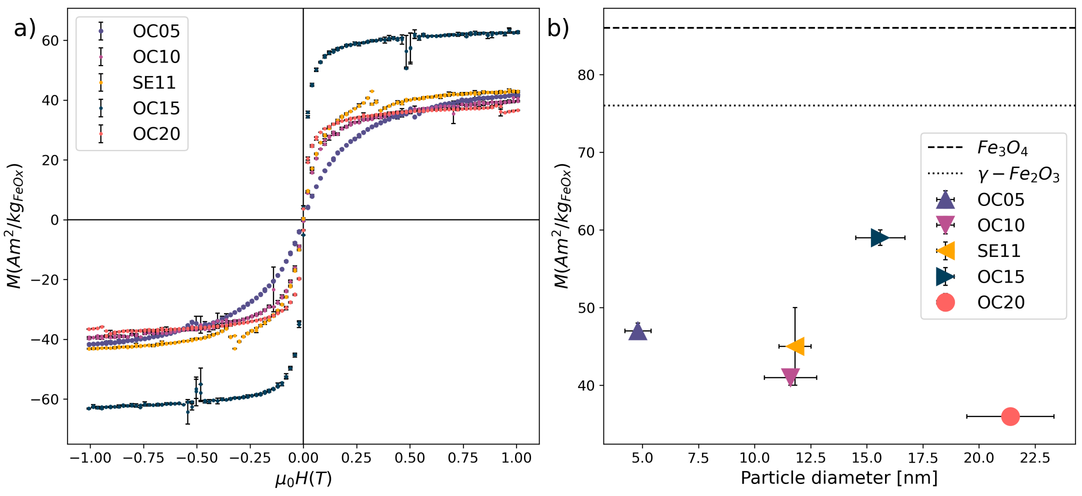


Fig. 4.56 a) Room temperature $M(H)$ curves for the different particle sizes. They are normalized to the iron oxide weight obtained from the iron content in the sample as determined by ICP-OES. b) Saturation magnetization obtained from Langevin fits to the magnetization curves is shown in dependence of the particle size. The dashed and the dotted lines correspond to the theoretical room temperature values of magnetite (Fe_3O_4) and maghemite (γ-Fe_2O_3), respectively.

A comparison of the room temperature $M(H)$ curves and the determined saturation magnetizations is shown in fig 4.56. It is remarkable that for the particles considered in this thesis a trend of decreasing saturation magnetization with increasing particle size is observed, which is in contrast to previous systematic studies of the magnetic properties of iron oxide nanoparticles,[33,48,241] but in agreement with results from Levy et al.[45] An outlier is the sample OC15. As mentioned in section 4.3.1 these particles have already aged in the bottle for several years when the experiments were performed. Baaziz et al. noted already after 1 month an increased oxidation of iron oxide nanoparticle samples.[241] This increased oxidation and the subsequent lattice rearrangements and relaxations lead to less structural disorder and a larger degree of vacancy ordering as observed with X-ray diffraction and PDF analysis for sample OC15. Nevertheless, for all particles a reduced magnetization compared to the bulk materials is observed. The larger error on the saturation magnetization of SE11 is due to a larger error in the Fe-content determination with ICP-OES. However, the saturation magnetization determined from the magnetic scattering length density suggests a value close to the one obtained for OC10. As seen in section 4.3.3, the magnetite content increases with increasing particle size. Thus, compositional variations cannot explain the observed trend. Another possibility is an increasing thickness of a magnetically dead layer on the particle surface with increasing particle size. As shown in section 4.3.5 this is not in agreement with experimental results obtained from SANS measurements. Most likely is therefore a complex interplay of different effects that superimpose to varying degrees depending on the particle size. One contribution is spin disorder in the particle core. As mentioned before, APBs introduce significant disorder in the spin structure. From the fits to the X-ray powder diffraction data the amounts of APBs were determined, with the lowest value for sample OC15 (apart from OC05 where no APB contribution was found) and the largest for OC20. Additionally, samples OC10 and SE11 were found to exhibit very similar broadening of peaks relating to a similar amount of APBs, which is reflected in the similarity of the determined saturation magnetizations. For sample OC05 no significant APB broadening was observed, thus the reduction of the magnetization for this sample must be of different origin. For this sample likely surface effects are important due to the small particle size. Moreover, the increased amount of iron vacancies on octahedral lattice sites further reduces the magnetization. The iron site occupancy is also reduced for samples OC10, SE11 and OC20 adding to the effect of APBs on the magnetization. For sample OC15 a larger iron site occupancy was found. As mentioned previously vacancies are ordered to varying degrees for the samples of this work, with the most ordered structures found in samples OC15 and OC20.

4.3.7 Conclusive remarks

Some conclusions may be drawn based on the results and considerations. The non-magnetic surface layer likely remains of similar thickness for all particle sizes, thereby explaining the often observed increase in magnetization for otherwise defect-free particles as the surface-to-volume ratio decreases. Further SANSPOL experiments are needed to conclusively show this also for smaller particles, but SQUID-measurements on the smallest particles do not suggest a significantly larger surface layer. Additionally, APBs have a strong effect on the magnetization being mainly responsible for the size dependence of the magnetization observed for the particles used in this work. APBs seem to be absent for particles smaller than about 10 nm in diameter. Structural disorder and the amount of vacancies contribute to the magnetic behaviour as well. However, as previously noted these are likely linked to the presence of a disordered surface layer as well as some structural distortions at the APBs and thus an individual assessment of their influence on the magnetization is difficult. Vacancy ordering appears to be related to the observed exchange bias fields, where a higher degree of ordering corresponds to smaller detected exchange bias effects. Finally, the magnetite content seems to have only a small influence on the saturation magnetization for the particles considered here but in general it increases with increasing particle size. For otherwise defect-free particles the increasing magnetite content would likely lead to a higher saturation magnetization for increasing particle sizes. The synthesis route most likely plays an important role in the exhibited particle properties. The particles used in the present work were synthesized via the thermal decomposition route with an iron oleate precursor[52] that appears to be prone to the formation of APBs. Kemp et al.[163] report almost bulk-like magnetization values for particles produced by a slightly modified route based on the approach published by Park et al.[52]. The main difference in their approach is that they used and Ar-atmosphere during the synthesis and controlled the particle oxidation very precisely by adding $1\,\%$ of O_2. However, they also observe reduced magnetizations for larger particles ($59\,\mathrm{Am^2/kg}$ for particles with diameters of $28.6\,\mathrm{nm}$). Large magnetization values in the range of $70\,\mathrm{Am^2/kg}$ to $82\,\mathrm{Am^2/kg}$ were observed by Sun et al.[3], who used $Fe(acac)_3$ as iron precursor instead of iron oleate. This observation was confirmed by Nedelkoski et al.[47] suggesting that this precursor should be preferred if the reduction of APB formation and the increase of saturation magnetization is desired.

Chapter 5

Summary and conclusion

In this work the complex interplay of structural, compositional, and magnetic properties of superparamagnetic iron oxide nanoparticles was studied by the combined use of computer simulations and experiments on nanoparticle samples with various sizes. In the approach followed here a wealth of complementary techniques was employed to obtain an as complete as possible picture of the different contributing effects. Antiphase boundaries (APBs) as a main factor were especially considered.

Monte Carlo simulations provide insights into the effect of APBs on the local spin structure and the macroscopic magnetization of iron oxide nanoparticles. In saturation fields the induced spin disorder is spatially confined to the region close to the APB plane and the particle as a whole remains in the single domain state as would be expected for the considered particle sizes. The effect on the net saturation magnetization of the particles at an applied field of 1.5 T is found to be that of a reduction by about 8 % for the smallest particles (5.0 nm). It is less severe for the largest particles (9.2 nm) with about 5 %. This drop in magnetization is dependent on the applied field strength, where the difference between the magnetization of particles with APB and those without is less pronounced for larger fields. In addition to the studied effect of APBs it is shown that vacancy ordering in an otherwise defect free particle leads to a slight increase in the saturation magnetization. In this work for the first time simulations of iron oxide nanoparticle powder diffraction patterns with APBs are performed that show a distinct peak broadening effect in X-ray powder diffraction patterns that differs from peak to peak. This unique property and its connection to APBs known already from different compounds has found until now no attention in the literature on iron oxide nanoparticles. Fourier coefficients, that describe the observed peak broadening, to be used in a whole powder pattern modeling approach are validated by the simulations. In addition, an expression is developed on the basis of the simulations that allows the quantification of APBs from the fitting parameter. Furthermore also the effect of APBs on the pair distribution function (PDF) is studied revealing a mismatch between PDFs calculated from particles with and without these defects, that gets more severe with increasing interatomic distance. This has to be considered in the fitting of real data and may provide an additional explanation for the difficulties in obtaining good fits to iron oxide nanoparticle samples. As a third part of this thesis a comprehensive investigation of iron oxide nanoparticle samples with particle sizes of 4.8, 11.6, 11.8, 15.6 and 21.4 nm is reported. The particle samples are first characterized by small-angle X-ray scattering and transmission electron microscopy (TEM), which shows that with the exception of the smallest particle sample the particle size distribution is reasonably monodisperse. For the 4.8 nm particles a bi-modal size distribution is found. Further characterization was performed via Mössbauer spectroscopy, which shows a size dependence of the magnetite content, where this contribution is larger for increasing particle sizes. For the 11.6 nm sample no magnetite is found and a composition entirely of maghemite is concluded. Structural studies of all particle sizes were performed with X-ray powder diffraction and PDF analysis. The previously mentioned APB line broadening effect is used to estimate the amount of APBs in the particle structures. From the PDF analysis information on the amount and ordering of vacancies is obtained. For the 15.6 nm particles high-resolution TEM offers a direct way the confirm the presence of APBs. In this part it is shown that the peak broadening related to APBs generally becomes stronger with increasing particle size and is virtually absent for the smallest particle sample. A detailed study of the magnetization distribution within the particle is performed for the 11.8 and 15.6 nm particles with the help of small-angle neutron scattering with polarized neutrons. A non-magnetic surface layer of 0.4(1) nm and 0.3(1) nm exists respectively. Within errors and by comparison with literature values it seems that the surface layer thickness is rather independent of the particle size, at least for particles larger than 10 nm. A peculiar trend of decreasing saturation magnetization with increasing particle size is evidenced from SQUID-magnetometry. An exception are the 15.6 nm particles, where the largest magnetization is present. Based on the previous findings it could be concluded that the decrease in saturation magnetization for the other samples is connected to an increase in the number of APBs as concluded from the X-ray powder diffraction measurements. The non-magnetic surface layer also contributes to the magnetic properties explaining the low magnetization of the smallest particles, where no APB-related broadening of the diffraction peaks occurs. Other effects such as an increased saturation magnetization due to an increased contribution of magnetite to the particle composition are overshadowed by the presence of the APBs. The large saturation magnetization observed for the 15.6 nm particles is in agreement with the observed smaller peak broadening and the less distorted crystal structure compared to the other samples.

In conclusion the strong influence of APBs on the spin structure and subsequently on the macroscopic magnetization could be shown with Monte Carlo simulations. Simulations of X-ray powder diffraction patterns allowed the development of a method to detect and assess the amount of APBs in real samples. With experiments on particle samples with various sizes the influences of the particle size, the composition, the number and ordering of defects, the presence and thickness of a non-magnetic surface layer as well as the presence and quantity of APBs was investigated. It was shown that while for smaller particles the non-magnetic surface and canting around vacancies plays the major role, for increasing particle sizes the effects of APBs become increasingly important.

Chapter 6
Outlook

Some questions remain that could be addressed in future works. The presented experimental findings as well as the simulations, especially the signature of antiphase boundaries in the X-ray diffraction patterns provide a solid basis for these possible investigations. For example further small-angle neutron scattering with polarized neutrons (SANSPOL) especially on small particles with sizes below 10 nm will help to conclusively answer the question of the size dependence of a non-magnetic surface layer. However, a challenge will be the small signal generated by these small nanoparticles as well as the difficulties in preparing samples with a small size distribution that is necessary for a precise analysis of the subtle contribution of the surface layer. With the distinct APB signature in XRD patterns an in-situ heating study could be performed, where the development of the peak widths is monitored with increasing temperature. It is possible that the increased temperature allows the rearrangement of the crystal lattice thereby removing APBs, which would lead to an increase in the saturation magnetization and thus an improvement of the particle performance for several applications. Finally, neutron pair distribution function analysis could be used to study the local magnetic structure and the spin structure disturbances introduced by antiphase boundaries or other structural defects. The developed programs can be used for further simulation studies, e.g. of surface effects, mutliple APBs and different particle shapes. Further development might include the introduction of lattice strains, thermal displacements or core-shell structures. The adaptation to different systems is also possible.

Bibliography

[1] R. Bailey. “Lesser known applications of ferrofluids.” *J. Magn. Magn. Mater* **39**, 1-2 (1983), 178–182.

[2] K. Raj, B. Moskowitz, and R. Casciari. “Advances in ferrofluid technology.” *J. Magn. Magn. Mater* **149**, 1-2 (1995), 174–180.

[3] Y.-P. Sun. *Supercritical Fluid Technology in Materials Science and Engineering: Syntheses: Properties, and Applications.* Crc Press (2002).

[4] E. Jol, I. Kamei, and W. Jones. “Induction charging system, US 9,479,007 B1.” (2016).

[5] L. Josephson, C.-H. Tung, A. Moore, and R. Weissleder. “High-efficiency intracellular magnetic labeling with novel superparamagnetic-Tat peptide conjugates.” *Bioconjugate Chem.* **10**, 2 (1999), 186–191.

[6] R. C. Semelka and T. K. Helmberger. “Contrast agents for MR imaging of the liver.” *Radiology* **218**, 1 (2001), 27–38.

[7] C. Corot, P. Robert, J.-M. Idée, and M. Port. “Recent advances in iron oxide nanocrystal technology for medical imaging.” *Adv. Drug Deliv. Rev* **58**, 14 (2006), 1471–1504.

[8] Y.-W. Jun, J.-H. Lee, and J. Cheon. “Chemical design of nanoparticle probes for high-performance magnetic resonance imaging.” *Angew. Chem. Int. Ed.* **47**, 28 (2008), 5122–5135.

[9] S. Laurent, D. Forge, M. Port, A. Roch, C. Robic, L. Vander Elst, and R. N. Muller. “Magnetic iron oxide nanoparticles: synthesis, stabilization, vectorization, physicochemical characterizations, and biological applications.” *Chem. Rev.* **108**, 6 (2008), 2064–2110.

[10] C. Sun, J. S. Lee, and M. Zhang. “Magnetic nanoparticles in MR imaging and drug delivery.” *Adv. Drug Deliv. Rev* **60**, 11 (2008), 1252–1265.

[11] X. Ma, A. Gong, B. Chen, J. Zheng, T. Chen, Z. Shen, and A. Wu. “Exploring a new SPION-based MRI contrast agent with excellent water-dispersibility, high specificity to cancer cells and strong MR imaging efficacy.” *Colloids Surf. B* **126** (2015), 44–49.

[12] C. Alexiou, W. Arnold, R. J. Klein, F. G. Parak, P. Hulin, C. Bergemann, W. Erhardt, S. Wagenpfeil, and A. S. Luebbe. “Locoregional cancer treatment with magnetic drug targeting.” *Cancer research* **60**, 23 (2000), 6641–6648.

[13] A. S. Lübbe, C. Alexiou, and C. Bergemann. “Clinical applications of magnetic drug targeting.” *J. Surg. Res.* **95**, 2 (2001), 200–206.

[14] R. Hergt, S. Dutz, R. Müller, and M. Zeisberger. "Magnetic particle hyperthermia: nanoparticle magnetism and materials development for cancer therapy." *J. Phys. Condens. Matter* **18**, 38 (2006), S2919.

[15] M. Gonzales-Weimuller, M. Zeisberger, and K. M. Krishnan. "Size-dependant heating rates of iron oxide nanoparticles for magnetic fluid hyperthermia." *J. Magn. Magn. Mater.* **321**, 13 (2009), 1947–1950.

[16] K. M. Krishnan. "Biomedical Nanomagnetics: A Spin Through Possibilities in Imaging, Diagnostics, and Therapy." *IEEE Trans. Magn.* **46**, 7 (2010), 2523–2558.

[17] C. S. Kumar and F. Mohammad. "Magnetic nanomaterials for hyperthermia-based therapy and controlled drug delivery." *Adv. Drug Deliv. Rev* **63**, 9 (2011), 789–808.

[18] S. Laurent, S. Dutz, U. O. Häfeli, and M. Mahmoudi. "Magnetic fluid hyperthermia: focus on superparamagnetic iron oxide nanoparticles." *Adv. Colloid Interface Sci.* **166**, 1-2 (2011), 8–23.

[19] A. E. Deatsch and B. A. Evans. "Heating efficiency in magnetic nanoparticle hyperthermia." *J. Magn. Magn. Mater* **354** (2014), 163–172.

[20] A. Lak, S. Disch, and P. Bender. "Embracing defects and disorder in magnetic nanoparticles." *Adv. Sci.* **8**, 7 (2021), 2002682.

[21] B. Thiesen and A. Jordan. "Clinical applications of magnetic nanoparticles for hyperthermia." *Int. J. Hyperthermia* **24**, 6 (2008), 467–474.

[22] K. Maier-Hauff, F. Ulrich, D. Nestler, H. Niehoff, P. Wust, B. Thiesen, H. Orawa, V. Budach, and A. Jordan. "Efficacy and safety of intratumoral thermotherapy using magnetic iron-oxide nanoparticles combined with external beam radiotherapy on patients with recurrent glioblastoma multiforme." *J. Neurooncol.* **103**, 2 (2011), 317–324.

[23] J. L. Urraca, B. Cortés-Llanos, C. Aroca, P. d. l. Presa, L. Pérez, and M. C. Moreno-Bondi. "Magnetic field-induced polymerization of molecularly imprinted polymers." *J. Phys. Chem. C* **122**, 18 (2018), 10189–10196.

[24] A. Bordet, L.-M. Lacroix, P.-F. Fazzini, J. Carrey, K. Soulantica, and B. Chaudret. "Magnetically induced continuous CO2 hydrogenation using composite iron carbide nanoparticles of exceptionally high heating power." *Angew. Chem. Int. Ed.* **55**, 51 (2016), 15894–15898.

[25] A. Lappas, G. Antonaropoulos, K. Brintakis, M. Vasilakaki, K. N. Trohidou, V. Iannotti, G. Ausanio, A. Kostopoulou, M. Abeykoon, I. K. Robinson, et al. "Vacancy-Driven Noncubic Local Structure and Magnetic Anisotropy Tailoring in Fe x O- Fe 3- δ O 4 Nanocrystals." *Phys. Rev. X* **9**, 4 (2019), 041044.

[26] P. V. Hendriksen, S. Linderoth, and P.-A. Lindgård. "Finite-size modifications of the magnetic properties of clusters." *Phys. Rev. B* **48**, 10 (1993), 7259.

[27] A. Berkowitz, W. Schuele, and P. Flanders. “Influence of crystallite size on the magnetic properties of acicular γ-Fe2O3 particles.” *J. Appl. Phys.* **39**, 2 (1968), 1261–1263.

[28] J. M. D. Coey. “Noncollinear spin arrangement in ultrafine ferrimagnetic crystallites.” *Phys. Rev. Lett.* **27**, 17 (1971), 1140.

[29] A. Morrish and K. Haneda. “Surface magnetic properties of fine particles.” *J. Magn. Magn. Mater.* **35**, 1-3 (1983), 105–113.

[30] F. Parker, M. Foster, D. Margulies, and A. Berkowitz. “Spin canting, surface magnetization, and finite-size effects in γ-Fe2O3 particles.” *Phys. Rev. B* **47**, 13 (1993), 7885.

[31] M. Morales, C. Serna, F. Bødker, and S. Mørup. “Spin canting due to structural disorder in maghemite.” *J. Phys. Condens. Matter* **9**, 25 (1997), 5461.

[32] K. Butter, A. Hoell, A. Wiedenmann, A. V. Petukhov, and G.-J. Vroege. “Small-angle neutron and X-ray scattering of dispersions of oleic-acid-coated magnetic iron particles.” *J. Appl. Crystallogr.* **37**, 6 (2004), 847–856.

[33] D. Li, W. Y. Teoh, C. Selomulya, R. C. Woodward, P. Munroe, and R. Amal. “Insight into microstructural and magnetic properties of flame-made γ-Fe2O3 nanoparticles.” *J. Mater. Chem.* **17**, 46 (2007), 4876–4884.

[34] T. Daou, J. Greneche, G. Pourroy, S. Buathong, A. Derory, C. Ulhaq-Bouillet, B. Donnio, D. Guillon, and S. Begin-Colin. “Coupling agent effect on magnetic properties of functionalized magnetite-based nanoparticles.” *Chem. Mater.* **20**, 18 (2008), 5869–5875.

[35] A. Roca, D. Niznansky, J. Poltierova-Vejpravova, B. Bittova, M. González-Fernández, C. Serna, and M. Morales. “Magnetite nanoparticles with no surface spin canting.” *J. Appl. Phys.* **105**, 11 (2009), 114309.

[36] K. L. Krycka, R. Booth, J. A. Borchers, W. Chen, C. Conlon, T. Gentile, C. Hogg, Y. Ijiri, M. Laver, B. Maranville, et al. “Resolving 3D magnetism in nanoparticles using polarization analyzed SANS.” *Physica B Condens. Matter* **404**, 17 (2009), 2561–2564.

[37] K. L. Krycka, R. A. Booth, C. R. Hogg, Y. Ijiri, J. A. Borchers, W. Chen, S. Watson, M. Laver, T. R. Gentile, L. R. Dedon, et al. “Core-shell magnetic morphology of structurally uniform magnetite nanoparticles.” *Phys. Rev. Lett.* **104**, 20 (2010), 207203.

[38] K. L. Krycka, J. A. Borchers, R. Booth, C. Hogg, Y. Ijiri, W. Chen, S. Watson, M. Laver, T. Gentile, S. Harris, et al. “Internal magnetic structure of magnetite nanoparticles at low temperature.” *J. Appl. Phys.* **107**, 9 (2010), 09B525.

[39] K. L. Krycka, J. A. Borchers, R. Booth, Y. Ijiri, K. Hasz, J. Rhyne, and S. Majetich. “Origin of surface canting within Fe 3 O 4 nanoparticles.” *Phys. Rev. Lett.* **113**, 14 (2014), 147203.

[40] A. Michels, D. Honecker, S. Erokhin, and D. Berkov. "Comment on "Origin of Surface Canting within Fe 3 O 4 Nanoparticles"." *Phys. Rev. Lett.* **114**, 14 (2015), 149701.

[41] K. Krycka, J. Borchers, R. Booth, Y. Ijiri, K. Hasz, J. Rhyne, and S. Majetich. "Krycka et al. Reply." *Phys. Rev. Lett.* **114**, 14 (2015), 149702.

[42] P. Dutta, S. Pal, M. Seehra, N. Shah, and G. Huffman. "Size dependence of magnetic parameters and surface disorder in magnetite nanoparticles." *J. Appl. Phys.* **105**, 7 (2009), 07B501.

[43] S. Disch, E. Wetterskog, R. P. Hermann, A. Wiedenmann, U. Vainio, G. Salazar-Alvarez, L. Bergström, and T. Brückel. "Quantitative spatial magnetization distribution in iron oxide nanocubes and nanospheres by polarized small-angle neutron scattering." *New J. Phys.* **14**, 1 (2012), 013025.

[44] D. Zákutná, D. Niznansky, L. Barnsley, A. Feoktystov, D. Honecker, and S. Disch. "Field-Dependence of Magnetic Disorder in Nanoparticles." *Phys. Rev. X* **10** (2020), 031019.

[45] M. Levy, A. Quarta, A. Espinosa, A. Figuerola, C. Wilhelm, M. García-Hernández, A. Genovese, A. Falqui, D. Alloyeau, R. Buonsanti, P. D. Cozzoli, M. A. García, F. Gazeau, and T. Pellegrino. "Correlating magneto-structural properties to hyperthermia performance of highly monodisperse iron oxide nanoparticles prepared by a seeded-growth route." *Chem. Mater.* **23**, 18 (2011), 4170–4180.

[46] E. Wetterskog, C.-W. Tai, J. Grins, L. Bergström, and G. Salazar-Alvarez. "Anomalous magnetic properties of nanoparticles arising from defect structures: topotaxial oxidation of Fe(1-x)O—Fe(3-δ)O4 core—shell nanocubes to single-phase particles." *ACS nano* **7**, 8 (2013), 7132–7144.

[47] Z. Nedelkoski, D. Kepaptsoglou, L. Lari, T. Wen, R. A. Booth, S. D. Oberdick, P. L. Galindo, Q. M. Ramasse, R. F. Evans, S. Majetich, and V. K. Lazarov. "Origin of reduced magnetization and domain formation in small magnetite nanoparticles." *Sci. Rep.* **7**, 1 (2017), 1–8.

[48] H. Sharifi Dehsari, V. Ksenofontov, A. Möller, G. Jakob, and K. Asadi. "Determining Magnetite/Maghemite Composition and Core-Shell Nanostructure from Magnetization Curve for Iron Oxide Nanoparticles." *J. Phys. Chem. C* **122**, 49 (2018), 28292–28301.

[49] S. Mühlbauer, D. Honecker, É. A. Périgo, F. Bergner, S. Disch, A. Heinemann, S. Erokhin, D. Berkov, C. Leighton, M. R. Eskildsen, et al. "Magnetic small-angle neutron scattering." *Rev. Mod. Phys.* **91**, 1 (2019), 015004.

[50] H. L. Andersen, B. A. Frandsen, H. P. Gunnlaugsson, M. R. Jørgensen, S. J. Billinge, K. Jensen, and M. Christensen. "Local and long-range atomic/magnetic structure of non-stoichiometric spinel iron oxide nanocrystallites." *IUCrJ* **8**, 1 (2021).

[51] M. Herlitschke, S. Disch, I. Sergueev, K. Schlage, E. Wetterskog, L. Bergström, and R. P. Hermann. "Spin disorder in maghemite nanoparticles investigated using polarized neutrons and nuclear resonant scattering." *J. Phys. Conf. Ser.* **711**, 1 (2016), 012002.

[52] J. Park, K. An, Y. Hwang, J.-G. Park, H.-J. Noh, J.-Y. Kim, J.-H. Park, N.-M. Hwang, and T. Hyeon. "Ultra-large-scale syntheses of monodisperse nanocrystals." *Nat. Mater.* **3**, 12 (2004), 891–895.

[53] S. Blundell. *Magnetism in Condensed Matter.* Oxford Master Series in Condensed Matter Physics. OUP Oxford (2001).

[54] M. Getzlaff. *Fundamentals of Magnetism.* Springer-Verlag Berlin Heidelberg (2008).

[55] W. Nolting and A. Ramakanth. *Quantum Theory of Magnetism.* Springer-Verlag Berlin Heidelberg (2009).

[56] D. J. Griffiths. *Introduction to electrodynamics.* Pearson Education Limited, 4 edition (2013).

[57] C.-G. Stefanita. *Magnetism.* Springer-Verlag Berlin Heidelberg (2012).

[58] S. Bedanta and W. Kleemann. "Supermagnetism." *J. Phys. D: Appl. Phys.* **42**, 1 (2008), 013001.

[59] J. B. Goodenough. *Magnetism and the chemical bond.* Interscience Publishers (1963).

[60] R. H. Kodama and A. E. Berkowitz. "Atomic-scale magnetic modeling of oxide nanoparticles." *Phys. Rev. B* **59**, 9 (1999), 6321.

[61] M. Uhl and B. Siberchicot. "A first-principles study of exchange integrals in magnetite." *J. Phys. Condens. Matter* **7**, 22 (1995), 4227.

[62] P. Bruno. "Physical origins and theoretical models of magnetic anisotropy." In "Magnetismus von Festkörpern und grenzflächen," chapter 24. Ferienkurse des Forschungszentrums Jülich (1993), pages 24.1–24.28.

[63] R. C. O'handley. *Modern magnetic materials: principles and applications.* John Wiley & Sons, Inc. (2000).

[64] G. Goya, T. Berquo, F. Fonseca, and M. Morales. "Static and dynamic magnetic properties of spherical magnetite nanoparticles." *J. Appl. Phys.* **94**, 5 (2003), 3520–3528.

[65] J. Garcıa-Otero, M. Porto, J. Rivas, and A. Bunde. "Influence of the cubic anisotropy constants on the hysteresis loops of single-domain particles: A Monte Carlo study." *J. Appl. Phys.* **85**, 4 (1999), 2287–2292.

[66] R. F. Evans, W. J. Fan, P. Chureemart, T. A. Ostler, M. O. Ellis, and R. W. Chantrell. "Atomistic spin model simulations of magnetic nanomaterials." *J. Phys. Condens. Matter* **26**, 10 (2014), 103202.

[67] O. Petracic. "Superparamagnetic nanoparticle ensembles." *Superlattices Microstruct.* **47**, 5 (2010), 569–578.

[68] J. Nogués and I. K. Schuller. "Exchange bias." *J. Magn. Magn. Mater.* **192**, 2 (1999), 203–232.

[69] J. Nogués, J. Sort, V. Langlais, V. Skumryev, S. Suriñach, J. Muñoz, and M. Baró. "Exchange bias in nanostructures." *Phys. Rep.* **422**, 3 (2005), 65–117.

[70] S. Arora, R. Sofin, A. Nolan, and I. Shvets. "Antiphase boundaries induced exchange coupling in epitaxial Fe3O4 thin films." *J. Magn. Magn. Mater.* **286** (2005), 463–467.

[71] U. Nowak, K.-D. Usadel, J. Keller, P. Miltényi, B. Beschoten, and G. Güntherodt. "Domain state model for exchange bias. I. Theory." *Phys. Rev. B* **66**, 1 (2002), 014430.

[72] P. Miltényi, M. Gierlings, J. Keller, B. Beschoten, G. Güntherodt, U. Nowak, and K.-D. Usadel. "Diluted antiferromagnets in exchange bias: Proof of the domain state model." *Phys. Rev. Lett.* **84**, 18 (2000), 4224.

[73] A. Malozemoff. "Random-field model of exchange anisotropy at rough ferromagnetic-antiferromagnetic interfaces." *Phys. rev. B* **35**, 7 (1987), 3679.

[74] C. Martínez-Boubeta, K. Simeonidis, M. Angelakeris, N. Pazos-Pérez, M. Giersig, A. Delimitis, L. Nalbandian, V. Alexandrakis, and D. Niarchos. "Critical radius for exchange bias in naturally oxidized Fe nanoparticles." *Phys. Rev. B* **74**, 5 (2006), 054430.

[75] H. Khurshid, W. Li, M.-H. Phan, P. Mukherjee, G. C. Hadjipanayis, and H. Srikanth. "Surface spin disorder and exchange-bias in hollow maghemite nanoparticles." *Appl. Phys. Lett.* **101**, 2 (2012), 022403.

[76] Y. Hwang, S. Angappane, J. Park, K. An, T. Hyeon, and J.-G. Park. "Exchange bias behavior of monodisperse Fe3O4/γ-Fe2O3 core/shell nanoparticles." *Curr. Appl. Phys.* **12**, 3 (2012), 808–811.

[77] B. Lilley. "LXXI. Energies and widths of domain boundaries in ferromagnetics." *London, Edinburgh Dublin Philos. Mag. J. Sci.* **41**, 319 (1950), 792–813.

[78] C. Kittel. "Theory of the structure of ferromagnetic domains in films and small particles." *Phys. Rev.* **70**, 11-12 (1946), 965.

[79] R. S. Williams, J. C. Figueroa, and C. D. Graham. "Effect of the Second Anisotropy Constant K2 on the Orientation Dependence of Domain Wall Energy." *AIP Conf. Proc.* **34**, 1 (1976), 102–104.

[80] B. M. Moskowitz and S. L. Halgedahl. "Theoretical temperature and grain-size dependence of domain state in x= 0.6 titanomagnetite." *J. Geophys. Res. Solid Earth* **92**, B10 (1987), 10667–10682.

[81] B. Moskowitz and S. Banerjee. “Grain size limits for pseudosingle domain behavior in magnetite: Implications for paleomagnetism.” *IEEE Trans. Magn.* **15**, 5 (1979), 1241–1246.

[82] D. L. Leslie-Pelecky and R. D. Rieke. “Magnetic properties of nanostructured materials.” *Chem. Mater.* **8**, 8 (1996), 1770–1783.

[83] R. F. Butler and S. K. Banerjee. “Theoretical single-domain grain size range in magnetite and titanomagnetite.” *J. Geophys. Res.* **80**, 29 (1975), 4049–4058.

[84] T. Köhler, A. Feoktystov, O. Petracic, E. Kentzinger, T. Bhatnagar-Schöffmann, M. Feygenson, N. Nandakumaran, J. Landers, H. Wende, A. Cervellino, U. Rücker, R. Dunin-Borkowski, and T. Brückel. “Mechanism of magnetization reduction in iron oxide nanoparticles.” *Nanoscale* **13**, 14 (2021), 6965–6976.

[85] E. C. Stoner and E. Wohlfarth. “A mechanism of magnetic hysteresis in heterogeneous alloys.” *Philos. Trans. Royal Soc. A* **240**, 826 (1948), 599–642.

[86] S. Bedanta, O. Petracic, and W. Kleemann. “Supermagnetism.” In “Handbook of magnetic materials,” volume 23. Elsevier (2015), pages 1–83.

[87] O. Petracic, X. Chen, S. Bedanta, W. Kleemann, S. Sahoo, S. Cardoso, and P. Freitas. “Collective states of interacting ferromagnetic nanoparticles.” *J. Magn. Magn. Mater.* **300**, 1 (2006), 192–197.

[88] F. Wiekhorst, E. Shevchenko, H. Weller, and J. Kötzler. “Anisotropic superparamagnetism of monodispersive cobalt-platinum nanocrystals.” *Phys. Rev. B* **67**, 22 (2003), 224416.

[89] D. P. Landau and K. Binder. *A guide to Monte Carlo simulations in statistical physics.* Cambridge university press (2014).

[90] K. Binder and D. Heermann. *Monte Carlo simulation in statistical physics.* Springer (2019).

[91] K. Binder. *The Monte Carlo method in condensed matter physics.* Springer, Berlin, Heidelberg, 2 edition (1995).

[92] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of state calculations by fast computing machines.” *Chem. Phys.* **21**, 6 (1953), 1087–1092.

[93] D. Hinzke and U. Nowak. “Monte Carlo simulation of magnetization switching in a Heisenberg model for small ferromagnetic particles.” *Comput. Phys. Commun.* **121** (1999), 334–337.

[94] G. Marsaglia, W. W. Tsang, et al. “The ziggurat method for generating random variables.” *J. Stat. Softw.* **5**, 8 (2000), 1–7.

[95] G. Marsaglia et al. “Choosing a point from the surface of a sphere.” *Ann. math. stat.* **43**, 2 (1972), 645–646.

[96] S. Kirkpatrick and E. P. Stoll. “A very fast shift-register sequence random number generator.” *J. Comput. Phys.* **40**, 2 (1981), 517–526.

[97] W. Greiner. *Quantum mechanics: an introduction.* Springer Science & Business Media (2011).

[98] L. De Broglie. *Recherches sur la théorie des quanta.* Ph.D. thesis, Migration-université en cours d’affectation (1924).

[99] G. L. Squires. *Introduction to the theory of thermal neutron scattering.* Cambridge University Press (1996).

[100] H. L. Monaco and G. Artioli. “Experimental methods in X-ray and neutron crystallography.” In C. Giacovazzo, editor, “Fundamentals of Crystallography,” chapter 5. Oxford University Press (2011), pages 157–234.

[101] C. Giacovazzo. “The diffraction of X-rays by crystals.” In C. Giacovazzo, editor, “Fundamentals of Crystallography,” chapter 3. Oxford University Press (2011), pages 157–234.

[102] W. Massa. *Kristallstrukturbestimmung.* Springer (2007).

[103] U. Klemradt. “Synchrotron radiation sources and instrumentation.” In M. Angst, T. Brückel, S. Förster, K. Friese, and R. Zorn, editors, “Lecture Notes of the 50th IFF Spring School ”Scattering! Soft, Functional and Quantum Materials”,” chapter C3. Forschungszentrum Jülich GmbH (2019).

[104] H. Friedrich. *Scattering Theory.* Springer (2013).

[105] D. J. Griffiths. *Introduction to quantum mechanics.* Cambridge University Press, 3 edition (2010).

[106] L. Feigin, D. I. Svergun, et al. *Structure analysis by small-angle X-ray and neutron scattering.* Springer (1987).

[107] J. M. Cowley. *Diffraction physics.* Elsevier (1995).

[108] E. M. Anitas. “Small-Angle Scattering Technique.” In “Small-Angle Scattering (Neutrons, X-Rays, Light) from Complex Systems,” Springer (2019), pages 33–63.

[109] B. E. Warren. *X-ray Diffraction.* Dover Publications, Inc. (1990).

[110] P. Debye. “Zerstreuung von Röntgenstrahlen.” *Ann. Phys.* **351**, 6 (1915), 809–823.

[111] T. Egami and S. J. Billinge. *Underneath the Bragg peaks: structural analysis of complex materials*, volume 16 of *Pergamon Materials Series.* Pergamon (2012).

[112] A. Cervellino, C. Giannini, and A. Guagliardi. “DEBUSSY: a Debye user system for nanocrystalline materials.” *J. Appl. Crystallogr.* **43**, 6 (2010), 1543–1547.

[113] W. H. Bragg and W. L. Bragg. “The reflection of X-rays by crystals.” *Proc. Math. Phys. Eng. Sci.* **88**, 605 (1913), 428–438.

[114] A. Guinier, G. Fournet, and K. L. Yudowitch. *Small-angle scattering of X-rays.* Wiley New York (1955).

[115] P. W. Schmidt. "Some fundamental concepts and techniques useful in small-angle scattering studies of disordered solids." In "Modern aspects of small-angle scattering," Springer (1995), pages 1–56.

[116] G. Porod. "General Theory." In O. Glatter and K. O, editors, "Small Angle X-ray Scattering," Academic Press, London (1982), pages 17–51.

[117] J. S. Pedersen. "Analysis of small-angle scattering data from colloids and polymer solutions: modeling and least-squares fitting." *Adv. Colloid Interface Sci.* **70** (1997), 171–210.

[118] C. Forbes, M. Evans, N. Hastings, and B. Peacock. *Statistical distributions.* John Wiley & Sons (2011).

[119] J. S. Pedersen, D. Posselt, and K. Mortensen. "Analytical treatment of the resolution function for small-angle scattering." *J. Appl. Crystallogr.* **23**, 4 (1990), 321–333.

[120] R. Borsali and R. Pecora. *Soft-matter characterization.* Springer Science & Business Media (2008).

[121] R. Moon, T. Riste, and W. Koehler. "Polarization analysis of thermal-neutron scattering." *Phys. Rev.* **181**, 2 (1969), 920.

[122] R. Pynn, J. B. Hayter, and S. W. Charles. "Determination of ferrofluid structure by neutron polarization analysis." *Phys. Rev. Lett.* **51**, 8 (1983), 710.

[123] A. Wiedenmann. "Small-angle neutron scattering investigations of magnetic nanostructures using polarized neutrons." *J. Appl. Crystallogr.* **33**, 3 (2000), 428–432.

[124] A. Wiedenmann, A. Hoell, and M. Kammel. "Small-angle scattering investigations of cobalt-ferrofluids using polarised neutrons." *J. Magn. Magn. Mater.* **252** (2002), 83–85.

[125] M. V. Avdeev. "Contrast variation in small-angle scattering experiments on polydisperse and superparamagnetic systems: basic functions approach." *J. Appl. Crystallogr.* **40**, 1 (2007), 56–70.

[126] O. Glatter and R. May. *Small-angle techniques*, chapter 2.6. American Cancer Society (2006), pages 89–112.

[127] S. Menon, C. Manohar, and K. S. Rao. "A new interpretation of the sticky hard sphere model." *J. Chem. Phys.* **95**, 12 (1991), 9186–9190.

[128] H. M. Rietveld. "A profile refinement method for nuclear and magnetic structures." *J. appl. Crystallogr.* **2**, 2 (1969), 65–71.

[129] P. Scardi, M. Ortolani, and M. Leoni. "WPPM: microstructural analysis beyond the Rietveld method." In "Materials Science Forum," volume 651. Trans Tech Publ (2010), pages 155–171.

[130] P. Scherrer. *Bestimmung der inneren Struktur und der Größe von Kolloidteilchen mittels Röntgenstrahlen.* Springer Berlin Heidelberg, Berlin, Heidelberg (1912), pages 387–409.

[131] A. Patterson. “The Scherrer formula for X-ray particle size determination.” *Phys. Rev* **56**, 10 (1939), 978.

[132] A. Monshi, M. R. Foroughi, and M. R. Monshi. “Modified Scherrer equation to estimate more accurately nano-crystallite size using XRD.” *World J. Nano Sci. Eng.* **2**, 3 (2012), 154–160.

[133] P. Scardi, M. Leoni, and R. Delhez. “Line broadening analysis using integral breadth methods: a critical review.” *J. Appl. Crystallogr.* **37**, 3 (2004), 381–390.

[134] P. Scardi and M. Leoni. “Line profile analysis: pattern modelling versus profile fitting.” *J. Appl. Crystallogr.* **39**, 1 (2006), 24–31.

[135] P. Scardi, M. Leoni, and K. R. Beyerlein. “On the modelling of the powder pattern from a nanocrystalline material.” *Z. Kristallogr. -Cryst. Mater.* **226**, 12 (2011), 924–933.

[136] A. Leonardi, M. Leoni, S. Siboni, and P. Scardi. “Common volume functions and diffraction line profiles of polyhedral domains.” *J. Appl. Crystallogr.* **45**, 6 (2012), 1162–1172.

[137] P. Scardi and M. Leoni. “Fourier modelling of the anisotropic line broadening of X-ray diffraction profiles due to line and plane lattice defects.” *J. Appl. Crystallogr.* **32**, 4 (1999), 671–682.

[138] C. A. Young and A. L. Goodwin. “Applications of pair distribution function methods to contemporary problems in materials chemistry.” *J. Mater. Chem* **21**, 18 (2011), 6464–6476.

[139] H. Y. Playford, A. C. Hannon, M. G. Tucker, D. M. Dawson, S. E. Ashbrook, R. J. Kastiban, J. Sloan, and R. I. Walton. “Characterization of structural disorder in γ-Ga2O3.” *J. Phys. Chem. C* **118**, 29 (2014), 16188–16198.

[140] A. Mancini and L. Malavasi. “Recent advances in the application of total scattering methods to functional materials.” *Chem. Commun.* **51**, 93 (2015), 16592–16604.

[141] P. Juhás, T. Davis, C. L. Farrow, and S. J. Billinge. “PDFgetX3: a rapid and highly automatable program for processing powder diffraction data into total scattering pair distribution functions.” *J. Appl. Crystallogr.* **46**, 2 (2013), 560–566.

[142] C. Farrow, P. Juhas, J. Liu, D. Bryndin, E. Božin, J. Bloch, T. Proffen, and S. Billinge. “PDFfit2 and PDFgui: computer programs for studying nanostructure in crystals.” *J. Phys. Condens. Matter* **19**, 33 (2007), 335219.

[143] D. B. Williams and C. B. Carter. *Transmission electron microscopy.* Springer Science+Business Media (2009).

[144] P. W. Hawkes and J. C. Spence. *Springer Handbook of Microscopy.* Springer Nature (2019).

[145] R. F. Egerton et al. *Physical principles of electron microscopy*, volume 56. Springer (2005).

[146] J. Cowley. "The electron-optical imaging of crystal lattices." *Acta Crystallogr.* **12**, 5 (1959), 367–375.

[147] L. Reimer. *Transmission electron microscopy: physics of image formation and microanalysis*, volume 36. Springer (2013).

[148] T. C. Gibb. *Principles of Mössbauer spectroscopy.* Springer (2013).

[149] P. Gütlich, E. Bill, and A. X. Trautwein. *Mössbauer spectroscopy and transition metal chemistry: fundamentals and applications.* Springer Science & Business Media (2010).

[150] J. Landers, S. Salamon, H. Remmer, F. Ludwig, and H. Wende. "In-Field Orientation and Dynamics of Ferrofluids Studied by Mössbauer Spectroscopy." *ACS Appl. Mater. Interfaces* **11**, 3 (2018), 3160–3168.

[151] G. Da Costa, E. De Grave, and R. Vandenberghe. "Mössbauer studies of magnetite and Al-substituted maghemites." *Hyperfine Interact.* **117**, 1 (1998), 207–243.

[152] D. Jones and K. Srivastava. "Many-state relaxation model for the Mössbauer spectra of superparamagnets." *Phys. Rev. B* **34**, 11 (1986), 7542.

[153] K. Momma and F. Izumi. "VESTA: a three-dimensional visualization system for electronic and structural analysis." *J. Appl. Crystallogr.* **41**, 3 (2008), 653–658.

[154] U. Schwertmann and R. M. Cornell. *The Iron Oxides.* John Wiley & Sons, Ltd (2003).

[155] M.-H. Phan, J. Alonso, H. Khurshid, P. Lampen-Kelley, S. Chandra, K. Stojak Repa, Z. Nemati, R. Das, Ó. Iglesias, and H. Srikanth. "Exchange bias effects in iron oxide-based nanoparticle systems." *Nanomaterials* **6**, 11 (2016), 221.

[156] F. Bosi, U. Hålenius, and H. Skogby. "Crystal chemistry of the magnetite-ulvospinel series." *Am. Mineral.* **94**, 1 (2009), 181–189.

[157] A. Cervellino, R. Frison, G. Cernuto, A. Guagliardi, and N. Masciocchi. "Lattice parameters and site occupancy factors of magnetite-maghemite core-shell nanoparticles. A critical study." *J. Appl. Crystallogr.* **47**, 5 (2014), 1755–1761.

[158] D. Levy, R. Giustetto, and A. Hoser. "Structure of magnetite (Fe3O4) above the Curie temperature: a cation ordering study." *Phys. Chem. Miner.* **39**, 2 (2012), 169–176.

[159] A. Figueroa, J. Bartolomé, L. García, F. Bartolomé, A. Arauzo, A. Millán, and F. Palacio. "Magnetic anisotropy of maghemite nanoparticles probed by RF transverse susceptibility." *Phys. Procedia* **75** (2015), 1050–1057.

[160] S. P. Gubin, Y. A. Koksharov, G. Khomutov, and G. Y. Yurkov. "Magnetic nanoparticles: preparation, structure and properties." *Russ. Chem. Rev.* **74**, 6 (2005), 489.

[161] J. Birks. "The properties of ferromagnetic compounds at centimetre wavelengths." *Proc. Phys. Soc. B* **63**, 2 (1950), 65.

[162] C. Goss. "Saturation magnetisation, coercivity and lattice parameter changes in the system Fe3O4-γFe2O3, and their relationship to structure." *Phys. Chem. Miner.* **16**, 2 (1988), 164–171.

[163] S. J. Kemp, R. M. Ferguson, A. P. Khandhar, and K. M. Krishnan. "Monodisperse magnetite nanoparticles with nearly ideal saturation magnetization." *RSC advances* **6**, 81 (2016), 77452–77464.

[164] A. Millan, A. Urtizberea, N. Silva, F. Palacio, V. Amaral, E. Snoeck, and V. Serin. "Surface effects in maghemite nanoparticles." *J. Magn. Magn. Mater.* **312**, 1 (2007), L5–L9.

[165] R. J. Hill, J. R. Craig, and G. Gibbs. "Systematics of the spinel structure type." *Phys. Chem. Miner.* **4**, 4 (1979), 317–339.

[166] J. Loos and P. Novák. "Double exchange and superexchange in a ferrimagnetic half-metal." *Phys. Rev. B* **66**, 13 (2002), 132403.

[167] E. Verwey and P. Haayman. "Electronic conductivity and transition point of magnetite ("Fe3O4")." *Physica* **8**, 9 (1941), 979–987.

[168] M. S. Senn, J. P. Wright, and J. P. Attfield. "Charge order and three-site distortions in the Verwey structure of magnetite." *Nature* **481**, 7380 (2012), 173–176.

[169] R. Aragón, D. J. Buttrey, J. P. Shepherd, and J. M. Honig. "Influence of nonstoichiometry on the Verwey transition." *Phys. Rev. B* **31**, 1 (1985), 430.

[170] K. Kelm and W. Mader. "The symmetry of ordered cubic γ-Fe2O3 investigated by TEM." *Z. Naturforsch. B* **61**, 6 (2006), 665–671.

[171] P. Sidhu, R. Gilkes, and A. Posner. "Mechanism of the low temperature oxidation of synthetic magnetites." *J. Inorg. Nucl. Chem.* **39**, 11 (1977), 1953–1958.

[172] A. J. C. Wilson. "The reflexion of X-rays from the 'anti-phase nuclei' of AuCu3." *Proc. Math. Phys. Eng. Sci.* **181**, 987 (1943), 360–368.

[173] R. Fisher and M. Marcinkowski. "Direct observation of antiphase boundaries in the AuCu3 superlattice." *Philos. Mag.* **6**, 71 (1961), 1385–1405.

[174] A. J. C. Wilson and L. Zsoldos. "The reflexion of X-rays from the 'anti-phase nuclei' of AuCu3. II." *Proc. Math. Phys. Eng. Sci.* **290**, 1423 (1966), 508–514.

[175] P. Scardi and M. Leoni. "Diffraction whole-pattern modelling study of anti-phase domains in Cu3Au." *Acta Mater.* **53**, 19 (2005), 5229–5239.

[176] R. Kikuchi and J. W. Cahn. "Theory of interphase and antiphase boundaries in FCC alloys." *Acta Metall.* **27**, 8 (1979), 1337–1353.

[177] T. Hibma, F. Voogt, L. Niesen, P. Van der Heijden, W. De Jonge, J. Donkers, and P. Van der Zaag. "Anti-phase domains and magnetism in epitaxial magnetite layers." *J. Appl. Phys.* **85**, 8 (1999), 5291–5293.

[178] W. Eerenstein, T. Palstra, and T. Hibma. "Spin-valve behaviour of antiferromagnetic boundaries in ultrathin magnetite films." *Thin Solid Films* **400**, 1-2 (2001), 90–94.

[179] W. Eerenstein, T. Palstra, S. Saxena, and T. Hibma. "Spin-polarized transport across sharp antiferromagnetic boundaries." *Phys. Rev. Lett.* **88**, 24 (2002), 247204.

[180] W. Eerenstein, T. Palstra, T. Hibma, and S. Celotto. "Origin of the increased resistivity in epitaxial Fe3O4 films." *Phys. Rev. B* **66**, 20 (2002), 201101.

[181] M. Luysberg, R. Sofin, S. Arora, and I. Shvets. "Strain relaxation in Fe3O4/MgAl2O4 heterostructures: Mechanism for formation of antiphase boundaries in an epitaxial system with identical symmetries of film and substrate." *Phys. Rev. B* **80**, 2 (2009), 024111.

[182] D. Gilks, L. Lari, J. Naughton, O. Cespedes, Z. Cai, A. Gerber, S. Thompson, K. Ziemer, and V. Lazarov. "Origin of anomalous magnetite properties in crystallographic matched heterostructures: Fe3O4 (111)/MgAl2O4 (111)." *J. Phys. Condens. Matter* **25**, 48 (2013), 485004.

[183] R. Moreno, S. Jenkins, A. Skeparovski, Z. Nedelkoski, A. Gerber, V. K. Lazarov, and R. F. Evans. "Role of anti-phase boundaries in the formation of magnetic domains in magnetite thin films." *J. Phys. Condens. Matter* **33**, 17 (2021), 175802.

[184] K. P. McKenna, F. Hofer, D. Gilks, V. K. Lazarov, C. Chen, Z. Wang, and Y. Ikuhara. "Atomic-scale structure and properties of highly stable antiphase boundary defects in Fe3O4." *Nat. Commun.* **5**, 1 (2014), 1–8.

[185] J. Jakubovics, A. Lapworth, and T. Jolly. "Electron microscope studies of ferromagnetic ordered structures." *J. Appl. Phys.* **49**, 3 (1978), 2002–2006.

[186] M. Rudee, D. Margulies, and A. Berkowitz. "Antiphase domain boundaries in thin films of magnetite." *Microsc. Microanal.* **3**, 2 (1997), 126–129.

[187] W. Wu, X. Xiao, S. Zhang, T. Peng, J. Zhou, F. Ren, and C. Jiang. "Synthesis and magnetic properties of maghemite (γ-Fe2O3) short-nanotubes." *Nanoscale Res. Lett.* **5**, 9 (2010), 1474–1479.

[188] W. Bragg. "The structure of a cold-worked metal." *Proc. Phys. Soc.* **52**, 1 (1940), 105.

[189] W. Eerenstein, T. Palstra, T. Hibma, and S. Celotto. "Diffusive motion of antiphase domain boundaries in Fe3O4 films." *Phys. Rev. B* **68**, 1 (2003), 014428.

[190] Jülich Centre for Neutron Science. "GALAXI: Gallium anode low-angle X-ray instrument." *JLSRF* **2** (2016), A61.

[191] A. Hammersley. "FIT2D: a multi-purpose data reduction, analysis and visualization program." *J. Appl. Crystallogr.* **49**, 2 (2016), 646–652.

[192] Hilgenberg GmbH. "Technical data sheet 0500, version 1-09/2003." (2003).

[193] H. Frielinghaus, A. Feoktystov, I. Berts, and G. Mangiapia. "KWS-1: Small-angle scattering diffractometer." *JLSRF* **1** (2015), 28.

[194] A. V. Feoktystov, H. Frielinghaus, Z. Di, S. Jaksch, V. Pipich, M.-S. Appavou, E. Babcock, R. Hanslik, R. Engels, G. Kemmerling, et al. "KWS-1 high-resolution small-angle neutron scattering instrument at JCNS: current state." *J. Appl. Crystallogr.* **48**, 1 (2015), 61–70.

[195] V. Pipich. "QtiKWS: user-friendly program for reduction, visualization, analysis and fit of SA(N)S data." `http://www.qtisas.com` (2020).

[196] P. Willmott, D. Meister, S. Leake, M. Lange, A. Bergamaschi, M. Böge, M. Calvi, C. Cancellieri, N. Casati, A. Cervellino, et al. "The materials science beamline upgrade at the swiss light source." *J. Synchrotron Radiat.* **20**, 5 (2013), 667–682.

[197] B. H. Toby and R. B. Von Dreele. "GSAS-II: the genesis of a modern open-source all purpose crystallography software package." *J. Appl. Crystallogr.* **46**, 2 (2013), 544–549.

[198] R. H. Wendt and V. A. Fassel. "Induction-Coupled Plasma Spectrometric Excitation Source." *Anal. Chem.* **37**, 7 (1965), 920–922.

[199] R. Fagaly. "Superconducting quantum interference device instruments and applications." *Rev. Sci. Instrum.* **77**, 10 (2006), 101101.

[200] Quantum Design. *Magnetic Property Measurement System - MPMS-XL* (2011).

[201] Ernst Ruska-Centre for Microscopy and Spectroscopy with Electrons. "FEI Tecnai G2 F20." *JLSRF* **2** (2016), 77.

[202] C. T. Koch. *Determination of core structure periodicity and point defect density along dislocations.* Ph.D. thesis, Arizona State University (2002).

[203] C. A. Schneider, W. S. Rasband, and K. W. Eliceiri. "NIH Image to ImageJ: 25 years of image analysis." *Nat. Methods* **9**, 7 (2012), 671–675.

[204] N. Lohmann. "JSON for Modern C++, (version 3.9.1)." `https://github.com/nlohmann/json/` (2019).

[205] P. Ramachandran and G. Varoquaux. "Mayavi: 3D visualization of scientific data." *Comput. Sci. Eng.* **13**, 2 (2011), 40–51.

[206] P. J. Brown, A. G. Fox, E. N. Maslen, M. A. O'Keefe, and B. T. M. Willis. "Intensity of diffracted intensities." In "International Tables for Crystallography," chapter 6.1. American Cancer Society (2006), pages 554–595.

[207] M. Tokonami. “Atomic scattering factor for O2-.” *Acta Cryst.* **19**, 3 (1965), 486–486.

[208] D. M. Cooke, F. Alted, et al. “NumExpr: Fast numerical expression evaluator for NumPy, v2.6.9.” `https://github.com/pydata/numexpr` (2018).

[209] A. Cervellino, C. Giannini, and A. Guagliardi. “On the efficient evaluation of Fourier patterns for nanoparticles and clusters.” *J. Comput. Chem.* **27**, 9 (2006), 995–1008.

[210] A. Cervellino, R. Frison, F. Bertolotti, and A. Guagliardi. “DEBUSSY 2.0: the new release of a Debye user system for nanocrystalline and/or disordered materials.” *J. Appl. Crystallogr.* **48**, 6 (2015), 2026–2032.

[211] R. B. Neder and T. Proffen. *Diffuse Scattering and Defect Structure Simulations: A cook book using the program DISCUS.* Oxford University Press (2008).

[212] T. Köhler, A. Feoktystov, O. Petracic, N. Nandakumaran, A. Cervellino, and T. Brückel. “Signature of antiphase boundaries in iron oxide nanoparticles.” *J. Appl. Crystallogr.* **54**, 6 (2021).

[213] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by simulated annealing.” *Science* **220**, 4598 (1983), 671–680.

[214] V. Herynek, M. Babič, O. Kaman, H. Charvátová, M. Veselá, O. Buchholz, M. Vosmanská, D. Kubániová, J. Kohout, U. G. Hofmann, and L. Šefc. “Maghemite nanoparticles coated by methacrylamide-based polymer for magnetic particle imaging.” *J. Nanoparticle Res.* **23**, 2 (2021), 1–15.

[215] P. Scardi and M. Leoni. “Whole powder pattern modelling.” *Acta Crystallogr. A* **58**, 2 (2002), 190–200.

[216] A. Nunes and D. Lin. “Effects of surface relaxation on powder diffraction patterns of very fine particles.” *J. Appl. Crystallogr.* **28**, 3 (1995), 274–278.

[217] M. Leoni and P. Scardi. “Grain surface relaxation effects in powder diffraction.” In “Diffraction Analysis of the Microstructure of Materials,” Springer (2004), pages 413–454.

[218] I. Groma and A. Borbély. “X-ray peak broadening due to inhomogeneous dislocation distributions.” In “Diffraction Analysis of the Microstructure of Materials,” Springer (2004), pages 287–307.

[219] J.-D. Kamminga, L. Seijbel, and R. Delhez. “Determination of Non-uniform Dislocation Distributions in Polycrystalline Materials.” In “Diffraction Analysis of the Microstructure of Materials,” Springer (2004), pages 309–331.

[220] N. Popa. “The (hkl) dependence of diffraction-line broadening caused by strain and size for all Laue groups in Rietveld refinement.” *J. Appl. Crystallogr.* **31**, 2 (1998), 176–180.

[221] T. Ungár, I. Dragomir, Á. Révész, and A. Borbély. "The contrast factors of dislocations in cubic crystals: the dislocation model of strain anisotropy in practice." *J. Appl. Crystallogr.* **32**, 5 (1999), 992–1002.

[222] Y. V. Kolen'ko, M. Bañobre-López, C. Rodríguez-Abreu, E. Carbó-Argibay, A. Sailsman, Y. Piñeiro-Redondo, M. F. Cerqueira, D. Y. Petrovykh, K. Kovnir, O. I. Lebedev, et al. "Large-scale synthesis of colloidal Fe3O4 nanoparticles exhibiting high heating efficiency in magnetic hyperthermia." *J. Phys. Chem. C* **118**, 16 (2014), 8691–8701.

[223] A. Rečnik, I. Nyirő-Kósa, I. Dódony, and M. Pósfai. "Growth defects and epitaxy in Fe3O4 and γ-Fe2O3 nanocrystals." *CrystEngComm* **15**, 37 (2013), 7539–7547.

[224] M. H. Koch, P. Vachette, and D. I. Svergun. "Small-angle scattering: a view on the properties, structures and structural changes of biological macromolecules in solution." *Q. Rev. Biophys.* **36**, 2 (2003), 147.

[225] Sarveena, D. Muraca, P. M. Zélis, Y. Javed, N. Ahmad, J. M. Vargas, O. Moscoso-Londoño, M. Knobel, M. Singh, and S. Sharma. "Surface and interface interplay on the oxidizing temperature of iron oxide and Au–iron oxide core–shell nanoparticles." *RSC advances* **6**, 74 (2016), 70394–70404.

[226] M. Ghoshani, E. Sánchez, S. S. Lee, G. Singh, N. Yaacoub, D. Peddis, M. Mozaffari, C. Binns, J. De Toro, and P. Normile. "On the detection of surface spin freezing in iron oxide nanoparticles and its long-term evolution under ambient oxidation." *Nanotechnology* **32**, 6 (2020), 065704.

[227] M. Avdeev, D. Bica, L. Vekas, V. Aksenov, A. Feoktystov, O. Marinica, L. Rosta, V. Garamus, and R. Willumeit. "Comparative structure analysis of non-polar organic ferrofluids stabilized by saturated mono-carboxylic acids." *J. Colloid Interface Sci.* **334**, 1 (2009), 37–41.

[228] A. Bhadani, K. Iwabata, K. Sakai, S. Koura, H. Sakai, and M. Abe. "Sustainable oleic and stearic acid based biodegradable surfactants." *RSC advances* **7**, 17 (2017), 10433–10442.

[229] T. Maurer, F. Zighem, S. Gautrot, F. Ott, G. Chaboussant, L. Cagnon, and O. Fruchart. "Magnetic nanowires investigated by Polarized SANS." *Physics Procedia* **42** (2013), 74–79.

[230] A. Sundaresan, R. Bhargavi, N. Rangarajan, U. Siddesh, and C. Rao. "Ferromagnetism as a universal feature of nanoparticles of the otherwise nonmagnetic oxides." *Phys. Rev. B* **74**, 16 (2006), 161306.

[231] D. Fiorani, A. Testa, F. Lucari, F. D'orazio, and H. Romero. "Magnetic properties of maghemite nanoparticle systems: surface anisotropy and interparticle interaction effects." *Physica B Condens. Matter* **320**, 1-4 (2002), 122–126.

[232] J. K. Vassiliou, V. Mehrotra, M. W. Russell, E. P. Giannelis, R. McMichael, R. Shull, and R. F. Ziolo. "Magnetic and optical properties of γ-Fe2O3 nanocrystals." *J. Appl. Phys.* **73**, 10 (1993), 5109–5116.

[233] M. D. Kuz'min, M. Richter, and A. N. Yaresko. "Factors determining the shape of the temperature dependence of the spontaneous magnetization of a ferromagnet." *Phys. Rev. B* **73**, 10 (2006), 100401.

[234] A. Demortiere, P. Panissod, B. Pichon, G. Pourroy, D. Guillon, B. Donnio, and S. Begin-Colin. "Size-dependent properties of magnetic iron oxide nanocrystals." *Nanoscale* **3**, 1 (2011), 225–232.

[235] C. Bean and I. Jacobs. "Magnetic granulometry and super-paramagnetism." *J. Appl. Phys.* **27**, 12 (1956), 1448–1452.

[236] J. Yang, X. Zhou, W. B. Yelon, W. J. James, Q. Cai, K. Gopalakrishnan, S. K. Malik, X. Sun, and D. Nikles. "Magnetic and structural studies of the Verwey transition in $Fe_{3-\delta}O_4$ nanoparticles." *J. Appl. Phys.* **95**, 11 (2004), 7540–7542.

[237] F. Bødker, S. Mørup, and S. Linderoth. "Surface effects in metallic iron nanoparticles." *Phys. rev. lett.* **72**, 2 (1994), 282.

[238] X. Batlle, N. Pérez, P. Guardia, O. Iglesias, A. Labarta, F. Bartolomé, L. García, J. Bartolomé, A. Roca, M. Morales, et al. "Magnetic nanoparticles with bulklike properties." *J. Appl. Phys.* **109**, 7 (2011), 07B524.

[239] I. Hrianca, C. Caizer, and Z. Schlett. "Dynamic magnetic behavior of Fe 3 O 4 colloidal nanoparticles." *J. Appl. Phys* **92**, 4 (2002), 2125–2132.

[240] S. Disch. *The spin structure of magnetic nanoparticles and in magnetic nanostructures.* Ph.D. thesis (2010).

[241] W. Baaziz, B. P. Pichon, S. Fleutot, Y. Liu, C. Lefevre, J.-M. Greneche, M. Toumi, T. Mhiri, and S. Begin-Colin. "Magnetic iron oxide nanoparticles: reproducible tuning of the size and nanosized-dependent composition, defects, and spin canting." *J. Phys. Chem. C* **118**, 7 (2014), 3795–3810.

Acknowledgments

Without the help and advice of many people this work would not have been possible. I would like to express my sincere gratitude to:

Prof. Dr. Thomas Brückel for giving me the opportunity of working on this interesting topic in an excellent environment. For helpful discussions and invaluable support during the preparations of manuscripts and for reviewing my thesis. I would like to also thank him for giving me the freedom to follow my own ideas, while always asking the right questions to advance the research project.

Prof. Dr. Joachim Mayer for the second review of this thesis.

Dr. Artem Feoktystov for introducing me into the field of small-angle scattering, providing consistent and valuable support at every stage of the research project, for helping in the preparation, conduction and analysis of SANS experiments and for assistance and guidance in writing manuscripts, beamtime proposals and the thesis.

PD Dr. Oleg Petracic for introducing me to the field of magnetism and nanomagnetism, for providing help and guidance at the SQUID instrument, for help in writing and reviewing the Monte Carlo simulation program, for fruitful discussions and ideas that shaped this thesis, for his advise on scientific writing and for reviewing the thesis.

Dr. Emmanuel Kentzinger for the guidance and support during this research project, for valuable inputs and ideas, for help in preparation, conduction and analysis of SAXS measurements and for reviewing my thesis.

Dr. Sascha Ehlert for the provision of excellent iron oxide nanoparticle samples.

Tanvi Bhatnagar-Schöffmann for help in performing TEM/HRTEM measurements at the ER-C.

Prof. Dr. Rafal Dunin-Borkowski and **Dr. András Kovács** for agreeing to collaborate and giving me the opportunity to perform TEM/HRTEM measurements at the excellent instruments of the ER-C. I am also thankful for valuable insights and corrections during preparation of the manuscript published in Nanoscale.

Dr. Antonio Cervellino for support in performing XRD measurements at the beamline MS-X04SA of the SLS and in the preparation of manuscripts.

Dr. Leighanne Gallington and the rest of the beamline staff of 11-ID-B at the Argonne National Laboratory for performing the remote X-ray powder diffraction experiments.

Nileena Nandakumaran for support in preparing samples for several beamtimes, including those at SLS and FRM-II, for the contributions to the writing of manuscripts and for always being helpful.

Dr. Mikhail Feygenson for inviting me to join the beamtime at SLS, for valuable support in the data evaluation, especially regarding the PDF analysis and for helpful discussions, important insights and suggestions.

Prof. Dr. Heiko Wende and **Dr. Joachim Landers** for agreeing to collaborate, for providing help in performing and interpreting the Mössbauer experiments and for support in writing the manuscript published in Nanoscale.

Dr. Marie-Sousai Appavou for help in transmission electron microscopy experiments at JCNS.

Tabea Bartelt for help in the chemical laboratories and for support in the ordering and international shipping of chemicals and samples.

Dr. Ulrich Rücker for support at the GALAXI instrument.

Berthold Schmitz for technical support at the MPMS.

Lisa Fruhner for providing help in preparing samples for SAXS.

Michael Smik for introducing me into the institute in my first weeks and for help in performing and evaluating SQUID and SAXS experiments.

Dr. Volker Nischwitz for help in the elemental analysis at ZEA-3.

Franziska Michel, **Monika Krug** and **Barbara Daegener** for their help in administrative questions.

All my colleagues, especially **Benjamin Krugmann** and **Heiko Trepka** for pleasant discussions, coffee breaks and the great time working with them at the JCNS in Garching and during my trips to Jülich.

My Family and friends for believing in me and supporting me during all this time.

Eidesstattliche Erklärung

Ich, Tobias Köhler,

erkläre hiermit, dass diese Dissertation und die darin dargelegten Inhalte die eigenen sind und selbstständig, als Ergebnis der eigenen originären Forschung, generiert wurden.

Hiermit erkläre ich an Eides statt

1. Diese Arbeit wurde vollständig oder größtenteils in der Phase als Doktorand dieser Fakultät und Universität angefertigt;
2. Sofern irgendein Bestandteil dieser Dissertation zuvor für einen akademischen Abschluss oder eine andere Qualifikation an dieser oder einer anderen Institution verwendet wurde, wurde dies klar angezeigt;
3. Wenn immer andere eigene- oder Veröffentlichungen Dritter herangezogen wurden, wurden diese klar benannt;
4. Wenn aus anderen eigenen- oder Veröffentlichungen Dritter zitiert wurde, wurde stets die Quelle hierfür angegeben. Diese Dissertation ist vollständig meine eigene Arbeit, mit der Ausnahme solcher Zitate;
5. Alle wesentlichen Quellen von Unterstützung wurden benannt;
6. Wenn immer ein Teil dieser Dissertation auf der Zusammenarbeit mit anderen basiert, wurde von mir klar gekennzeichnet, was von anderen und was von mir selbst erarbeitet wurde;
7. Ein Teil dieser Arbeit wurde zuvor veröffentlicht und zwar in:

   T. Köhler, A. Feoktystov, O. Petracic, E. Kentzinger, T. Bhatnagar-Schöffmann, M. Feygenson, N. Nandakumaran, J. Landers, H. Wende, A. Cervellino, U. Rücker, A. Kovács, R. E. Dunin-Borkowski and T. Brückel. "Mechanism of magnetization reduction in iron oxide nanoparticles." *Nanoscale* **13**, 14 (2021), 6965-6976.

   T. Köhler, A. Feoktystov, O. Petracic, N. Nandakumaran, A. Cervellino, T. Brückel. "Signature of antiphase boundaries in iron oxide nanoparticles." *J. Appl. Cryst.* **54**, 6 (2021), 1719-1729.

Ort, Datum   Unterschrift

Appendix A

PDF analysis results

Tab. A.1 Results of the total PDF fit for sample OC05 as shown in fig. 4.27a). The tetragonal unit cell with space group symmetry $P4_32_12$ was used. Only a single isotropic displacement parameter was refined for all oxygen positions to reduce the number of parameters.

Position	mult.	x/a	y/a	z/c	occ.	U_{iso}
Fe1(tet.)	8	0.7587(5)	0.0021(5)	0.1208(5)	0.98(3)	0.0044(5)
Fe2(oct.)	4	0.3717(5)	0.6283(5)	0.75	0.75(3)	0.0085(5)
Fe3(oct.)	4	0.1283(5)	0.8714(5)	0.25	0.95(3)	0.0085(5)
Fe4(oct.)	8	0.3789(5)	0.8749(5)	-0.0102(5)	0.598(3)	0.0095(5)
O1	8	0.1275(5)	0.3725(5)	0.5090(5)	1	0.0090(5)
O2	8	0.3530(5)	0.1209(5)	-0.0181(5)	1	0.0090(5)
O3	8	0.1441(5)	0.8894(5)	0.0023(5)	1	0.0090(5)
O4	8	0.3907(5)	0.6061(5)	-0.0175(5)	1	0.0090(5)
a=b (Å)	8.339(2)					
c	8.411(2)					
δ (Å^{-1})	1.28(1)					
Particle size (Å)	52(5)					
R_w	0.172					

Tab. A.2 Results of the PDF fit for sample OC10 as shown in fig. 4.30a). The tetragonal unit cell with space group symmetry $P4_32_12$ was used. Only a single isotropic displacement parameter was refined for all oxygen positions to reduce the number of parameters.

Position	mult.	x/a	y/a	z/c	occ.	U_{iso}
Fe1(tet.)	8	0.7422(5)	0.9970(5)	0.1239(5)	0.80(3)	0.0057(5)
Fe2(oct.)	4	0.3597(5)	0.6403(5)	0.75	0.089(3)	0.0134(5)
Fe3(oct.)	4	0.1180(5)	0.8820(5)	0.25	0.87(3)	0.0109(5)
Fe4(oct.)	8	0.3711(5)	0.8747(5)	0.9880(5)	0.78(3)	0.0052(5)
O1	8	0.1332(5)	0.3940(5)	0.5008(5)	1	0.0144(5)
O2	8	0.3665(5)	0.1076(5)	-0.0017(5)	1	0.0144(5)
O3	8	0.1258(5)	0.8750(5)	-0.0008(5)	1	0.0144(5)
O4	8	0.3766(5)	0.6267(5)	0.0006(5)	1	0.0144(5)
a=b (Å)	8.356(2)					
c	8.427(2)					
δ (Å^{-1})	1.86(1)					
Particle size (Å)	99(5)					
R_w	0.190					

Tab. A.3 Results of the PDF fit for sample SE11 as shown in fig. 4.32a). The tetragonal unit cell with space group symmetry $P4_32_12$ was used. Only a single isotropic displacement parameter was refined for all oxygen positions to reduce the number of parameters. It should be noted that the particle size determined for the PDF is not very reliable as it compensates for imperfections in the fit resulting from the presence of APBs and other structural defects.

Position	mult.	x/a	y/a	z/c	occ.	U_{iso}
Fe1(tet.)	8	0.7448(5)	0.0038(5)	0.1276(5)	0.89(3)	0.0060(5)
Fe2(oct.)	4	0.3771(5)	0.6229(5)	0.75	1.00(3)	0.0083(5)
Fe3(oct.)	4	0.1247(5)	0.8753(5)	0.25	0.17(3)	0.0073(5)
Fe4(oct.)	8	0.3723(5)	0.8716(5)	0.0139(5)	0.87(3)	0.0063(5)
O1	8	0.1236(5)	0.3960(5)	0.5002(5)	1	0.0076(5)
O2	8	0.3692(5)	0.1107(5)	0.0044(5)	1	0.0076(5)
O3	8	0.1310(5)	0.0872(5)	0.0012(5)	1	0.0076(5)
O4	8	0.3759(5)	0.6335(5)	0.0028(5)	1	0.0076(5)
a=b (Å)	8.378(2)					
c	8.421(2)					
δ (Å^{-1})	1.49(1)					
Particle size (Å)	78(5)					
R_w	0.194					

Tab. A.4 Results of the PDF fit for sample OC15 as shown in fig. 4.36a). The tetragonal unit cell with space group symmetry $P4_32_12$ was used. Only a single isotropic displacement parameter was refined for all oxygen positions to reduce the number of parameters.

Position	mult.	x/a	y/a	z/c	occ.	U_{iso}
Fe1(tet.)	8	0.7438(5)	0.0005(5)	0.1242(5)	0.99(3)	0.0070(5)
Fe2(oct.)	4	0.3799(5)	0.6201(5)	0.75	0.86(3)	0.0086(5)
Fe3(oct.)	4	0.1234(5)	0.8766(5)	0.25	0.37(3)	0.0070(5)
Fe4(oct.)	8	0.3696(5)	0.8660(5)	0.9900(5)	0.99(3)	0.0068(5)
O1	8	0.1423(5)	0.3965(5)	0.4498(5)	1	0.0103(5)
O2	8	0.3713(5)	0.1274(5)	0.0037(5)	1	0.0103(5)
O3	8	0.1277(5)	0.8564(5)	0.0235(5)	1	0.0103(5)
O4	8	0.3749(5)	0.6303(5)	0.9950(5)	1	0.0103(5)
a=b (Å)	8.358(2)					
c	8.359(2)					
δ (Å^{-1})	1.78(1)					
Particle size (Å)	97(5)					
R_w	0.134					

Tab. A.5 Results of the PDF fit for sample OC20 as shown in fig. 4.39a). The tetragonal unit cell with space group symmetry $P4_32_12$ was used. Only a single isotropic displacement parameter was refined for all oxygen positions to reduce the number of parameters.

Position	mult.	x/a	y/a	z/c	occ.	U_{iso}
Fe1(tet.)	8	0.7457(5)	0.0013(5)	0.1263(5)	0.87(3)	0.0071(5)
Fe2(oct.)	4	0.3830(5)	0.6170(5)	0.75	0.67(3)	0.0047(5)
Fe3(oct.)	4	0.1215(5)	0.8786(5)	0.25	0.25(3)	0.0026(5)
Fe4(oct.)	8	0.3765(5)	0.8692(5)	0.0047(5)	0.93(3)	0.0080(5)
O1	8	0.1228(5)	0.3884(5)	0.5108(5)	1	0.0175(5)
O2	8	0.3740(5)	0.1108(5)	0.0044(5)	1	0.0175(5)
O3	8	0.1280(5)	0.8785(5)	0.0054(5)	1	0.0175(5)
O4	8	0.3781(5)	0.6283(5)	0.9991(5)	1	0.0175(5)
a=b (Å)	8.386(2)					
c	8.391(2)					
δ (Å^{-1})	1.39(1)					
Particle size (Å)	118(5)					
R_w	0.198					

Appendix B

Debye scattering equation simulation program

The source code for the Debye scattering equation simulation program as described in section 3.7.2 is shown here. The file *Example.py* contains an example on how to use this program to simulate powder patterns.

B.1 crystal.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Crystal building module

Generate a nanocrystal from input CIF file with vacancies,cube or sphere shape
and antiphase boundaries.

Classes:
    Atom
    Crystal
"""
import numpy as np
from itertools import product
from scipy.spatial.distance import pdist
import sys
import plotting
from pympler import asizeof

class Atom():
    """
    Class to store atom information.

    Attributes
    ----------
    coordinates : ndarray
        Fractional x,y,z coordinates of the atom.
    element : str
        String label of the atoms element.
    label : str
        Position label, e.g. 'Fe1'.
    probability : float
        Selection probability if a gradient is applied for vacancy generation.
    isAPB : int
        set to 1 if the atom is involved in the modified exchange interactions.
    ucn : int
        unit cell number the atom belongs to.

    """
    __slots__ = {"coordinates", "element", "label", "probability", "isAPB", "ucn"
        }

    def __init__(self, coordinates=(0.0,0.0,0.0), element="Fe",
                 label=None,isAPB=0, ucn=0):
        self.coordinates = np.array(coordinates)
        self.element = element
        self.label = label
        self.probability = 0
        self.isAPB = isAPB
        self.ucn = ucn

class Crystal():
    """
    Class to store crystal information and perform calculations to build the
    nanocrystal.
    """
    def __init__(self, diameter, unitcell,occupancies, shape, n_APBs, offset,
        output_dir):
        """
        The unit cell contents are provided as parameters, and the crystal is
            built up
        by repeating the unit cell in all dimensions n times, where n is
        specified by the parameter diameter. In the repitiion loop Atom objects
        are initialized and stored in self.Atoms.

        Parameters
        ----------
        diameter : Int
            Number of unit cell repititions in x,y and z.
        unitcell : unitcell object
            Object containing the unitcell information obtained from CifParser.
        shape : str
            If "Sphere" cut the crystal in spherical shape with diameter
            specified in crystal initialization.
            If "Cube" cut into cube shape with same volume as a sphere of
                diameter
            specified in crystal initialization.
        """
        self.diameter = diameter
        valid_shapes = ["Sphere", "Cube"]
        if shape not in valid_shapes:
            raise ValueError("The input shape is not valid.")
        else:
            self.shape = shape
        self.radius = diameter/2.0
        self.offset = offset
        self.atoms = list()
        self.unitcell = unitcell.positions
        self.elements = unitcell.elements
        self.labels = unitcell.labels
        self.n_APBs = n_APBs
        self.cif_filename = ((unitcell.filename).split('/'))[-1]
        self.atom_numbers = {}
        self.lattice_a = unitcell.lattice_a
        self.lattice_b = unitcell.lattice_b
        self.lattice_c = unitcell.lattice_c
        self.occupancies = occupancies
        self.formfactor_array = []
        self.a_numbers = 0
        self.output_dir = output_dir

        for i in self.elements:
            self.atom_numbers[i] = 0

        for a,e,l in zip(self.unitcell, self.elements, self.labels):
            for f in product(range(self.diameter+1),
                             range(self.diameter+1),
                             range(self.diameter+1)):
                self.atoms.append(Atom(coordinates= a + np.array(f),
                                       element = e, label = l, ucn = f))
                self.atom_numbers[e] += 1

        self.atoms = np.array(self.atoms)
        print("\nCrystal initialized")
        for key in self.atom_numbers.keys():
                print("\t %s: %d"%(key,self.atom_numbers[key]))

        def gaussian(self,x,sig):
            """ Gaussian distribution """
            n = 1/(np.sqrt(2*np.pi)*sig)
            e = np.exp(-(x/sig)**2 / 2)
            return n*e

        def lorentzian(self,x,gamma):
            """ Lorentzian distribution """
            return (1/np.pi)*(gamma/2)/((x**2)+(gamma/2)**2)

        def gradient(self,gradient_sig, plot=False):
            """
            Setup a probability gradient for vacancy formation.

            The Gradient is used to calculate a selection probability for every
            atom in the structure.

            Parameters
            ----------
            gradient_sig : float
                Sigma parameter for distribution functions.
            plot : Bool, optional
                Plot the probabilities against the distance from particle center.
                The default is False.
            """
            ps = []
            ds = []
            xp = np.arange(0,self.diameter,0.01)
            #g = self.gaussian(xp,gradient_sig)
            g = self.lorentzian(xp,gradient_sig)
            fp = (-1*g)+g.max()
            for atom in self.atoms:
                atom.coordinates -= self.radius
                d = np.linalg.norm(atom.coordinates)
                ds.append(d)
                p = np.interp(d, xp,fp)
                ps.append(p)
                atom.probability = p
                atom.coordinates += self.radius
            if plot:
                data = {}
                data["PDF"] = (xp,fp,1,'')
                data["Probabilites for atoms"] = (ds,ps,0,'.')
                plotting.plot(data,"Radius [unit cells]", "Probability")

        def generate_vacancies(self, occ,gradient=None, SEED=0):
            """
            Generate vacancies on iron sites.

            This is only intended for maghemite or magnetite structures.

            First the list of atom labels is generated from the list of atoms. Then
            the unique labels and their occurrances are determined with np.unique().
            Dictionaries are set up for the atom labels, the indices and the
                vacancies.
            After that the keys relating to oxygen in the structure are stored in
            O_keys. Now in an iteration over all atoms the atom indices corresponding
            to the labels determined previously are stored in indices_dict. The keys
            relating to oxygen are removed. A check is performed if the input keys
            match the ones determined from the crystal. If not the program is
            terminated. Random indices are drawn from the indices lists corresponding
            to the iron positions. The number of atoms to be removed is determined
            from the occupancy factor given in the input dictionary. The retrieved
            indices are stored all together in vacancies_merged, which is finally
            used to remove the selected atoms from the crystal.

        Parameters
        ----------
        occ : Dictionary
            Dictionary containing the site labels with occupancies.
        gradient : Bool
            Use a gradient to determine the probability of selection.
        SEED : Int
            Seed value for numpy random.
        """
        np.random.seed(SEED)

        if gradient != None:
            self.gradient(gradient_sig=gradient,plot = True)

        label_dict = {}
        indices_dict = {}
        vacancies_dict = {}

        label_list = np.array([atom.label for atom in self.atoms])
        uniq, counts = np.unique(label_list, return_counts=True)

        for n,l in enumerate(uniq):
            label_dict[l] = counts[n]
            indices_dict[l] = []
            vacancies_dict[l] = []

        O_keys = []
        for i in indices_dict.keys():
            if "O" in i:
                O_keys.append(i)

        for c,a in enumerate(self.atoms):
            for n in range(len(uniq)):
                if a.label == uniq[n]:
                    indices_dict[a.label].append(c)

        for i in O_keys:
            indices_dict.pop(i)
            vacancies_dict.pop(i)

        for i in occ.keys():
            if i not in vacancies_dict.keys():
                print("Occupancy dict contains wrong labels!")
                print("Valid keys are: ", vacancies_dict.keys())
                print("Vacancies were not set up.")
                return 1

        for i in indices_dict.keys():
            if gradient:
                p = []
                for a in indices_dict[i]:
                    p.append(self.atoms[a].probability)
                p=np.array(p)
                p/=p.sum()
                vacancies_dict[i] = np.random.choice(indices_dict[i],
                                                     int((1-occ[i])*
                                                     len(indices_dict[i])),
                                                     replace=False, p=p)
            else:
                vacancies_dict[i] = np.random.choice(indices_dict[i],
                                                     int((1-occ[i])*
                                                     len(indices_dict[i])),
                                                     replace=False)
        vacancies = []
        vacancies_merged = []
        print("   generating vacancies...")
        for i in vacancies_dict.keys():
            vacancies.append(vacancies_dict[i])
```

```
                occ_calc = (1-(len(vacancies_dict[i])/len(indices_dict[i])))
                print("\t\t%s: %d/%d -> occ: %.2f" %(i,len(vacancies_dict[i]),
                        len(indices_dict[i]),occ_calc))

        for i in vacancies:
            vacancies_merged += i.tolist()

        self.atoms = np.delete(self.atoms, vacancies_merged)

        return 0

    def generate_APB(self,APB, n):
        """
        Generate antiphase boundary (APB).

        Generate an antiphase boundary through the center of the particle. First
        set the particle into the origin, then for all atoms on one side of
        the space diagonal, i.e. atoms whose x coordinate is larger than the y
        coordinate, get shifted along the APB by one quarter of a unit cell.
        Finally the particle is shifted back to the original position. With n=2
        two APBs are produced
        """
        if n == 2:
            for atom in self.atoms:
                atom.coordinates -= self.radius
                if (atom.coordinates[0]- atom.coordinates[1] > (-np.sqrt(2))):
                    atom.coordinates += np.array([0.25, 0.25, 0.0])
                if (atom.coordinates[0]- atom.coordinates[1] > (np.sqrt(2))):
                    atom.coordinates += np.array([0.25, 0.25, 0.0])
                atom.coordinates += self.radius

        if n == 1:
            for atom in self.atoms:
                atom.coordinates -= self.radius

                if (atom.coordinates[0]-atom.coordinates[1]) > 0.0:
                    atom.coordinates += np.array([0.25, 0.25, 0.0])
                if (((atom.coordinates[0]-atom.coordinates[1] < 0.0) and
                     (atom.coordinates[0]-atom.coordinates[1] > -0.4))
                    or ((atom.coordinates[0]-atom.coordinates[1] < 0.4) and
                        (atom.coordinates[0]-atom.coordinates[1] > 0.0))):
                    atom.isAPB = 1
                atom.coordinates += self.radius

    def cut_sphere(self, diameter,offset):
        """
        Generate sphere shape.

        Cut the particle into a spherical shape by appending only atoms with
        x^2 + y^2 + z^2 < r^2 to the new structure.

        Parameters
        ----------
        diameter : Float
            Diameter of the nanocrystal after shaping in fractions of unit cells.
        """

        radius = diameter/2.0 * self.lattice_a
        cut_crystal = []
        for atom in self.atoms:
            atom.coordinates *= np.array([self.lattice_a, self.lattice_b, self.
                lattice_c])
            atom.coordinates -= radius
            atom.coordinates -= offset
            if (np.dot(atom.coordinates, atom.coordinates) < radius**2):
                atom.coordinates /= np.array([self.lattice_a, self.lattice_b,
                    self.lattice_c])
                atom.coordinates += radius
                atom.coordinates += offset
                cut_crystal.append(atom)
        self.atoms = cut_crystal

    def cut_cube(self, edgelength):
        """
        Generate cube shape.

        Remove atoms from initialized crystal to generate a cube shaped crystal
        with specified edgelength.

        Parameters
        ----------
        edgelength : Float
            Edge length of the nanocrystal after shaping in fractions of unit
            cells.
        """
        cut_crystal = []
        for atom in self.atoms:
            atom.coordinates -= (edgelength/2)
            if (abs(atom.coordinates[0]) <= (edgelength/2)
                and abs(atom.coordinates[1]) <= (edgelength/2)
                and abs(atom.coordinates[2]) <= (edgelength/2)):
                atom.coordinates += (edgelength/2)
                cut_crystal.append(atom)
        self.atoms = cut_crystal

    def return_coordinate_list(self):
        """
        Return coordinates and elements.

        Return the coordinates of all atoms as list to be used in
        other methods, e.g. calculate_distance_array,
        calculate_formfactor_matrix and 3D plotting. Elements contains the
        strings designating the elements in the same order as the atomic
        coordinates.

        Returns
        -------
        coordinates : List of lists of floats
            list containing the x,y,z coordinates for every atom.
        elements_all_atoms : List of strings
            element symbols.

        """
        coordinates = []
        elements_all_atoms = []
        for atom in self.atoms:
            coordinates.append(atom.coordinates.tolist())
            elements_all_atoms.append(atom.element)

        return coordinates,elements_all_atoms

    def calculate_distance_array(self):
        """
        Calculate the distances for every atom pair.

        Calculate the pair distances r_ij in the crystal structure in unitcells
        using the scipy.spatial.distance method pdist that takes a list of
        coordinates as argument and returns the upper triangle of the euclidian
        pair distance matrix as a flattened array.
        """
        coordinates,elements = self.return_coordinate_list()
        #coordinates *= np.array([self.lattice_a, self.lattice_b, self.lattice_c
            ])
        coords = coordinates * np.array([self.lattice_a, self.lattice_b, self.
            lattice_c])
        self.distance_array = pdist(coords)
```

```python

    def calculate_formfactor_matrix(self):
        """
        Calculate the formfactor products for each atom pair.

        First the element symbols in the same order as the atomic coordinates
        are retrieved from the return_coordinate_list method.

        Depending on the type of element numbers 2 or 3 are stored in ff_nums.
        The pariwise matrix is constructed directly as a
        flattened array via the double for loop, where values of 4, 5 and 6
            corrspond
        to Fe-Fe, Fe-O and O-O. The diagonal elements are not
        present in this array due to the "+1" in the list argument. They are
        calculated directly in the calculate_Dse_total method of the dse.py
            module
        from the number of atoms in the structure.
        """
        coordinates,formfactors = self.return_coordinate_list()
        ff_nums = []
        keys= list(self.atom_numbers.keys())

        for i in formfactors:
            if i == keys[0]:
                ff_nums.append(2)
            else:
                ff_nums.append(3)

        formfactor_matrix = []
        for n,i in enumerate(ff_nums):
           for k in ff_nums[n+1:]:
               formfactor_matrix.append(i+k)

        self.formfactor_array  = np.array(formfactor_matrix)

    def build_nanoparticle(self, APB, occ,gradient, plot,SEED):
        """
        Setup the nanoparticle shape and calculate all parameters.

        Parameters
        ----------
        APB : Bool
            If True generate an APB along [110] through the particle center.
        occ : dictionary
            Set the occupancies iron positions.
        gradient : Float
            Set sigma parameter of gaussian distribution for probabilities.
        plot : Bool
            If True the crystal is plotted in 3D
        """
        print("\nBuilding Nanoparticle...")
        if occ:
            res = self.generate_vacancies(self.occupancies,gradient, SEED=SEED)
            if res == 1:
                sys.exit("terminating program")
            elif res == 0:
                print("   - vacancies generated")

        if APB:
            self.generate_APB(APB,self.n_APBs)
            print("   - APB generated")

        if self.shape == "Sphere":
            self.cut_sphere(self.diameter,self.offset)
            volume = 4/3*np.pi*(self.diameter/2)**3
            self.diameter = self.diameter*self.lattice_a/10
            print("   - sphere shape generated")
            print("\t\tdiameter: %.1f nm \n\t\tvolume: %.2f" %(self.diameter,
                volume))
        elif self.shape == "Cube":
            edgelength = (4/3*np.pi*(self.diameter/2)**3)**(1/3)
            self.diameter = edgelength*self.lattice_a/10
            volume = edgelength**3
            self.cut_cube(edgelength)
            print("    - cube shape generated, edge length: %.1f unit cells,
                volume: %.2f"
                  %(self.diameter, volume))

        self.calculate_formfactor_matrix()
        print("    - formfactor matrix generated")

        self.calculate_distance_array()
        print("    - pair distances calculated")

        print("Nanoparticle generated.")
        atom_el_numbers = {}

        for i in self.elements:
            atom_el_numbers[i] = 0
        for i in self.atoms:
            self.a_numbers += 1
            atom_el_numbers[i.element] += 1

        self.atom_numbers = atom_el_numbers

        print(atom_el_numbers)

        if plot:
            a,b = self.return_coordinate_list()
            plotting.plot_3D(coordinates=a ,element_list=b)

    def get_properties(self):
        """
        Return the calculated crystal properties for further use.

        Returns
        -------
        numpy.ndarray
            Array containing pair wise distances.
        numpy.ndarray
            Array containing pair wise form factor string-placeholders.
        numpy.chararray
            Array containing pair wise form factor string-placeholders of the
            diagonal elements, i.e. the entries with equal indices.
        """
        return self.distance_array, self.formfactor_array

    def output_crystal_structure(self, filename, oxygen):
        Fe_atoms = [atom for atom in self.atoms if atom.element == "Fe"]
        O_atoms = [atom for atom in self.atoms if atom.element == "O"]

        x = [atom.coordinates[0] for atom in Fe_atoms]
        y = [atom.coordinates[1] for atom in Fe_atoms]
        z = [atom.coordinates[2] for atom in Fe_atoms]

        position = [1 if (atom.label == "Fe(tet)" or atom.label == "Fe1")
                    else 0 for atom in Fe_atoms]

        isAPB = [atom.isAPB for atom in Fe_atoms]
        unit_cell = [atom.ucn for atom in Fe_atoms]

        if oxygen:
            for i in O_atoms:
                x.append(i.coordinates[0])
                y.append(i.coordinates[1])
                z.append(i.coordinates[2])
                position.append(2)
                isAPB.append(0)
                unit_cell.append(i.ucn)

        data = np.column_stack((x,y,z,position,isAPB,unit_cell))
        np.savetxt(slef.output_dir+filename, data, fmt='%.5f')

    def rotate(self,alpha,beta,gamma):
        alpha = np.pi/180.0
        beta = np.pi/180.0
        gamma = np.pi/180.0

        rotation_x = np.array([[1.0,0.0,0.0],
                               [0.0, np.cos(alpha), -np.sin(alpha)],
                               [0.0, np.sin(alpha), np.cos(alpha)]])

        rotation_y = np.array([[np.cos(beta), 0.0, np.sin(beta)],
                               [0.0, 1.0, 0.0],
                               [-np.sin(beta), 0.0, np.cos(beta)]])

        rotation_z = np.array([[np.cos(gamma),-np.sin(gamma),0.0],
                               [np.sin(gamma), np.cos(gamma), 0.0],
                               [0.0, 0.0,1.0]])

        for atom in self.atoms:
            atom.coordinates -= self.radius
            atom.coordinates = rotation_x.dot(atom.coordinates)
            atom.coordinates = rotation_y.dot(atom.coordinates)
            atom.coordinates = rotation_z.dot(atom.coordinates)
            atom.coordinates += self.radius
```

B.2 parsers.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

class CifParser():
    """
    Class to retrieve information from input CIF files. Currently only
    VESTA Cifs are supported.
    """
    def __init__(self, filename):
        self.filename = filename
        file = open(filename, "r")
        lines = file.readlines()
        space = []
        symmetry_elements = []
        unitcell = []
        coordinatelist = []
        expanded = []
        elementlist = []
        expanded_elements = []
        labels = []
        expanded_labels = []
        self.lattice_a = 0.0
        self.lattice_b = 0.0
        self.lattice_c = 0.0

        for n,line in enumerate(lines):
            if "cell_length_a" in line:
                self.lattice_a = float(line.split()[1])
            if "cell_length_b" in line:
                self.lattice_b = float(line.split()[1])
            if "cell_length_c" in line:
                    self.lattice_c = float(line.split()[1])
                if "space_group_name" in line:
                    self.spcgr = line.split()[1]
                if line == "\n":
                    space.append(n)

        for line in lines[space[2]+3:space[3]]:
            l = line.strip()
            l = l[1:-1]
            symmetry_elements.append(l)

        for line in lines[space[3]+10:]:
            unitcell.append(line.split())

        for i in unitcell:
            coordinatelist.append([float(i[2]),float(i[3]),float(i[4])])
            elementlist.append(i[-1])
            labels.append(i[0])

        for k,i in enumerate(coordinatelist):
            for n in symmetry_elements:
                x = i[0]
                y = i[1]
                z = i[2]

                # calculate symmetry equivalent positions
                symm = n.split(',')
                new_x = eval(symm[0])
                new_y = eval(symm[1])
                new_z = eval(symm[2])

                if new_x < 0:
                    new_x = 1+new_x
                if new_y < 0:
                    new_y = 1+new_y
                if new_z < 0:
                    new_z = 1+new_z

                if new_x == -0.0:
                    new_x = 0.0
                if new_y == -0.0:
                    new_y = 0.0
                if new_z == -0.0:
                    new_z = 0.0

                if new_x >= 1.0:
                    new_x -= 1.0
                if new_y >= 1.0:
                    new_y -= 1.0
                if new_z >= 1.0:
                    new_z -= 1.0

                new_x = round(new_x,4)
                new_y = round(new_y,4)
                new_z = round(new_z,4)

                expanded.append([new_x, new_y, new_z])
                expanded_elements.append(elementlist[k])
                expanded_labels.append(labels[k])

        self.positions = []
        self.elements = []
        self.labels = []
        for n,i in enumerate(expanded):
            if i not in self.positions:
                self.positions.append(i)
                self.elements.append(expanded_elements[n])
                self.labels.append(expanded_labels[n])

        for n,i in enumerate(self.positions):
            for k,l in enumerate(self.positions):
                if i == l and n != k:
                    print("duplicate at %d, %d"% (n,k))
                    print(i, l)

        print("\nCif import succesful\n")
```

B.3 debye.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import crystal
import dse
import plotting
import parsers
import sys
from pympler import asizeof

class Experiment():
    def __init__(self,wavelength,
                 x_step,
                 x_min,
                 x_max,
                 partitions,
                 diameter,
                 cif_file,
                 filename,
                 n_APBs = 1,
                 offset = 0.0,
                 x_par = "Theta",
                 occupancies = {},
                 shape="Sphere",
                 APB=False,
                 gradient=None,
                 plot_crystal=False,
                 plot_results=None,
                 output_path = "output/",
                 plot_2D_crystal = False,
                 dry_run = False,
                 calculate = True,
                 saveStructure = False,
                 oxygen=False,
                 SEED = 0,
                 selfscattering = True):

        if dry_run:
            diameter = 1
            partitions = 2

        if occupancies == {}:
            occ = False
        else:
            occ = True

        print("===============================")
        print("Experiment started")
        print("===============================")

        unitcell = parsers.CifParser(cif_file)

        Crystal = crystal.Crystal(diameter, unitcell, occupancies, shape, n_APBs,
                                  offset, output_path)

        Crystal.build_nanoparticle(APB = APB,
                                   occ = occ,
                                   gradient = gradient,
                                   plot = plot_crystal,
                                   SEED = SEED)

        if saveStructure:
            Crystal.output_crystal_structure(saveStructure, oxygen=oxygen)

        if plot_2D_crystal:
            plotting.plot_2D(Crystal)

        if calculate == True:
            DSE_calculator = dse.DseCalculator(Crystal,x_par, x_min, x_max,
                                               x_step, wavelength,
                                               partitions, output_path,
                                                   selfscattering)
            DSE_calculator.perform_calculation(filename = filename, plot =
                plot_results)

        print("================================")
        print("Experiment finished")
        print("================================")
```

B.4 dse.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Debye scattering equation (Dse) module

Generate atomic formfactors and calculate the Debye scattering equation for
nanocrystals.

Classes:
    AtomicFormfactorCalculator
    DseCalculator
"""
import numpy as np
import matplotlib.pyplot as plt
from collections import defaultdict
import numexpr as ne
import plotting

#Cromer-Mann coefficients for atomic formfactor from
# https://www.classe.cornell.edu/~dms79/x-rays/f0_CromerMann.txt

Fe3p_coeff = [11.17640,  7.386300 ,  3.394800 ,  7.2400004E-02 ,
             4.614700 ,  0.3005000 ,  11.67290 , 38.55660,
              0.9707000]
Fe2p_coeff = [11.04240,  7.374000 ,  4.134600 ,  0.4399000,
             4.653800 ,  0.3053000  , 12.05460 , 31.28090,
              1.009700]
Fe_coeff = [11.76950 ,  7.357300 ,  3.522200 ,  2.304500 ,
             4.761100  , 0.3072000 ,  15.35350 , 76.88050,
             1.036900]

Ce4p_coeff = [20.32350, 19.81860, 12.12330, 0.1445830,
             2.659410, 0.2188500, 15.79920, 62.23550,
             1.591800]

O2m_coeff = [4.758000,  3.637000,   0.000000,   0.000000,
             7.831000,   30.05000,   0.000000,  0.000000,
             1.594000]
```

```

F1m_coeff = [3.632200 ,  3.510570,   1.260640 ,  0.9407060 ,
             5.277560 ,  14.73530 ,  0.4422580 , 47.34370,
              0.6533960]

Ca2p_coeff = [15.63480  , 7.951800 ,  8.437200  , 0.8537000 ,
                -7.4000000E-03 ,  0.6089000  , 10.31160 , 25.99050,
                -14.87500]

class AtomicFormfactorCalculator():
    """
    Class to generate formfactors.

    Attributes
    ----------
    formfactors : defaultdict(list)
        formfactors stored in a dictionary, can be retrieved with the element key
            .
    theta : ndarray
        Theta angles in degrees between min and max values and step size
            specified
        in input.
    theta_rad : ndarray
        Theta angles in rad.
    q : ndarray
        Scattering vector q defined as 4pi*sin(theta)/lambda.
    q_4pi : ndarray
        Scattering vector q defined as sin(theta)/lambda.
    elements : list(str)
        List of strings representing the elements/ions to be calculated.

    Methods
    -------
    formfactor(q,element):
        Calculate the formfactor in range q of element.
    get_formfactor(element):
        Return formfactor array of element.
    get_theta():
        Return theta array.
    get_q():
        Return q array.
    plot_formfactors():
        Plot all formfactors in the range they were calculated.
    """
    def __init__(self,x_par, x_min, x_max, x_step, wavelength, elements):
        """
        Initialize the AtomicFormfactorCalculator with a specified q or Theta
        range.

        Parameters
        ----------
        x_par : str
            String label to set which x-values to use (2Theta or Q).
        x_min : float
            Lower limit for x.
        x_max : float
            Upper limit for x.
        x_step : float
            Step size of x..
        wavelength : float
            X-ray wavelength in Angstrom.
        elements : list of str
            List containing str labels of the elements to be calculated.
        """
        self.formfactors = defaultdict(list)
        if x_par == "Theta":
            self.theta = np.arange(x_min, x_max, x_step)
            self.theta_rad = np.arange(x_min, x_max, x_step)*np.pi/180
            self.q = 4*np.pi*np.sin(self.theta*np.pi/180)/wavelength
            self.q_4pi = np.sin(self.theta*np.pi/180)/wavelength
        elif x_par == "Q":
            self.q = np.arange(x_min, x_max, x_step)
            self.q_4pi = self.q/(4*np.pi)
            self.theta_rad = np.arcsin(self.q*wavelength/(4*np.pi))
            self.theta = self.theta_rad * 180/np.pi

        # set up form factors for all elements present in structure
        self.elements = elements
        for i in self.elements:
            self.formfactors[i].append(self.formfactor(self.q_4pi, i))

    def formfactor(self, q, element):
        """
        Calculate the formfactor in range q of element.

        The non-dispersive part of the atomic scattering factor is calculated
        in the specified q range according to
        f(k) = c+[SUM a_i*EXP(-b_i*(k^2))]; i=1,4
        with k = sin(theta)/lambda and c,a_i and b_i the Cromer-Mann coefficients
        given in the International Tables of Crystallography vol. 4 or vol C
        (pg 500-502) and tabulated in
        https://www.classe.cornell.edu/~dms79/x-rays/f0_CromerMann.txt. This
        parametrization is only good up to sin(theta)/lambda < 2.0 [Angstrom^-1].
        For lambda = 0.21148 this means an upper limit of theta of 25 [  ]. The
        correct parameters are chosen according to the element/ion name provided
        in the function argument.

        Parameters
        ----------
        q : ndarray
            Q array used for the calculation.
        element : str
            Element/Ion to be calculated.

        Returns
        -------
        ndarray
            Formfactor of element in the specified q range.
        """
        coeff = None
        if element == "Fe3+":
            coeff = Fe3p_coeff

        elif element == "Fe2+":
            coeff = Fe2p_coeff

        elif element =="O2-":
            coeff = O2m_coeff

        elif element =="Ce4+":
            coeff = Ce4p_coeff

        elif element =="F1-":
            coeff = F1m_coeff

        elif element =="Ca2+":
            coeff = Ca2p_coeff

        f = (coeff[0]*np.exp(-coeff[4]*q**2) +
             coeff[1]*np.exp(-coeff[5]*q**2) +
             coeff[2]*np.exp(-coeff[6]*q**2) +
             coeff[3]*np.exp(-coeff[7]*q**2) + coeff[8])
        return np.array(f)

    def get_formfactor(self, element):
        return np.array(self.formfactors[element][0])
```

```python

    def get_theta(self):
        return self.theta

    def get_q(self):
        return self.q

    def plot_formfactors(self):
        fig,ax = plt.subplots(1,1)

        for i in self.elements:
            ax.plot(self.theta*2, self.get_formfactor(i), label=i)

        ax.legend()
        ax.set_xlim(left=0)
        ax.set_xlabel("$2\Theta (  )$")
        ax.set_ylabel("f (electrons)")
        plt.show()

class DseCalculator():
    """
    Class to calculate the Debye scattering equation.

    Attributes
    ----------
    distances : ndarray
        Array containing the pair wise distances.
    f_ij : chararray
        Array containing the numbers corresponding to atomic scattering factor
        products in the same order as the distances.
    latticepar : float
        Lattice parameter a of the unit cell.
    atom_numbers : dictionary
        Dictionary containing the total number of atoms grouped by elements.
    t_min : float
        Lower limit of the theta range.
    t_max : float
        Upper limit of the theta range.
    step : float
        Theta step.
    wavelength : float
        X-ray wavelength in Angstrom.
    num_partitions : int
        Number of partitions to be performed prior to calculation.
    intensity : ndarray
        Array containing the I(theta).
    selfscattering: bool
        Decide if the self scattering contribution should be calculated.

    Methods
    -------
    partition_crystal():
        Divide the crystal into parts to keep memory usage within RAM limits.
    calculate_Dse(distances, f_ij):
        Calculate the Debye scattering equation for non-equal pairs of atoms.
    calculate_Dse_total():
        Calculate the scattered intensity from all parts, including the self
        correlations.
    perform_calculation(filename=None, plot=True):
        Wrapper function to ensure results are saved and plotted or intentionally
        discarded.
    save_to_text(filename):
        Utility function to save q,theta and I to text file.

    """

    def __init__(self, crystal, x_par, x_min, x_max,x_step, wavelength,
                 num_partitions, output_path, selfscattering):
```

```
        """
        Initialize the DseCalculator.

        Parameters
        ----------
        crystal : Crystal object
            Crystal object obtained from crystal module.
        x_par : str
            Parameter to specify the x-values to be used (Theta or Q).
        x_min : float
            Lower limit of x range.
        x_max : TYPE
            Upper limit of x range.
        x_step : float
            Step size of x.
        wavelength : float
            X-ray wavelength in Angstrom.
        num_partitions : int
            Number of partitions to be performed prior to calculation.
        output_path: str
            Path for output files, only existing directories are accepted.
        """
        self.cif_filename = crystal.cif_filename
        self.occupancies = crystal.occupancies
        self.diameter = crystal.diameter
        self.shape = crystal.shape
        self.distances, self.f_ij = crystal.get_properties()
        self.latticepar = crystal.lattice_a
        self.atom_numbers = crystal.atom_numbers
        self.x_par = x_par
        self.x_min = x_min
        self.x_max = x_max
        self.x_step = x_step
        self.wavelength = wavelength
        self.num_partitions = num_partitions
        self.intensity = 0
        self.output_path = output_path
        self.num_Atoms = crystal.a_numbers
        self.selfscattering = selfscattering

        elements = ["O2-", "Fe3+", "Fe2+", "Ce4+", "Ca2+", "F1-"]

        formfactors = AtomicFormfactorCalculator(self.x_par,self.x_min,
                                                 self.x_max,
                                                 self.x_step,
                                                 self.wavelength,
                                                 elements)
        #formfactors.plot_formfactors()
        self.q = formfactors.get_q()
        self.theta = formfactors.get_theta()

        global CaF, FCa, CaCa, FF, CeO, OCe, CeCe, FeFe, FeO, OFe, OO

        CaF = formfactors.get_formfactor("Ca2+") * formfactors.get_formfactor("F1
            -")
        FCa = CaF
        CaCa = formfactors.get_formfactor("Ca2+") * formfactors.get_formfactor("
            Ca2+")
        FF = formfactors.get_formfactor("F1-") * formfactors.get_formfactor("F1-"
            )
        CeO = formfactors.get_formfactor("Ce4+") * formfactors.get_formfactor("O2
            -")
        OCe = CeO
        CeCe = formfactors.get_formfactor("Ce4+") * formfactors.get_formfactor("
            Ce4+")
        FeFe = formfactors.get_formfactor("Fe3+") * formfactors.get_formfactor("
            Fe3+")
        FeO = formfactors.get_formfactor("Fe3+") * formfactors.get_formfactor("O2-")
        OFe = FeO
        OO = formfactors.get_formfactor("O2-") * formfactors.get_formfactor("O2-")

        print("\nDSE calculator initialized")
        print("\t 2\u03B8-range: %.2f-%.2f" % (self.theta.min()*2, self.theta.max()*2))
        print("\t Q-range: %.2f-%.2f" % (self.q.min(), self.q.max()))
        print("\t wavelength: %.5f \u212B" % (self.wavelength))

    def partition_crystal(self):
        """
        Slice the crystal.

        Divides the distance and f_ij arrays into parts. The number of the
        partitions is specified during initialization of the DseCalculator
        with the num_partitions parameter.

        Returns
        -------
        sliced_crystal : ndarray
            Array containing the distances in num_partitions separate arrays.
        sliced_formfactors : chararray
            Array containing the str labels for the formfactor products in
            num_partitions separate chararrays.
        """
        num_distances = len(self.distances)
        step_size = int(num_distances/self.num_partitions)
        sliced_crystal = []
        sliced_formfactors = []

        for i in range(self.num_partitions):
            if i == self.num_partitions-1:
                sliced_crystal.append(self.distances[step_size*i:])
                sliced_formfactors.append(self.f_ij[step_size*i:])
            else:
                sliced_crystal.append(self.distances[step_size*i:step_size*(i+1)])
                sliced_formfactors.append(self.f_ij[step_size*i:step_size*(i+1)])
        return sliced_crystal, sliced_formfactors

    def calculate_Dse(self, distances, f_ij):
        """
        Calculate the Debye scattering equation for non-equal indices.

        The str labels of the formfactor products are evaluated with the built-in
        function eval(). f_ij_e now contains f(q) arrayis for every pair of indices.
        f_ij_e has shape (num_pairs, q_steps).
        Distances in angstrom are calculated by multiplying the distances
        obtained previously with the lattice parameter. This distance array is
        reshaped into a column vector with the numpy method reshape() in order
        to allow broadcasting with f_ij_e array and q array. q*rij_reshaped
        results in an array with shape (num_pairs, q_steps).
        The Debye scattering equation is evaluated for the upper triangle of indices
        using the numexpr module, afterwards all I_ij(q) arrays are summed up
        to obtain I(q). Numexpr is a fast numerical expression evaluator for NumPy
        developed by David M. Cooke et al., making use of all cores to do as many
        calculations in parallel as possible. Additionally better performance than
        NumPy is achieved by avoiding allocation of memory for intermediate results
        and making use of a virtual machine written in C.
        For more information see the documentation
        http://numexpr.readthedocs.io/en/latest/ .

        Parameters
        ----------
        distances : ndarray
            Array containing the pair wise distances.
        f_ij : chararray
            Array containing the formfactor product str labels in same order as
            the distances.

        Returns
        -------
        ndarray
            Scattered intensity of the sum over non-equal indices.
        """

        f_ij_e = []
        aa = eval(list((self.atom_numbers.keys()))[0]*2)
        ab = eval(list((self.atom_numbers.keys()))[0]+list((self.atom_numbers.
            keys()))[1])
        bb = eval(list((self.atom_numbers.keys()))[1]*2)

        f_ij_e = [ab if i == 5 else bb if i == 6 else aa for i in f_ij]

        f_ij_e = np.array(f_ij_e)
        rij_reshaped = distances.reshape(len(distances),1)
        q = self.q

        fijsincqr = ne.evaluate("f_ij_e*sin(q*rij_reshaped)/(q*rij_reshaped)")
        I_sum = fijsincqr.sum(axis=0)

        return 2*I_sum

    def calculate_Dse_total(self):
        """
        Calculate the Debye scattering equation for the whole nanocrystal.

        The partitioned distance and formfactor arrays are retrieved from the
        partition_crystal method. The for loop is over the number of partitions
        set during initialization. The intensities obtained from the parts are
        summed up and stored in I_ne (intensity for non-equal indices). The
        contributions from pairs with equal indices is obtained by summing over
        the kinds of atoms and adding the product of the number of atoms and the
        squared formfactor to I_diagonal. The numbers of atoms are stored in the
        atom_numbers dictionary. The squared formfactors are retrieved from the
        global variables corresponding to the str label. The string label is
        obtained by doubling the respective element label, e.g. eval(i*2) with
        i = 'Fe' --> eval('FeFe') = FeFe = ndarray.

        Returns
        -------
        ndarray
            Total scattered intensity as a function of q.

        """
        sliced_crystal, sliced_formfactors = self.partition_crystal()

        I_ne = 0
        for i in range(self.num_partitions):
            I_ne += self.calculate_Dse(sliced_crystal[i], sliced_formfactors[i])
            print("\t\t %d/%d finished" % (i+1, self.num_partitions))

        I_diagonal = 0
        if self.selfscattering:
            for i in self.atom_numbers.keys():
                print(self.atom_numbers[i], i)
                I_diagonal += self.atom_numbers[i]*eval(i*2)

        self.intensity = I_diagonal + I_ne
        return self.intensity

    def save_to_text(self, filename):
        """
        Saves the data in ASCII columns file, with columns q, 2Theta and I.

        Parameters
        ----------
        filename : str
            The filename under which the data will be saved. If no output path
            is given the file will be stored in the package folder.
        """
        data = np.column_stack((self.q,self.theta*2,self.intensity))
        header = "CIF-file: "+ self.cif_filename
        header += "\nOccupancies: "
        for key in self.occupancies.keys():
            header += key + ": " + str(self.occupancies[key])+ " "
        header += "\nCrystal shape: " + self.shape
        if self.shape == "cube":
             header += "\nEdgelength: %.2f nm" % (self.diameter)
        else:
            header += "\nDiameter: %.2f nm" % (self.diameter)

        header += "\nWavelength: %.5f" % (self.wavelength)
        header += " \u212B \nq\t\t\t\t2Theta\t\t\tIntensity\n"
        header += "\nTotal number of atoms: %d" %(self.num_Atoms)
        np.savetxt(self.output_path+filename, data, header=header)
        print("\nFile saved: " + self.output_path+filename)

    def perform_calculation(self, filename=None, plot=None):
        """
        Wrapper function to ensure results are saved and plotted or intentionally
        discarded.

        Parameters
        ----------
        filename : str, optional
            Filename used for saving the data. The default is None.
        plot : bool, optional
            Bool flag for plotting the result. The default is True.
        """
        print("\n\t starting calculation...")
        I = self.calculate_Dse_total()
        print("\t calculation finished.")
        if plot != None:
            if plot == "Q":
                plotting.plot_results("Q",self.q, self.intensity)
            elif plot == "2Theta":
                plotting.plot_results("2Theta",self.theta*2, self.intensity)
        if filename != None:
            self.save_to_text(filename)
        if filename == None:
            input1 = input("Do you want to save the file? [y/n]\n")
            if input1 == "y":
                input2 = input("Please enter a filename: ")
                self.save_to_text(input2)
            elif input1 == "n":
                return
```

B.5 plotting.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

```

```python
import matplotlib.pyplot as plt
import numpy as np

def plot_3D(crystal=None, coordinates = None, element_list = None):
    if crystal != None:
        coordinates, element_list = crystal.return_coordinate_list()
    else:
        coordinates = coordinates
        element_list = element_list

    colors = []
    for e in element_list:
        if e =="Fe":
            colors.append((0.2,0.2,0.0))
        elif e == "O":
            colors.append((0.8,0.0,0.0))
        else:
            colors.append((0.0,0.0,0.0))
    colors = np.array(colors)

    fig = plt.figure()
    ax = fig.gca(projection='3d', proj_type='ortho')

    x,y,z = zip(*coordinates)
    ax.scatter(x,y,z, color=colors)
    ax.xaxis.set_pane_color((1.0, 1.0, 1.0, 0.0))
    ax.yaxis.set_pane_color((1.0, 1.0, 1.0, 0.0))
    ax.zaxis.set_pane_color((1.0, 1.0, 1.0, 0.0))
    ax.xaxis._axinfo["grid"]['color'] =  (1,1,1,0)
    ax.yaxis._axinfo["grid"]['color'] =  (1,1,1,0)
    ax.zaxis._axinfo["grid"]['color'] =  (1,1,1,0)

    plt.show()

def plot_2D(crystal=None, coordinates = None, element_list = None):
    if crystal != None:
        coordinates, element_list = crystal.return_coordinate_list()
    else:
        coordinates = coordinates
        element_list = element_list

    colors = []
    for e in element_list:
        if e =="Fe":
            colors.append((0.2,0.2,0.0,0.5))
        elif e == "O":
            colors.append((0.8,0.0,0.0,0.0))
        else:
            colors.append((0.0,0.0,0.0))
    colors = np.array(colors)

    fig,ax = plt.subplots(1,1)

    x,y,z = zip(*coordinates)
    indices = []
    lim = crystal.diameter/2
    for n,i in enumerate(z):
        if i  >= (crystal.diameter/2)-lim and i <= (crystal.diameter/2)+lim:
            indices.append(n)
    xs = []
    ys = []
    cs = []

    for i in indices:
        xs.append(x[i])
        ys.append(y[i])
        cs.append(colors[i])

    ax.scatter(xs,ys, color=cs)
    ax.set_aspect('equal')
    plt.show()

def plot(data,xlabel,ylabel):
    fig,ax = plt.subplots(1,1)
    for d in data.keys():
        x,y,linewidth, marker = data[d]
        ax.plot(x,y,linewidth=linewidth, marker=marker,label=d)
    ax.legend()
    ax.set_xlabel(xlabel)
    ax.set_ylabel(ylabel)
    plt.show()

def plot_results(x_par,x,y):
    fig,ax = plt.subplots(1,1)
    ax.plot(x,y)
    if x_par == "Q":
        ax.set_xlabel("$Q [\AA^{-1}]$")
    elif x_par == "2Theta":
        ax.set_xlabel("$2\Theta$ [  ]")
    ax.set_ylabel("Intensity [a.u.]")
    plt.show()
```

B.6 Example.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import debye

Fe3O4_Fd3m = "/CIF-path/Fd-3m.cif"

debye.Experiment(wavelength = 0.21148,
                 occupancies = {'Fe(oct)':0.88, 'Fe(tet)':1.0},
                 gradient = None,
                 x_par = "Q",
                 x_min = 0.1,
                 x_max = 20.0,
                 x_step = 0.01,
                 partitions = 600,
                 diameter = 8,
                 offset = np.array([0.5,0.5,0.5]),
                 shape = 'Sphere',
                 cif_file = Fe3O4_Fd3m,
                 filename = "D8_APB_Fd3m_center_1",
                 plot_crystal=False,
                 APB = True,
                 plot_results= "Q",
                 plot_2D_crystal = False,
                 calculate = True,
                 output_path="output/path/",
                 dry_run=False,
                 saveStructure = "D8_APB_Fd3m_center" ,
                 SEED=20200104,
                 selfscattering=True)
```

Appendix C

Monte Carlo simulation program

In this appendix the source code for the Monte Carlo simulation program is given as described in section 3.7.1. The files *crystal.py* and *parsers.py* of the Debye scattering equation program were also used for the Monte Carlo program to build the nanoparticle structures and are not repeated here. The files *randNumGenerator.hpp* and *randNumGenerator.cpp* contain the random number generator developed by Ralf Meyer and Fred Hucht (Copyright 1995 Ralf Meyer, 47058 Duisburg, Germany) adapted for the use in this program. The Gaussian random number generator using the Ziggurat method developed by Jochen Voss is also contained in this file. The JSON parser (version 3.9.1) developed by Niels Lohmann is not printed here as it comprises almost 26 000 lines of code. The reader is referred to the GitHub repository at `https://github.com/nlohmann/json`. For the program the file *json.hpp* from this repository was included as JSON.h.

C.1 main.cpp

```cpp
#include <iostream>
#include "randNumGenerator.hpp"
#include "Crystal.hpp"
#include "constants.h"
#include "Measurement.hpp"
#include "JSON.h"
#include "testing.hpp"
#include <chrono>

int main(int argc, char* argv[]) {
    seed250(SEEED);

    if(argv[1] != NULL){
        // read configuration file
        std::string configfile = argv[1];
        std::ifstream ifs(configfile);
        nlohmann::json jf = nlohmann::json::parse(ifs);

        double macrocell_size;
        if(jf["dipole_interactions"]=="macrocell_method"){
            macrocell_size = jf["macrocell_size"];
        } else {
            macrocell_size = 0.0;
        }

        if(jf["Measurement"] == "M vs B"){
            run_MvsB(jf["output_dir"],
                     jf["structure_file"],
                     jf["dipole_interactions"],
                     jf["steps"],
                     jf["num_orientations"],
                     jf["temperature"],
                     jf["B_upper"],
                     jf["B_lower"],
                     jf["B_step"],
                     jf["cooling_field"],
                     jf["cooling_steps"],
                     jf["start_temperature"],
                     jf["temperature_step"],
                     jf["FeTT"],
                      jf["FeOO"],
                      jf["FeTO"],
                      jf["FeOO_APB"],
                      jf["anisotropy constant"],
                      macrocell_size,
                      jf["lattice_a"],
                      jf["lattice_b"],
                      jf["lattice_c"],
                      jf["particle center"],
                      jf["sigma"]);
        } else if (jf["Measurement"] == "spin structure"){
            run_spinstructure(jf["dipole_interactions"],
                           jf["steps"],
                           jf["magnetic field"],
                           jf["temperature"],
                           jf["output_dir"],
                           jf["structure_file"],
                           jf["FeTT"],
                           jf["FeOO"],
                           jf["FeTO"],
                           jf["FeOO_APB"],
                           jf["anisotropy constant"],
                           jf["alpha"],
                           jf["beta"],
                           jf["gamma"],
                           macrocell_size,
                           jf["particle center"],
                           jf["lattice_a"],
                           jf["lattice_b"],
                           jf["lattice_c"],
                           jf["sigma"]);
        } else if(jf["Measurement"] == "M vs T"){
            run_MvsT(jf["output_dir"],
                      jf["structure_file"],
                      jf["dipole_interactions"],
                      jf["steps"],
                      jf["averaging steps"],
                      jf["num_orientations"],
                      jf["measurement_field"],
                      jf["cooling_field"],
                      jf["TUpperLimit"],
                      jf["TLowerLimit"],
                      jf["TstepSize"],
                      jf["FeTT"],
                      jf["FeOO"],
                      jf["FeTO"],
                      jf["FeOO_APB"],
                      jf["anisotropy constant"],
                      jf["ZFC"],
                      jf["FC"],
                      macrocell_size,
                      jf["particle center"],
                      jf["lattice_a"],
                      jf["lattice_b"],
                      jf["lattice_c"],
                      jf["sigma"]);
        }
        else if (jf["Measurement"] == "test"){
            run_MvsB(jf["output_dir"],
                      jf["structure_file"],
                      "None",
                      jf["steps"],
                      jf["num_orientations"],
                      jf["temperature"],
                      jf["B_upper"],
                      jf["B_lower"],
                      jf["B_step"],
                      jf["cooling_field"],
                        jf["cooling_steps"],
                        jf["start_temperature"],
                        jf["temperature_step"],
                        jf["FeTT"],
                        jf["FeOO"],
                        jf["FeTO"],
                        jf["FeOO_APB"],
                        jf["anisotropy constant"],
                        macrocell_size,
                        jf["lattice_a"],
                        jf["lattice_b"],
                        jf["lattice_c"],
                        jf["particle center"],
                        jf["sigma"]);
            }
    } else{
        //dipole_calcs(false, 3, 4);


//          dipole_calcs(true);


//          relaxation_test(6, 0.03, 5000);
//          relaxation_test(6, 0.1, 5000);
//          relaxation_test(6, 0.3, 5000);
//
//          relaxation_test(8, 0.03, 5000);
//          relaxation_test(8, 0.1, 5000);
//          relaxation_test(8, 0.3, 5000);

//          relaxation_test(11, 0.03, 5000);
//          relaxation_test(11, 0.1, 5000);
//          relaxation_test(11, 0.3, 5000);

       // sigma_tests(0.4);
    }

    return 0;
}
```

C.2 Atom.hpp

```
#ifndef Atom_hpp
#define Atom_hpp

#include <stdio.h>
#include <vector>
#include <iostream>
#include <cmath>
#include "constants.h"
#include "randNumGenerator.hpp"
#include "omp.h"

class Atom;

class Macrocell{
public:
    double center_x;
    double center_y;
    double center_z;
    double total_moment[3];
    double previous_total_moment[3];
    double inv_effective_volume;
    bool isEmpty = true;
    std::vector<Atom*> macrocell_atoms;
    std::vector<Macrocell*> all_other_macrocells;

    std::vector<double> inv_distances_cubed;
    std::vector<double> distVecX;
    std::vector<double> distVecY;
    std::vector<double> distVecZ;

    Macrocell(double x,double y,double z);
    void update_total_moment();
    void get_demag_field(double*, double);
    void reset_total_moment();
};

class Atom {
    /*
     class to represent atomic magnetic moments with spatial position
     x,y and z, spin vector components spinx, spiny and spinz and
     other magnetic properties.
     */
public:
    double x = 0;   // atom coordinates
    double y = 0;
    double z = 0;
    double spinx = 0; // spin components
    double spiny = 0;
    double spinz = 0;
    double uc_x = 0.0; // unitCell_comp
    double uc_y = 0.0;
    double uc_z = 0.0;

    // exchange interaction constants
    double FeTT = 0.0;
    double FeOO = 0.0;
    double FeTO = 0.0;
    double FeOO_APB = 0.0;

    double anisotropyConstant = 0.0;

    int APB=0; // APB=1 --> APB position
    int position = 0;// Octahedral = 0 or tetrahedral = 1 position
    bool isApbAtom = false;
    bool isSurfaceAtom = false;
    int dipole_interactions; // 0 --> brute force,
    //1 --> macrocell method, 2 --> no dipole interactions
    bool macrocell_method;

    double sigma = 0.0; // defines opening angle of Gaussian cone
    double MagMagMu0 = MAGFE3*MAGFE3*MU0;

    Macrocell* macrocell_link = NULL;
    double H_demag[3] = {0.0,0.0,0.0};

    // Arrays containing pointers to Atom objects needed for the calculations
    std::vector<Atom*> neighboursAntiparallel;
    std::vector<Atom*> neighboursParallel;
    std::vector<Atom*> allOtherAtomsInCrystal;
    std::vector<double> inv_distances_cubed;
    std::vector<double> inv_distances_five;
    std::vector<double> distVecX;
    std::vector<double> distVecY;
    std::vector<double> distVecZ;
    std::vector<double> magmag;

    // initializer
    Atom(int dipole_interactions,
         double x, double y, double z,
         int position, int APB,
         double FeTT, double FeOO, double FeTO, double FeOO_APB,
            double anisotropyConstant);

    // energy functions
    double anisotropy();
    double exchange();
    double zeeman(double);
    double zeeman3D(double*);
    double dipole();

    void dipole_field(double*);

    // trial move functions
    void uniform(double *old_spin);
    void uniform_ziggurat(double *old_spin);
    void spin_flip(double*);
    void angle(double *old_spin);
    void hinzke_nowak(double *old_spin);
    void cattaneo_sun(double *old_spin);

    void MonteCarloStep(double Bx, double temperature);
};

#endif /* Atom_hpp */
```

C.3 Atom.cpp

```
#include "Atom.hpp"

Atom::Atom(int dipole_interactions,
           double x, double y , double z,
           int position, int APB,
           double FeTT, double FeOO, double FeTO, double FeOO_APB,
           double anisotropyConstant):
           dipole_interactions(dipole_interactions),
           x(x), y(y), z(z),
           position(position), APB(APB),
           FeTT(FeTT), FeOO(FeOO), FeTO(FeTO), FeOO_APB(FeOO_APB),
           anisotropyConstant(anisotropyConstant){

    if (APB == 1){
        isApbAtom = true;
    }

    /* when an atom object gets initialized, the spin vector is
    set to a random orientation */
    double s[3];
    marsaglia(s);
    spinx = s[0];
    spiny = s[1];
    spinz = s[2];

}

double Atom::anisotropy() {
    /* magnetocrystalline anisotropy according to
       E_anis = -k/2 (S_x^4 + S_y^4 + S_z^4),
       where k is the anisotropy constant per magnetic moment.
     */
    return -anisotropyConstant*0.5 * (spinx*spinx*spinx*spinx +
                                      spiny*spiny*spiny*spiny +
                                      spinz*spinz*spinz*spinz);
}

double Atom::exchange() {
    double Enna = 0.0;
    double Ennp = 0.0;

    for(unsigned int i = 0; i < neighboursParallel.size(); i++){
        if((neighboursParallel[i]->isApbAtom == true) && (isApbAtom == true)){
            Ennp += -FeOO_APB *(neighboursParallel[i]->spinx * spinx +
                                neighboursParallel[i]->spiny * spiny +
                                neighboursParallel[i]->spinz * spinz);
        }

        else if(position == 0){
            Ennp += -FeOO  *(neighboursParallel[i]->spinx * spinx +
                             neighboursParallel[i]->spiny * spiny +
                             neighboursParallel[i]->spinz * spinz);
        }
        else if(position == 1){
            Ennp += -FeTT * (neighboursParallel[i]->spinx * spinx +
                             neighboursParallel[i]->spiny * spiny +
                             neighboursParallel[i]->spinz * spinz);
        }

    }
    for(unsigned int i = 0; i < neighboursAntiparallel.size(); i++){
        Enna += -FeTO * (neighboursAntiparallel[i]->spinx * spinx +
                         neighboursAntiparallel[i]->spiny * spiny +
                         neighboursAntiparallel[i]->spinz * spinz);
    }
    return((Enna + Ennp)*KB);
}

double Atom::zeeman(double Bx) {
    return (-Bx * spinx * MAGFE3);
}

double Atom::zeeman3D(double *B) {
    return (-B[0] * spinx * MAGFE3 + -B[1] * spiny * MAGFE3 + -B[2] * spinz *
        MAGFE3);
}

void Atom::dipole_field(double *H_dip){
    H_dip[0] = 0.0;
    H_dip[1] = 0.0;
    H_dip[2] = 0.0;

    double A, Ax, Ay, Az;
    double Bx, By, Bz;

    for(int i=0; i<(int)inv_distances_cubed.size(); i++){
        A = allOtherAtomsInCrystal[i]->spinx * distVecX[i] +
            allOtherAtomsInCrystal[i]->spiny * distVecY[i] +
            allOtherAtomsInCrystal[i]->spinz * distVecZ[i];
        Bx = allOtherAtomsInCrystal[i]->spinx * inv_distances_cubed[i];
        By = allOtherAtomsInCrystal[i]->spiny * inv_distances_cubed[i];
        Bz = allOtherAtomsInCrystal[i]->spinz * inv_distances_cubed[i];
        Ax = 3.0 * A * distVecX[i] * inv_distances_five[i];
        Ay = 3.0 * A * distVecY[i] * inv_distances_five[i];
        Az = 3.0 * A * distVecZ[i] * inv_distances_five[i];

        H_dip[0] += Ax - Bx;
        H_dip[1] += Ay - By;
        H_dip[2] += Az - Bz;
    }
}


double Atom::dipole(){
    /*
     function to calculate the dipole energy of a magnetic moment by brute force,
     i.e. summing all contributions from each pair wise interaction. By looping
     over all other atoms in the structue no pairs are counted double.
     */
    long double E_d = 0.0;
    // loop through the list that contains all the distances from this atom to
        every other in the crystal
    // omp_set_dynamic disables automatic adjustment of the number of threads
    // 6 threads seemed to work best
    omp_set_dynamic(0);
    # pragma omp parallel for reduction(+:E_d) num_threads(6)
    {
        for(int i=0; i<(int)inv_distances_cubed.size(); i++){
            /* allOtherAtomsInCrystal contains the memory addresses of every
             atom in the crystal in the
             same order as the distances are stored in distances,
             dereferencing the addresses by "->" gives
             access to the spin porperties of these atoms spinx, spiny,
             spinz refer to the spin properties
             of the current atom object */

            double dotProd = spinx * allOtherAtomsInCrystal[i]->spinx +
                             spiny * allOtherAtomsInCrystal[i]->spiny +
                             spinz * allOtherAtomsInCrystal[i]->spinz;
            double m1r = spinx * distVecX[i] +
                         spiny * distVecY[i] +
                         spinz * distVecZ[i];
            double m2r = allOtherAtomsInCrystal[i]->spinx * distVecX[i] +
                         allOtherAtomsInCrystal[i]->spiny * distVecY[i] +
                         allOtherAtomsInCrystal[i]->spinz * distVecZ[i];

            /* constants are precomputed and pulled out of the for loop
             to increase speed:
             (5uB^2*u0=2.7019978...e-51)/(...distances_cubed*1e-30) */
            E_d += inv_distances_cubed[i]*((3.0*m1r*m2r)-dotProd);
        }
    } // end of parallel
    /* mu0*5uB*5uB/(4PI*1e-30) =
      (-2.701997855309151e-21)/(4.0*3.141592653589793238462643383)
    =-2.150181574480756e-22
     the factor 1e-30 is due to the distances being computed in Angstroms */
    return -2.150181574480756e-22*E_d;
}

void Atom::angle(double *old_spin){
    old_spin[0] = spinx;
    old_spin[1] = spiny;
    old_spin[2] = spinz;
    spinx += gaussian_ziggurat()*sigma;
    spiny += gaussian_ziggurat()*sigma;
    spinz += gaussian_ziggurat()*sigma;
    double vectorLength = 1.0/std::sqrt(spinx*spinx+spiny*spiny+spinz*spinz);
    spinx *= vectorLength;
    spiny *= vectorLength;
    spinz *= vectorLength;
}

void Atom::uniform_ziggurat(double *old_spin){
    old_spin[0] = spinx;
    old_spin[1] = spiny;
    old_spin[2] = spinz;
    spinx = gaussian_ziggurat();
    spiny = gaussian_ziggurat();
    spinz = gaussian_ziggurat();
}

void Atom::uniform(double *old_spin){
    old_spin[0] = spinx;
    old_spin[1] = spiny;
    old_spin[2] = spinz;
    double spinRotationVector[3];
    marsaglia(spinRotationVector);
    spinx = spinRotationVector[0];
    spiny = spinRotationVector[1];
    spinz = spinRotationVector[2];
}

void Atom::spin_flip(double* old_spin){
    old_spin[0] = spinx;
    old_spin[1] = spiny;
    old_spin[2] = spinz;
    spinx = -spinx;
    spiny = -spiny;
    spinz = -spinz;
}

void Atom::cattaneo_sun(double *old_spin){
    old_spin[0] = spinx;
    old_spin[1] = spiny;
    old_spin[2] = spinz;
    double spinRotationVector[3];
    marsaglia(spinRotationVector);
    spinx += (testRotationVectorLength * spinRotationVector[0]);
    spiny += (testRotationVectorLength * spinRotationVector[1]);
    spinz += (testRotationVectorLength * spinRotationVector[2]);
    double vectorLength = 1.0/std::sqrt(spinx*spinx+spiny*spiny+spinz*spinz);
    spinx *= vectorLength;
    spiny *= vectorLength;
    spinz *= vectorLength;
}

void Atom::hinzke_nowak(double *old_spin){
    const int pick_move=int(3.0*rand0_1());

    switch(pick_move){
        case 0:
            spin_flip(old_spin);
            break;
        case 1:
            uniform(old_spin);
            break;
        case 2:
            angle(old_spin);
            break;
        default:
            angle(old_spin);
            break;
    }
}


void Atom::MonteCarloStep(double Bx, double temperature){

    double Ez = 0.0;
    double Ea = 0.0;
    double Ee = 0.0;
    double Ed = 0.0;
    double E0 = 0.0;
    double E1 = 0.0;
    double tempSpinVector[3];

    switch(dipole_interactions){
        case 0:
            // brute force
            Ea = anisotropy();
            Ee = exchange();
            Ez = zeeman(Bx);
            Ed = dipole();
```

```
                E0 = Ez + Ea + Ee + Ed;

                hinzke_nowak(tempSpinVector);

                Ea = anisotropy();
                Ee = exchange();
                Ez = zeeman(Bx);
                Ed = dipole();
                E1 = Ez + Ea + Ee + Ed;

                if(E1 > E0){
                    if(rand0_1() > std::exp((E0-E1)/(KB*temperature))){
                        spinx = tempSpinVector[0];
                        spiny = tempSpinVector[1];
                        spinz = tempSpinVector[2];
                    }
                }
                break;
            case 1:
                // macrocell method
                macrocell_link->get_demag_field(H_demag,Bx);

                Ea = anisotropy();
                Ee = exchange();
                Ez = zeeman3D(H_demag);
                E0 = Ez + Ea + Ee;

                hinzke_nowak(tempSpinVector);

                macrocell_link->update_total_moment();
                macrocell_link->get_demag_field(H_demag,Bx);

                Ea = anisotropy();
                Ee = exchange();
                Ez = zeeman3D(H_demag);
                E1 = Ez + Ea + Ee;

                if(E1 > E0){
                    if(rand0_1() > std::exp((E0-E1)/(KB*temperature))){
                        macrocell_link->reset_total_moment();

                        spinx = tempSpinVector[0];
                        spiny = tempSpinVector[1];
                        spinz = tempSpinVector[2];
                    }
                }
                break;
            default:
                // no dipole interactions
                Ea = anisotropy();
                Ee = exchange();
                Ez = zeeman(Bx);
                E0 = Ez + Ea + Ee;

                //cattaneo_sun(tempSpinVector);
                hinzke_nowak(tempSpinVector);

                Ea = anisotropy();
                Ee = exchange();
                Ez = zeeman(Bx);
                E1 = Ez + Ea + Ee;

                if(E1 > E0){
                    if(rand0_1() > std::exp((E0-E1)/(KB*temperature))){
                        spinx = tempSpinVector[0];
                        spiny = tempSpinVector[1];
                        spinz = tempSpinVector[2];
                    }
            }
    }
}



Macrocell::Macrocell(double x,double y,double z):center_x(x),center_y(y),center_z
    (z){}

void Macrocell::update_total_moment(){
    previous_total_moment[0] = total_moment[0];
    previous_total_moment[1] = total_moment[1];
    previous_total_moment[2] = total_moment[2];
    total_moment[0] = 0.0;
    total_moment[1] = 0.0;
    total_moment[2] = 0.0;
    for(int i=0; i<macrocell_atoms.size(); i++){
        total_moment[0] += macrocell_atoms[i]->spinx*MAGFE3;
        total_moment[1] += macrocell_atoms[i]->spiny*MAGFE3;
        total_moment[2] += macrocell_atoms[i]->spinz*MAGFE3;
    }
}

void Macrocell::reset_total_moment(){
    total_moment[0] = previous_total_moment[0];
    total_moment[1] = previous_total_moment[1];
    total_moment[2] = previous_total_moment[2];

}

void Macrocell::get_demag_field(double *H_demag, double Bx){

    double Hd_x = 0.0;
    double Hd_y = 0.0;
    double Hd_z = 0.0;
    double R;

    omp_set_dynamic(0);
    # pragma omp parallel for reduction(+:Hd_x,Hd_y,Hd_z) num_threads(6)
    {
    for(int i=0; i<(int)inv_distances_cubed.size(); i++){
            R = 3.0*(all_other_macrocells[i]->total_moment[0] * distVecX[i]
               + all_other_macrocells[i]->total_moment[1] * distVecY[i]
               + all_other_macrocells[i]->total_moment[2] * distVecZ[i]);
            Hd_x += (R*distVecX[i] - all_other_macrocells[i]->total_moment[0])
                    *inv_distances_cubed[i];
            Hd_y += (R*distVecY[i] - all_other_macrocells[i]->total_moment[1])
                    *inv_distances_cubed[i];
            Hd_z += (R*distVecZ[i] - all_other_macrocells[i]->total_moment[2])
                    *inv_distances_cubed[i];
    }
    } // end of parallel
    double f1 =MU0*1e30/(4.0*PI);
    Hd_x *=f1;
    Hd_y *=f1;
    Hd_z *=f1;

    H_demag[0] = Bx + (Hd_x - MU0/3.0*total_moment[0]*inv_effective_volume);
    H_demag[1] =(Hd_y - MU0/3.0*total_moment[1]*inv_effective_volume);
    H_demag[2] =(Hd_z - MU0/3.0*total_moment[2]*inv_effective_volume);
}
```

C.4 Crystal.hpp

```
#ifndef Crystal_hpp
```

```
#define Crystal_hpp

#include <stdio.h>
#include <vector>
#include <cmath>
#include <fstream>
#include <string>

#include "Atom.hpp"
#include "randNumGenerator.hpp"
#include "constants.h"



class Crystal{
public:
    std::vector<Atom> atoms;
    // lattice parameters
    double lattice_a;
    double lattice_b;
    double lattice_c;
    double center;

    Crystal(std::string filename, std::string dipole_interactions,
            double FeTT, double FeOO, double FeTO, double FeOO_APB,
            double anisotropyConstant, double alpha, double beta,
            double gamma, double macrocell_size, double center,
            double lattice_a, double lattice_b, double lattice_c, double sigma);
    void rotateCrystal(double, double, double, double);
    void random_orientation();
    void align_along_random_vector();
    void structure_snapshot(std::string filename);
    void reset_structure();
    int outputStats();
    void set_sigma(double sigma);
    void generate_dipole_lists();
    void generate_neighbour_lists();
    void read_structure_from_file(std::string filename,
                                  std::string dipole_interactions,
                                  double FeTT, double FeOO, double FeTO, double
                                      FeOO_APB,
                                  double anisotropyConstant);

    std::vector<Macrocell> macrocells;
    void generate_macrocells(double macrocell_size);
    void save_macrocells(std::string filename);

};

#endif /* Crystal2_hpp */
```

C.5 Crystal.cpp

```
#include "Crystal.hpp"


Crystal::Crystal(std::string filename, std::string dipole_interactions,
                 double FeTT, double FeOO, double FeTO, double FeOO_APB,
                 double anisotropyConstant, double alpha, double beta,
                 double gamma, double macrocell_size, double center,
                 double lattice_a, double lattice_b, double lattice_c, double
                     sigma):
                 lattice_a(lattice_a), lattice_b(lattice_b),
                lattice_c(lattice_c), center(center){

```

```
    read_structure_from_file(filename, dipole_interactions,
                             FeTT,  FeOO,  FeTO,  FeOO_APB,
                             anisotropyConstant);
    set_sigma(sigma);
    rotateCrystal(alpha, beta, gamma, center);
    generate_neighbour_lists();

    if(dipole_interactions == "brute_force"){
        generate_dipole_lists();
    } else if(dipole_interactions == "macrocell_method"){
        generate_macrocells(macrocell_size);
    }
}

void Crystal::read_structure_from_file(std::string filename,
                                       std::string dipole_interactions,
                                       double FeTT, double FeOO, double FeTO,
                                       double FeOO_APB,
                                       double anisotropyConstant){

    std::vector<double> x;
    std::vector<double> y;
    std::vector<double> z;
    std::vector<int> pos;
    std::vector<int> apb;

    std::ifstream inFile;
    inFile.open(filename);
    std::string line;
    int num = 0;
    while (std::getline(inFile, line)){
            ++num;
    }
    inFile.close();

    x.resize(num); y.resize(num); z.resize(num);
    pos.resize(num); apb.resize(num);

    inFile.open(filename);
    for(int i =0; i<num; i++){
        double xr,yr,zr;
        double posr,apbr,uc_xr,uc_yr,uc_zr;
        inFile >> xr >> yr >> zr >> posr >> apbr >> uc_xr >> uc_yr >> uc_zr ;
        x[i] = xr;
        y[i] = yr;
        z[i] = zr;
        pos[i] = posr;
        apb[i] = apbr;
    }
    inFile.close();

    int dip;
    if(dipole_interactions == "brute_force"){
        dip = 0;
    } else if(dipole_interactions == "macrocell_method"){
        dip = 1;
    } else {
        dip = 2;
    }

    for(int i=0; i<num; i++){
        atoms.push_back(Atom(dip,
                             x[i], y[i], z[i],
                             pos[i], apb[i],
                             FeTT,  FeOO,  FeTO, FeOO_APB,
                             anisotropyConstant));
    }
}
```

```

void Crystal::generate_neighbour_lists(){
    double xx, yy, zz;

    for(int i=0; i<atoms.size(); i++){
        atoms[i].neighboursAntiparallel.reserve(12);
        atoms[i].neighboursParallel.reserve(12);

        for(int j=0; j<atoms.size(); j++){
            xx = atoms[j].x - atoms[i].x;
            yy = atoms[j].y - atoms[i].y;
            zz = atoms[j].z - atoms[i].z;

            if((i!=j) and ((xx*xx + yy*yy + zz*zz) < 0.2505)){
                if(atoms[i].position == atoms[j].position){
                    atoms[i].neighboursParallel.push_back(&atoms[j]);
                }
                else{
                    atoms[i].neighboursAntiparallel.push_back(&atoms[j]);
                }
            }
        }
    }
}

void Crystal::generate_dipole_lists(){
    double x, y, z;
    double a_sq, b_sq, c_sq;
    double distance;
    double rx, ry, rz;

    a_sq = lattice_a*lattice_a;
    b_sq = lattice_b*lattice_b;
    c_sq = lattice_c*lattice_c;

    for(unsigned int i=0; i< atoms.size(); i++){
        x = atoms[i].x;
        y = atoms[i].y;
        z = atoms[i].z;
        for(unsigned int j=0; j< atoms.size(); j++){
            if(&atoms[i] != &atoms[j]){
                distance = std::sqrt((atoms[j].x - x)*(atoms[j].x - x)*a_sq+
                                     (atoms[j].y - y)*(atoms[j].y - y)*b_sq+
                                     (atoms[j].z - z)*(atoms[j].z - z)*c_sq);

                //unit vectors
                rx = (atoms[j].x - x)*lattice_a/distance;
                ry = (atoms[j].y - y)*lattice_b/distance;
                rz = (atoms[j].z - z)*lattice_c/distance;

                atoms[i].inv_distances_cubed.push_back(1.0/(std::pow((distance)
                    ,3)));
                atoms[i].inv_distances_five.push_back(1.0/(std::pow((distance),5)
                    ));
                atoms[i].distVecX.push_back(rx);
                atoms[i].distVecY.push_back(ry);
                atoms[i].distVecZ.push_back(rz);

                atoms[i].magmag.push_back(MAGFE3*MAGFE3);

                atoms[i].allOtherAtomsInCrystal.push_back(&atoms[j]);
            } // end of distance calculation
        } // end of loop over other atoms
    } // end of loop over all atoms
}

int Crystal::outputStats(){
    int totalAtoms = (int)atoms.size();
    int apbAtoms = 0;
    int oct = 0;
    int tet = 0;

    for(int i =0; i< totalAtoms; i++){
        if(atoms[i].isApbAtom){
            apbAtoms++;
            if(atoms[i].position == 0){
                oct++;
            } else if(atoms[i].position == 1){
                tet++;
            }
        }
    }
    std::cout << "\nTotal number of Atoms: " << totalAtoms << std::endl;
    std::cout << "Number of APB atoms: " << apbAtoms << " (tet: "<< tet
    << "; oct: "<< oct << ")\n"<< std::endl;
    return totalAtoms;
}

void Crystal::structure_snapshot(std::string filename){
       std::ofstream structure;
       structure.open(filename, std::fstream::out);
       for(int i=0; i< atoms.size(); i++){
           structure << atoms[i].x << ", " << atoms[i].y << ", " << atoms[i].z
           <<", "<< atoms[i].spinx << ", " << atoms[i].spiny << ", " << atoms[i].
               spinz
           << ", " << atoms[i].position << ", " << atoms[i].isApbAtom << "\n";
       }
}

void Crystal::generate_macrocells(double macrocell_size){

    // find minimum position
    double x_min=atoms[0].x, y_min=atoms[0].y, z_min=atoms[0].z;
    for(int i=0; i<atoms.size(); i++){
        if(atoms[i].x < x_min){
            x_min = atoms[i].x;
        }
        if(atoms[i].y < y_min){
            y_min = atoms[i].y;
        }
        if(atoms[i].z < z_min){
            z_min = atoms[i].z;
        }
    }
    // find maximum position
    double x_max=atoms[0].x, y_max=atoms[0].y, z_max=atoms[0].z;
    for(int i=0; i<atoms.size(); i++){
        if(atoms[i].x > x_max){
            x_max = atoms[i].x;
        }
        if(atoms[i].y > y_max){
            y_max = atoms[i].y;
        }
        if(atoms[i].z > z_max){
            z_max = atoms[i].z;
        }
    }

    // determine number of macrocells needed
    int num_macrocells = 5;
    while(num_macrocells*macrocell_size <= x_max+0.2){
        num_macrocells++;
    }

    // initialize Macrocell instances
    for(int i=0; i<num_macrocells; i++){
        for(int j=0; j<num_macrocells; j++){
            for(int k=0; k<num_macrocells; k++){
                Macrocell macrocell(macrocell_size/2.0+(double)i*macrocell_size,
                                    macrocell_size/2.0+(double)j*macrocell_size,
                                    macrocell_size/2.0+(double)k*macrocell_size);
                macrocells.push_back(macrocell);
            }
        }
    }

    // assign atoms to macrocells
    double w = macrocell_size/2.0;
    for(int j=0; j<atoms.size(); j++){
        for(int i=0; i<macrocells.size(); i++){
            if(atoms[j].x < macrocells[i].center_x+w and
               atoms[j].x >= macrocells[i].center_x-w and
               atoms[j].y < macrocells[i].center_y+w and
               atoms[j].y >= macrocells[i].center_y-w and
               atoms[j].z < macrocells[i].center_z+w and
               atoms[j].z >= macrocells[i].center_z-w){
                macrocells[i].macrocell_atoms.push_back(&atoms[j]);
            }
        }
    }

    // flag and remove empty cells
    for(int i=0; i<macrocells.size(); i++){
        if(macrocells[i].macrocell_atoms.size()!=0){
            macrocells[i].isEmpty = false;
        } else {
            macrocells[i].isEmpty = true;
        }
    }
    macrocells.erase(std::remove_if(macrocells.begin(),
                                    macrocells.end(),
                                    [](Macrocell i){ return i.isEmpty==true;}),
                     macrocells.end());

    // Set pointers to macrocells for atoms
    for(int i=0; i<macrocells.size(); i++){
        for(int j=0; j<macrocells[i].macrocell_atoms.size(); j++){
            macrocells[i].macrocell_atoms[j]->macrocell_link = &macrocells[i];
        }
    }

    // shift macrocell center to center of mass and calculate effective volume
    for(int i=0; i<macrocells.size(); i++){
        double n = 0.0;
        macrocells[i].center_x = 0.0;
        macrocells[i].center_y = 0.0;
        macrocells[i].center_z = 0.0;
        // effective volume = N_atoms_per_macrocell*V_atom =
            N_atoms_per_macrocell * V_unit_cell/N_atoms_per_unit_cell
        macrocells[i].inv_effective_volume = 1.0/((macrocells[i].macrocell_atoms.
            size()*std::pow(8.3965,3)/24.0)*1e-30);
        for(int j=0; j<macrocells[i].macrocell_atoms.size(); j++){
            macrocells[i].center_x += macrocells[i].macrocell_atoms[j]->x;
            macrocells[i].center_y += macrocells[i].macrocell_atoms[j]->y;
            macrocells[i].center_z += macrocells[i].macrocell_atoms[j]->z;
            n+=1.0;
        }
        macrocells[i].center_x /= n;
        macrocells[i].center_y /= n;
        macrocells[i].center_z /= n;
    }

    // precalculate distances and distance vectors and initialize macrocell
        moment
    for(int i=0; i< macrocells.size(); i++){
        double x = macrocells[i].center_x;
        double y = macrocells[i].center_y;
        double z = macrocells[i].center_z;
        double distance = 0.0;
        double rx = 0.0, ry = 0.0, rz = 0.0;
        for(int j=0; j< macrocells.size(); j++){
            if(&macrocells[i] != &macrocells[j]){
                distance = std::sqrt((macrocells[j].center_x - x)*(macrocells[j].
                    center_x - x)+
                                    (macrocells[j].center_y - y)*(macrocells[j].
                                        center_y - y)+
                                    (macrocells[j].center_z - z)*(macrocells[j].
                                        center_z - z));

                //unit vectors
                rx = (macrocells[j].center_x - x)/distance;
                ry = (macrocells[j].center_y - y)/distance;
                rz = (macrocells[j].center_z - z)/distance;

                macrocells[i].inv_distances_cubed.push_back(1.0/(std::pow((
                    distance*8.3965),3)));
                macrocells[i].distVecX.push_back(rx);
                macrocells[i].distVecY.push_back(ry);
                macrocells[i].distVecZ.push_back(rz);

                macrocells[i].all_other_macrocells.push_back(&macrocells[j]);
            }
        }
        macrocells[i].total_moment[0] = 0.0;
        macrocells[i].total_moment[1] = 0.0;
        macrocells[i].total_moment[2] = 0.0;
        for(int k=0; k<macrocells[i].macrocell_atoms.size(); k++){
            macrocells[i].total_moment[0] += macrocells[i].macrocell_atoms[k]->
                spinx*MAGFE3;
            macrocells[i].total_moment[1] += macrocells[i].macrocell_atoms[k]->
                spiny*MAGFE3;
            macrocells[i].total_moment[2] += macrocells[i].macrocell_atoms[k]->
                spinz*MAGFE3;
        }
    }
}

void Crystal::save_macrocells(std::string filename){
    std::ofstream macrocells_centers;
    macrocells_centers.open(filename, std::fstream::out);
    for(int i=0; i< macrocells.size(); i++){
        macrocells_centers << macrocells[i].center_x << " "
                           << macrocells[i].center_y << " "
                           << macrocells[i].center_z << std::endl;
    }
}

void Crystal::reset_structure(){
    for(int i=0; i<atoms.size(); i++){
        double s[3];
        marsaglia(s);
        atoms[i].spinx = s[0];
        atoms[i].spiny = s[1];
        atoms[i].spinz = s[2];
    }
}

void Crystal::set_sigma(double sigma){
    for(int i=0; i< atoms.size(); i++){
        atoms[i].sigma = sigma;
    }
}
```

```

void Crystal::rotateCrystal(double alpha,double beta,double gamma, double center)
    {

    double alphaRad = alpha * PI/180;
    double betaRad = beta * PI/180;
    double gammaRad = gamma * PI/180;

    double center_x = center;
    double center_y = center;
    double center_z = center;

    double x, y, z;
    double x2, y2, z2;
    double x3, y3, z3;


    for (unsigned int i = 0 ; i< atoms.size(); i++){
        // set particle into coordinate system origin
        x = atoms[i].x - center_x;
        y = atoms[i].y - center_y;
        z = atoms[i].z - center_z;

        // x-rotation with rotation matrix
        atoms[i].x = x*1.0 + y*0.0 + z*0.0;
        atoms[i].y = x*0.0 + y*std::cos(alphaRad) + z*-(std::sin(alphaRad));
        atoms[i].z = x*0.0 + y*std::sin(alphaRad) + z*std::cos(alphaRad);

        // temporary storing the new coordinates
        x2 = atoms[i].x;
        y2 = atoms[i].y;
        z2 = atoms[i].z;

        // y-rotation
        atoms[i].x = x2*std::cos(betaRad) + y2*0.0 + z2*std::sin(betaRad);
        atoms[i].y = x2*0.0 + y2*1.0 + z2*0.0;
        atoms[i].z = x2*-(std::sin(betaRad)) + y2*0.0 + z2*std::cos(betaRad);

        x3 = atoms[i].x;
        y3 = atoms[i].y;
        z3 = atoms[i].z;

        // z-rotation
        atoms[i].x = x3*std::cos(gammaRad) + y3*-(std::sin(gammaRad)) + z3*0.0;
        atoms[i].y = x3*std::sin(gammaRad) + y3*std::cos(gammaRad) + z3*0.0;
        atoms[i].z = x3*0.0 + y3*0.0 + z3*1.0;

        // particle gets set back to original position in space
        atoms[i].x = atoms[i].x + center_x;
        atoms[i].y = atoms[i].y + center_y;
        atoms[i].z = atoms[i].z + center_z;
    }
}

void Crystal::random_orientation(){
    double angles[3];
    rand0_360(angles);
    rotateCrystal(angles[0], angles[1], angles[2], center);
}

void Crystal::align_along_random_vector(){
    reset_structure();
    double random_magnetization_vector[3];
    marsaglia(random_magnetization_vector);
    for(int atom = 0; atom<(int)atoms.size(); atom++){
        if( atoms[atom].position == 0){
            atoms[atom].spinx = random_magnetization_vector[0];
            atoms[atom].spiny = random_magnetization_vector[1];
            atoms[atom].spinz = random_magnetization_vector[2];
        } else {
            atoms[atom].spinx = -random_magnetization_vector[0];
            atoms[atom].spiny = -random_magnetization_vector[1];
            atoms[atom].spinz = -random_magnetization_vector[2];
        }
    }
}
```

C.6 Measurement.hpp

```
#ifndef Measurement_hpp
#define Measurement_hpp

#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include "Crystal.hpp"
#include "randNumGenerator.hpp"
#include "omp.h"
#include "ProgressBar.hpp"

class MvsTMeasurement{
    // Class for M(T) simulations.
public:
    // general settings
    std::string dipoleInteractions;
    double steps;
    int averaging_steps;
    int numOrientations;
    double measurement_field;
    double cooling_field;
    double macrocell_size;

    // temperature sweep settings
    double TUpperLimit;
    double TLowerLimit;
    double TstepSize;

    // arrays to record the magnetization
    std::vector<double> mxZFC; // Zero field cooled
    std::vector<double> myZFC;
    std::vector<double> mzZFC;

    std::vector<double> mxCFZ; // cooling zero field
    std::vector<double> myCFZ;
    std::vector<double> mzCFZ;

    std::vector<double> mxFC; // field cooled
    std::vector<double> myFC;
    std::vector<double> mzFC;

    // class initializer
    MvsTMeasurement(std::string dipoleInteractions, double steps,
                    int averaging_steps, int numOrientations,
                    double measurement_field, double cooling_field,
                    double TUpperLimit, double TLowerLimit,
                    double TstepSize, double macrocell_size);

    // function to run the simulation
    void temperatureSweep(std::string output_dir, std::string structure_filename,
                          double FeTT, double FeOO, double FeTO, double FeOO_APB,
                          double anisotropyConstant,bool ZFC, bool FC, double
                              center,
                                double lattice_a, double lattice_b, double lattice_c,
                                double sigma);
};


class MvsBMeasurement{
    // Class for M(B) simulations.
public:
    // general settings
    std::string dipoleInteractions;
    int steps;
    int numOrientations;
    double temperature;
    double macrocell_size;

    // cooling setup
    double startTemp;
    double tempStep;
    double coolingField;
    int coolingSteps;

    // field sweep settings
    int BnumberOfSteps;
    double BUpperLimit;
    double BLowerLimit;
    double BstepSize;
    std::vector<double> magneticField;

    // MvsB arrays for the x, y and z components
    std::vector<double> mX;
    std::vector<double> mY;
    std::vector<double> mZ;

    // class initializer
    MvsBMeasurement(std::string dipoleInteractions,
                    int steps, int numOrientations,
                    double temperature, double BUpperLimit,
                    double BLowerLimit, double BstepSize,
                    double startTemp, double tempStep,
                    double coolingField, int coolingSteps,
                    double macrocell_size);

    // function to run the simulation
    void fieldSweep(std::string output_dir, std::string structure_filename,
                    double FeTT, double FeOO, double FeTO, double FeOO_APB,
                    double anisotropyConstant,
                    double lattice_a, double lattice_b, double lattice_c,
                    double center, double sigma);
};

class spinStructure{
    // Class for spin structure simulations
public:
    std::string output_dir;
    std::string structure_filename;

    std::string dipoleInteractions;
    int steps;
    double magneticField;
    double temperature;
    double macrocell_size;

    spinStructure(std::string dipoleInteractions,
                  int steps, double magneticField,
                  double temperature, double macrocell_size,
                  std::string output_dir, std::string structure_filename);

    void spinStructureMeasurement(double FeTT,double FeOO, double FeTO, double
        FeOO_APB,
                                  double anisotropyConstant,
                                  double alpha, double beta, double gamma,
                                  double center,
                                  double lattice_a, double lattice_b, double
                                      lattice_c,
                                  double sigma);
};

// wrapper function for M vs. B measurements
void run_MvsB(std::string output_dir,
              std::string structure_filename,
              std::string dipoleInteractions,
              int steps, int numOrientations,
              double temperature,
              double BUpperLimit, double BLowerLimit, double BstepSize,
              double coolingField, int coolingSteps,
              double startTemp, double tempStep,
              double FeTT, double FeOO, double FeTO, double FeOO_APB,
              double anisotropyConstant,
              double macrocell_size,
              double center,
              double lattice_a, double lattice_b, double lattice_c,
              double sigma);

// wrapper function for M vs. T measurements
void run_MvsT(std::string output_dir,
              std::string structure_filename,
              std::string dipoleInteractions,
              double steps, int averaging_steps,
              int numOrientations,
              double measurement_field, double cooling_field,
              double TUpperLimit, double TLowerLimit, double TstepSize,
              double FeTT, double FeOO, double FeTO, double FeOO_APB,
              double anisotropyConstant,
              bool ZFC, bool FC,
              double macrocell_size,
              double center,
              double lattice_a, double lattice_b, double lattice_c,
              double sigma);

// wrapper function for spin structure calculations
void run_spinstructure(std::string dipoleInteractions,
                       int steps,
                       double magneticField,
                       double temperature,
                       std::string output_dir,
                       std::string structure_filename,
                       double FeTT,double FeOO, double FeTO, double FeOO_APB,
                       double anisotropyConstant,
                       double alpha,double beta, double gamma ,
                       double macrocell_size,
                       double center,
                       double lattice_a, double lattice_b, double lattice_c,
                       double sigma);



#endif /* Measurement_hpp */
```

C.7 Measurement.cpp

```
#include "Measurement.hpp"

```

```cpp
// function to generate field and temperature arrays
template<typename T>
std::vector<T> arange(T start, T stop, T step = 1) {
    std::vector<T> values;
    if(start < stop){
        for (T value = start; value < stop; value += step)
        values.push_back(value);
    } else{
        for (T value = start; value > stop; value -= step)
            values.push_back(value);
    }
    return values;
}

// helper function to convert doubles to string for filenames
std::string num_to_string(double number, int precision){
    std::string double_string = std::to_string((int)(number))+"d";
    double multiplier = 10.0;
    int A = 0;
    for(int i=0; i<precision; i++){
        A = (int)(number*multiplier - (int)(number*multiplier/10.0)*10);
        double_string += std::to_string(A);
        multiplier *= 10.0;
    }
    return double_string;
}




MvsTMeasurement::MvsTMeasurement(std::string dipoleInteractions,
                                 double steps,
                                 int averaging_steps,
                                 int numOrientations,
                                 double measurement_field,
                                 double cooling_field,
                                 double TUpperLimit,
                                 double TLowerLimit,
                                 double TstepSize,
                                 double macrocell_size):
dipoleInteractions(dipoleInteractions),
steps(steps),
averaging_steps(averaging_steps),
numOrientations(numOrientations),
measurement_field(measurement_field),
cooling_field(cooling_field),
TUpperLimit(TUpperLimit),
TLowerLimit(TLowerLimit),
TstepSize(TstepSize),
macrocell_size(macrocell_size){
}

void MvsTMeasurement::temperatureSweep(std::string output_dir, std::string
    structure_filename,
                                       double FeTT, double FeOO, double FeTO,
                                           double FeOO_APB,
                                       double anisotropyConstant,
                                       bool ZFC, bool FC,
                                       double center,
                                       double lattice_a, double lattice_b, double
                                            lattice_c,
                                       double sigma){

    std::string output_filename = output_dir;
    output_filename += structure_filename.substr(structure_filename.find_last_of(
        "/")+1);

    output_filename += "_MvsT_sim_cF" + num_to_string(cooling_field, 2);
    output_filename += "T_mF" + num_to_string(measurement_field, 3);
    output_filename += "T_" + num_to_string(steps, 1) + "steps";
    output_filename += "_" + std::to_string((int)averaging_steps) + "avsteps";
    output_filename += "_" + std::to_string((int)numOrientations) + "or";
    output_filename += "_" + num_to_string(sigma, 2) + "sig";

    if(dipoleInteractions == "macrocell_method"){
        output_filename += "_0d" + std::to_string((int)(macrocell_size*100)) + "
            mcsize";
    }

    auto temperature_arr_down = arange<double>(TUpperLimit,TLowerLimit,TstepSize)
        ;
    auto temperature_arr_up = arange<double>(TLowerLimit,TUpperLimit,TstepSize);

    for(int i=0; i< temperature_arr_up.size();i++){
        mxZFC.push_back(0.0);
        myZFC.push_back(0.0);
        mzZFC.push_back(0.0);
    }
    for(int i=0; i< temperature_arr_down.size();i++){
        mxFC.push_back(0.0);
        myFC.push_back(0.0);
        mzFC.push_back(0.0);
    }

    ProgressBar bar;
    bar.set_bar_width(50);
    bar.fill_bar_progress_with("   ");
    bar.fill_bar_remainder_with(" ");
    bar.update(0);

    Crystal crystal(structure_filename,
                    dipoleInteractions,
                    FeTT, FeOO, FeTO, FeOO_APB,
                    anisotropyConstant,
                    0.0, 0.0, 0.0, // orientation angles
                    macrocell_size, center,
                    lattice_a, lattice_b, lattice_c, sigma);

    for(int i = 0; i < numOrientations; i++){
        bar.update((double)(i)/(double)(numOrientations));
        bar.set_status_text("orientation " + std::to_string(i+1));

        // set random particle orientation
        crystal.random_orientation();
        // set spin structure fully aligned along random field
        crystal.align_along_random_vector();

        int totalNumAtoms = (int)crystal.atoms.size();

        if(ZFC){
            for(int j=0; j<temperature_arr_up.size(); j++){

                // relaxation steps
                for(int k = 0; k<(int)(steps*totalNumAtoms); k++){
                    crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].
                        MonteCarloStep(
                                measurement_field,temperature_arr_up[j]);
                }

                double mx_temp = 0.0;
                double my_temp = 0.0;
                double mz_temp = 0.0;

                // averaging steps
                if(averaging_steps == 0){
                    for(int a=0; a<(int)crystal.atoms.size(); a++){
                        mx_temp += MAGFE3*crystal.atoms[a].spinx;
                        my_temp += MAGFE3*crystal.atoms[a].spiny;
                        mz_temp += MAGFE3*crystal.atoms[a].spinz;
                    }
                    mxZFC[j] += (mx_temp/(double)(totalNumAtoms))/(double)numOrientations;
                    myZFC[j] += (my_temp/(double)(totalNumAtoms))/(double)numOrientations;
                    mzZFC[j] += (mz_temp/(double)(totalNumAtoms))/(double)numOrientations;
                } else{
                    for(int m=0; m<averaging_steps*totalNumAtoms; m++){
                        crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].MonteCarloStep(
                                                        measurement_field, temperature_arr_up[j]);

                        for(int a=0; a<(int)crystal.atoms.size(); a++){
                            mx_temp += MAGFE3*crystal.atoms[a].spinx;
                            my_temp += MAGFE3*crystal.atoms[a].spiny;
                            mz_temp += MAGFE3*crystal.atoms[a].spinz;
                        }
                    }
                    mxZFC[j] += (mx_temp/(double)(averaging_steps*totalNumAtoms))/(double)numOrientations;
                    myZFC[j] += (my_temp/(double)(averaging_steps*totalNumAtoms))/(double)numOrientations;
                    mzZFC[j] += (mz_temp/(double)(averaging_steps*totalNumAtoms))/(double)numOrientations;
                } // end of averaging steps
            } // end of ZFC measurement

            // write parameters and results to file
            std::ofstream tSweepZfc;
            tSweepZfc.open (output_filename + "_ZFC.txt", std::fstream::out);
            tSweepZfc << "# structure_file: " << structure_filename << std::endl;
            tSweepZfc << "# dipole_interactions: " << dipoleInteractions << std::endl;
            tSweepZfc << "# steps: " << steps << std::endl;
            tSweepZfc << "# num_orientations: " << numOrientations  << std::endl;
            tSweepZfc << "# measurement_field: " << measurement_field << std::endl;
            tSweepZfc << "# cooling_field: " << cooling_field<< std::endl;
            tSweepZfc << "# TUpperLimit: " << TUpperLimit << std::endl;
            tSweepZfc << "# TLowerLimit: " << TLowerLimit << std::endl;
            tSweepZfc << "# TstepSize: " << TstepSize << std::endl;
            tSweepZfc << "# FeTT: " << FeTT << std::endl;
            tSweepZfc << "# FeTO: " << FeTO << std::endl;
            tSweepZfc << "# FeOO: " << FeOO << std::endl;
            tSweepZfc << "# FeOO_APB: " << FeOO_APB << std::endl;
            tSweepZfc << "# lattice parameters: " << lattice_a << ", " << lattice_b << ", "
            << lattice_c << std::endl;
            tSweepZfc << "# sigma: " << sigma << std::endl;
            for(int currentStep = 0; currentStep < temperature_arr_up.size(); currentStep++){
                tSweepZfc << temperature_arr_up[currentStep] << " "
                          << mxZFC[currentStep] << " "
                          << myZFC[currentStep] << " "
                          << mzZFC[currentStep] << "\n";
            }
        } // end of ZFC

        if(FC){
            for(int j=0; j<temperature_arr_down.size(); j++){

                // relaxation steps
                for(int k = 0; k< (int)(steps*totalNumAtoms); k++){
                    crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].
                        MonteCarloStep(
                                                measurement_field,
                                                    temperature_arr_down[j]);
                }

                double mx_temp_fc = 0.0;
                double my_temp_fc = 0.0;
                double mz_temp_fc = 0.0;

                // averaging steps
                if(averaging_steps == 0){
                    for(int a=0; a<(int)crystal.atoms.size(); a++){
                        mx_temp_fc += MAGFE3*crystal.atoms[a].spinx;
                        my_temp_fc += MAGFE3*crystal.atoms[a].spiny;
                        mz_temp_fc += MAGFE3*crystal.atoms[a].spinz;
                    }
                    mxFC[j] += (mx_temp_fc/(double)(totalNumAtoms))/(double)
                        numOrientations;
                    myFC[j] += (my_temp_fc/(double)(totalNumAtoms))/(double)
                        numOrientations;
                    mzFC[j] += (mz_temp_fc/(double)(totalNumAtoms))/(double)
                        numOrientations;
                } else{
                    for(int m=0; m<averaging_steps*totalNumAtoms; m++){
                        crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].
                            MonteCarloStep(
                                                    measurement_field,
                                                        temperature_arr_down[j]);
                        for(int a=0; a<(int)crystal.atoms.size(); a++){
                            mx_temp_fc += MAGFE3*crystal.atoms[a].spinx;
                            my_temp_fc += MAGFE3*crystal.atoms[a].spiny;
                            mz_temp_fc += MAGFE3*crystal.atoms[a].spinz;
                        }
                    }
                    mxFC[j] += (mx_temp_fc/(double)(averaging_steps*totalNumAtoms
                        ))/
                    (double)numOrientations;
                    myFC[j] += (my_temp_fc/(double)(averaging_steps*totalNumAtoms
                        ))/
                    (double)numOrientations;
                    mzFC[j] += (mz_temp_fc/(double)(averaging_steps*totalNumAtoms
                        ))/
                    (double)numOrientations;
                } // end of averaging steps
            } // end of FC measurement

            // write results and parameters to file
            std::ofstream tSweepFc;
            tSweepFc.open (output_filename + "_FC.txt", std::fstream::out);
            tSweepFc << "# structure_file: " << structure_filename << std::endl;
            tSweepFc << "# dipole_interactions: " << dipoleInteractions << std::
                endl;
            tSweepFc << "# steps: " << steps << std::endl;
            tSweepFc << "# num_orientations: " << numOrientations  << std::endl;
            tSweepFc << "# measurement_field: " << measurement_field << std::endl
                ;
            tSweepFc << "# cooling_field: " << cooling_field<< std::endl;
            tSweepFc << "# TUpperLimit: " << TUpperLimit << std::endl;
            tSweepFc << "# TLowerLimit: " << TLowerLimit << std::endl;
            tSweepFc << "# TstepSize: " << TstepSize << std::endl;
            tSweepFc << "# FeTT: " << FeTT << std::endl;
            tSweepFc << "# FeTO: " << FeTO << std::endl;
            tSweepFc << "# FeOO: " << FeOO << std::endl;
            tSweepFc << "# FeOO_APB: " << FeOO_APB << std::endl;
            tSweepFc << "# sigma: " << sigma << std::endl;
            tSweepFc << "# lattice parameters: " << lattice_a << ", " <<
                lattice_b << ", "
            << lattice_c << std::endl;
            tSweepFc << "# " << std::endl;
            tSweepFc << "# field (T)      M_x(B)      M_y(B)      M_z(B)" << std
                ::endl;
            for(int currentStep = 0; currentStep < temperature_arr_down.size();
                currentStep++){
                tSweepFc << temperature_arr_down[currentStep] << " "
                         << mxFC[currentStep] << " "
                         << myFC[currentStep] << " "
                         << mzFC[currentStep] << "\n";
            }
        } // end of FC
    } // end of orientation
} // end of temperature sweep

void run_MvsT(std::string output_dir, std::string structure_filename,
              std::string dipoleInteractions,
              double steps, int averaging_steps, int numOrientations,
              double measurement_field,
              double cooling_field, double TUpperLimit,
              double TLowerLimit, double TstepSize,
              double FeTT, double FeOO, double FeTO,
              double FeOO_APB, double anisotropyConstant,
              bool ZFC, bool FC, double macrocell_size, double center,
              double lattice_a,double lattice_b,
              double lattice_c, double sigma){

    MvsTMeasurement MvsTMeasurement(dipoleInteractions,
                                    steps,
                                    averaging_steps,
                                    numOrientations,
                                    measurement_field,
                                    cooling_field,
                                    TUpperLimit,
                                    TLowerLimit,
                                    TstepSize, macrocell_size);
    MvsTMeasurement.temperatureSweep(output_dir,structure_filename,
                                      FeTT,  FeOO,  FeTO,  FeOO_APB,
                                      anisotropyConstant, ZFC, FC, center,
                                     lattice_a, lattice_b, lattice_c,
                                     sigma);
}


MvsBMeasurement::MvsBMeasurement(std::string dipoleInteractions,
                                 int steps,
                                 int numOrientations,
                                 double temperature,
                                 double BUpperLimit,
                                 double BLowerLimit,
                                 double BstepSize,
                                 double startTemp,
                                 double tempStep,
                                 double coolingField,
                                 int coolingSteps, double macrocell_size):
dipoleInteractions(dipoleInteractions),
steps(steps),
numOrientations(numOrientations),
temperature(temperature),
BUpperLimit(BUpperLimit), BLowerLimit(BLowerLimit), BstepSize(BstepSize),
startTemp(startTemp), tempStep(tempStep),
coolingField(coolingField), coolingSteps(coolingSteps),
macrocell_size(macrocell_size) {

    // generate field arrays
    BnumberOfSteps = (BUpperLimit/BstepSize)+2*((BUpperLimit-BLowerLimit)/BstepSize);
    int BstepsUpStart = BUpperLimit/BstepSize;
    int BstepsDown = (BUpperLimit - BLowerLimit)/BstepSize;
    for(int i = 0; i<BstepsUpStart; i++){
        magneticField.push_back(0.0 + i*BstepSize);
    }
    for(int i = 0; i < BstepsDown; i++){
        magneticField.push_back(BUpperLimit - i*BstepSize);
    }
    for(int i = 0; i < BstepsDown; i++){
        magneticField.push_back(BLowerLimit + i*BstepSize);
    }
    magneticField.push_back(BUpperLimit);

    // generate magnetization vectors for each field step
    for(int i = 0; i <= BnumberOfSteps; i++){
        mX.push_back(0.0);
        mY.push_back(0.0);
        mZ.push_back(0.0);
    }
}

void MvsBMeasurement::fieldSweep(std::string output_dir,
                                 std::string structure_filename,
                                 double FeTT,double FeOO, double FeTO, double FeOO_APB,
                                 double anisotropyConstant,
                                 double lattice_a, double lattice_b, double lattice_c,
                                 double center, double sigma){

    std::string output_filename = structure_filename.substr(
                                 structure_filename.find_last_of("/")+1);
    output_filename += "_Hysteresis_sim_"+num_to_string(temperature, 1);
    output_filename += "K_"+std::to_string((int)(steps))+"steps";
    output_filename += "_"+std::to_string((int)(numOrientations))+"or";
    output_filename += "_"+num_to_string(coolingField, 2)+"T";
    output_filename += "_"+num_to_string(coolingSteps, 2)+"cs";
    output_filename += "_sT"+num_to_string(startTemp, 1)+"K";
    output_filename += "_dip"+dipoleInteractions;
    if(dipoleInteractions == "macrocell_method"){
        output_filename += "_0d" + std::to_string((int)(macrocell_size*100)) + "mcsize";
    }

    std::cout << "\nOutput filename: " << output_filename << std::endl;
    std::cout << "\n ------- starting M vs B measurement -------\n" << std::endl;

    output_filename = output_dir + output_filename;

    for(int i=0; i <= BnumberOfSteps; i++){
        mX[i] = 0.0;
        mY[i] = 0.0;
        mZ[i] = 0.0;
    }

    ProgressBar bar;
    bar.set_bar_width(50);
    bar.fill_bar_progress_with("   ");
    bar.fill_bar_remainder_with(" ");
    bar.update(0);

    for(int i = 0; i < numOrientations; i++){
        bar.update((double)(i)/(double)(numOrientations));
        bar.set_status_text("orientation " + std::to_string(i+1));

        double angles[3];
        rand0_360(angles);

        Crystal crystal(structure_filename, dipoleInteractions,
                        FeTT, FeOO, FeTO, FeOO_APB,
                        anisotropyConstant,
                        angles[0], angles[1], angles[2],
                        macrocell_size, center,
                        lattice_a, lattice_b, lattice_c,
                        sigma);

        int totalNumAtoms = int(crystal.atoms.size());

        // (field) cooling
        auto temperature_arr = arange<double>(startTemp,temperature,tempStep);
        for(int i=0; i<temperature_arr.size();i++){
            for(int j=0; j<coolingSteps * totalNumAtoms; j++){
                crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].MonteCarloStep(
                                                        coolingField,
                                                            temperature_arr[i]);
            }
        }

        // field sweep
        for(int j=0; j<=BnumberOfSteps; j++){
            for(int k = 0; k<steps*totalNumAtoms; k++){
                crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].MonteCarloStep(
                                                        magneticField[j],
                                                            temperature);
            }
            for(unsigned int l=0; l<=crystal.atoms.size(); l++){
                mX[j] += MAGFE3*crystal.atoms[l].spinx;
                mY[j] += MAGFE3*crystal.atoms[l].spiny;
                mZ[j] += MAGFE3*crystal.atoms[l].spinz;
            }
        }

        std::ofstream bSweep;
        bSweep.open (output_filename + ".txt", std::fstream::out);
        bSweep << "# structure_file: " << structure_filename << std::endl;
        bSweep << "# dipole_interactions: " << dipoleInteractions << std::endl;
        bSweep << "# steps: " << steps << std::endl;
        bSweep << "# num_orientations: " << numOrientations  << std::endl;
        bSweep << "# temperature: " << temperature << std::endl;
        bSweep << "# B_upper: " << BUpperLimit<< std::endl;
        bSweep << "# B_lower: " << BLowerLimit << std::endl;
        bSweep << "# B_step: " << BstepSize << std::endl;
        bSweep << "# cooling_field: " << coolingField << std::endl;
        bSweep << "# start_temperature: " << startTemp << std::endl;
        bSweep << "# temperature_step: " << tempStep << std::endl;
        bSweep << "# FeTT: " << FeTT << std::endl;
        bSweep << "# FeTO: " << FeTO << std::endl;
        bSweep << "# FeOO: " << FeOO << std::endl;
        bSweep << "# FeOO_APB: " << FeOO_APB << std::endl;
        bSweep << "# lattice parameters: " << lattice_a << ", " << lattice_b << "
            , "
        << lattice_c << std::endl;
        bSweep << "# " << std::endl;
        bSweep << "# field (T)      M_x(B)      M_y(B)      M_z(B)" << std::endl;
        for(int currentStep = 0; currentStep <= BnumberOfSteps; currentStep++){
            bSweep << magneticField[currentStep] << " "
                   << mX[currentStep] << " "
                   << mY[currentStep] << " "
                   << mZ[currentStep] << "\n";
        }
    } // end of field sweep for one particle orientation
} // end of field sweep

void run_MvsB(std::string output_dir,
              std::string structure_filename,
              std::string dipoleInteractions,
              int steps, int numOrientations,
              double temperature,
              double BUpperLimit, double BLowerLimit, double BstepSize,
              double coolingField, int coolingSteps, double startTemp, double tempStep,
              double FeTT, double FeOO, double FeTO, double FeOO_APB,
              double anisotropyConstant,
              double macrocell_size,
              double lattice_a, double lattice_b, double lattice_c,
              double center, double sigma){
    MvsBMeasurement MvsBMeasurement(dipoleInteractions,
                                    steps,
                                    numOrientations,
                                    temperature,
                                    BUpperLimit,
                                    BLowerLimit,
                                    BstepSize,
                                    startTemp,
                                    tempStep,
                                    coolingField,
                                    coolingSteps,
                                    macrocell_size);
    MvsBMeasurement.fieldSweep(output_dir, structure_filename,
                               FeTT,  FeOO,  FeTO,  FeOO_APB,
                               anisotropyConstant,
                               lattice_a,  lattice_b,  lattice_c,
                               center, sigma);
}

spinStructure::spinStructure(std::string dipoleInteractions,
                             int steps,
                             double magneticField,
                             double temperature,
                             double macrocell_size,
                             std::string output_dir,
                             std::string structure_filename):
                             dipoleInteractions(dipoleInteractions),
                             steps(steps),
                             magneticField(magneticField),
                             temperature(temperature),
                             macrocell_size(macrocell_size),
                             output_dir(output_dir),
                             structure_filename(structure_filename){
};

void spinStructure::spinStructureMeasurement(double FeTT,double FeOO, double FeTO,
                                             double FeOO_APB,
                                             double anisotropyConstant,
                                             double alpha, double beta, double gamma,
                                             double center,
                                             double lattice_a, double lattice_b,
                                             double lattice_c,
                                             double sigma){
    Crystal crystal(structure_filename, dipoleInteractions,
                    FeTT, FeOO, FeTO, FeOO_APB,
                    anisotropyConstant,
                    alpha, beta, gamma,
                    macrocell_size, center,
                    lattice_a, lattice_b, lattice_c,
                    sigma);

    int totalNumAtoms = int(crystal.atoms.size());

    ProgressBar bar;
    bar.set_bar_width(50);
    bar.fill_bar_progress_with("   ");
    bar.fill_bar_remainder_with(" ");
    bar.update(0);

    {
        for(int i =0; i < steps*totalNumAtoms; i++){
            crystal.atoms[rand0_crystalAtoms(totalNumAtoms)].MonteCarloStep(
                                                    magneticField,temperature);
            if(i % 100000 == 0){
                bar.update((double)(i)/(double)(steps*totalNumAtoms));
                bar.set_status_text("trial move " + std::to_string(i));
            }
        }


        std::string output_filename = output_dir;
        output_filename += structure_filename.substr(structure_filename.
            find_last_of("/")+1);
        output_filename += "_spin_structure_"+std::to_string((int)(temperature));
        output_filename += "K_"+std::to_string((int)(steps))+"MCS";
        output_filename += "_"+num_to_string(magneticField, 2)+"T";
        output_filename += "_dip"+dipoleInteractions;
        if(dipoleInteractions == "macrocell_method"){
            output_filename += "_0d" + std::to_string((int)(macrocell_size*100))
                + "mcsize";
        }

        std::ofstream structure;
        structure.open(output_filename + ".txt", std::fstream::out);

        structure << "# structure_file: " << structure_filename << std::endl;
        structure << "# dipole_interactions: " << dipoleInteractions << std::endl
            ;
        if(dipoleInteractions == "macrocell_method"){
            structure << "# macrocell_size: " << macrocell_size << std::endl;
        }
        structure << "# steps: " << steps << std::endl;
        structure << "# Number of atoms: " << totalNumAtoms << std::endl;
        structure << "# temperature: " << temperature << std::endl;
        structure << "# field: " << magneticField << std::endl;
        structure << "# FeTT: " << FeTT << std::endl;
        structure << "# FeTO: " << FeTO << std::endl;
        structure << "# FeOO: " << FeOO << std::endl;
        structure << "# FeOO_APB: " << FeOO_APB << std::endl;
        structure << "# anisotropy constant: " << anisotropyConstant << std::endl
            ;
        structure << "# lattice parameters: " << lattice_a << ", " << lattice_b
            << ", "
        << lattice_c << std::endl;
        structure << "# Orientation: " << alpha << ", " << beta << ", " << gamma
            << std::endl;
        structure << "# " << std::endl;
        structure << "# x      y        z      spinx    spiny    spinz  pos  APB"
             << std::endl;

        for(int i=0; i< crystal.atoms.size(); i++){
            structure << crystal.atoms[i].x << ", "
                      << crystal.atoms[i].y << ", "
                      << crystal.atoms[i].z << ", "
                      << crystal.atoms[i].spinx << ", "
                      << crystal.atoms[i].spiny << ", "
                      << crystal.atoms[i].spinz << ", "
                      << crystal.atoms[i].position << ", "
                      << crystal.atoms[i].isApbAtom << "\n";
        }
    } // end of spin structure simulation
}
```

```

void run_spinstructure(std::string dipoleInteractions,
                       int steps, double magneticField,
                       double temperature,
                       std::string output_dir,
                       std::string structure_filename,
                       double FeTT,double FeOO, double FeTO, double FeOO_APB,
                       double anisotropyConstant,
                       double alpha,double beta, double gamma,
                       double macrocell_size, double center,
                       double lattice_a, double lattice_b, double lattice_c,
                       double sigma){
    spinStructure spinStructure(dipoleInteractions,
                                steps,
                                magneticField,
                                temperature,
                                macrocell_size,
                                output_dir,
                                structure_filename
                                );
    spinStructure.spinStructureMeasurement(FeTT, FeOO, FeTO, FeOO_APB,
                                           anisotropyConstant, alpha,
                                           beta, gamma, center,  lattice_a,
                                           lattice_b, lattice_c, sigma);
}
```

C.8 randNumGenerator.hpp

```
#ifndef randNumGenerator_h
#define randNumGenerator_h

#include <cmath>
#include <iostream>

long rnd250();
void seed250(long);
void marsaglia(double*);
double rand0001_0999();
int rand0_crystalSize(int dimension);
double rand0_1();
double rand0_90();
void rand0_360(double*);
int rand0_crystalAtoms(int totalNumAtoms);
double gaussian_marsaglia(double stdDev);
double gaussian_ziggurat();

const int RND250_MAX = 0x7FFFFFFF;

static struct st_rnd250{
    int point;
    long field[256];
} Rnd250;

#endif /* randNumGenerator_h */
```

C.9 randNumGenerator.cpp

```
#include "randNumGenerator.hpp"
/*
 * Original header information for the random number generator:
 *
 * "Zufallszahlengenerator nach dem Verfahren von Kirkpatrick und Stoll.
 * Dieses C-modul enthaelt nur die globale Definition der Daten und die
 * Initialisierungsroutine. Der eignetliche Zufallszahlengenerator ist
 * als Makro in der Datei RND250.H kodiert.
 *
 * Implementation: Ralf Meyer, Fred Hucht
 * Version       : 2.0
 * entwickelt am : 14. Februar 1995
 *
 * Copyright (c) 1995 Ralf Meyer, 47058 Duisburg, Germany"
 *
 * The modifications for this program are the replacement of the macro
 * with the function rnd250() and the conversion of the initialization
 * routine into a function called seed250(long seed).
 */
long rnd250(){
    Rnd250.point = (Rnd250.point+1)&255;
    Rnd250.field[Rnd250.point] =
    Rnd250.field[(Rnd250.point-250)&255]^Rnd250.field[(Rnd250.point-103)&255];
    return Rnd250.field[Rnd250.point];
}

void seed250(long seed){
    int i;
    long j,k;

    if(seed < 1){
        seed = 1;
    }

    for(i = 0; i < 250; ++i){
        k = seed / 127773;
        seed = 16807 * (seed - k * 127773) - 2836 * k;
        if(seed < 0){
            seed += 0x7FFFFFFF;
        }
        Rnd250.field[i] = seed;
    }

    k = 0x7FFFFFFF;
    j = 0x40000000;
    for(i = 1; i < 250; i+= 8){
        Rnd250.field[i] = (Rnd250.field[i] & k) | j;
    }

    Rnd250.point = 249;

    for(i = 0; i < 4711; ++i){
        rnd250();
    }
}

double rand0001_0999(){
    return((rnd250() + 1.0) / (RND250_MAX + 2.0));
}

double gaussian_marsaglia(double stdDev){
    static double spare;
    static bool hasSpare = false;

    if (hasSpare) {
        hasSpare = false;
        return spare * stdDev;
        } else {
            double u, v, s;
            do {
                u = (rand0_1()) * 2.0 - 1.0;
                v = (rand0_1()) * 2.0 - 1.0;
                s = u * u + v * v;
            } while (s >= 1.0 || s == 0.0);
            s = std::sqrt(-2.0 * std::log(s) / s);
            spare = v * s;
            hasSpare = true;
            return stdDev * u * s;
        }
}

void marsaglia(double* V){
    double rsq, y1, y2;

    do{
        y1 = rand0001_0999() * 2.0 - 1.0;
        y2 = rand0001_0999() * 2.0 - 1.0;
        rsq = y1 * y1 + y2 * y2;
    } while (rsq > 1.0);

    V[0] = 2.0 * y1 * std::sqrt((1.0 - rsq));
    V[1] = 2.0 * y2 * std::sqrt((1.0 - rsq));
    V[2] = 1.0 - 2.0 * rsq;
}

int rand0_crystalSize(int dimension){
    return(rnd250() % (dimension*dimension*dimension*24));
}

int rand0_crystalAtoms(int totalNumAtoms){
    return(rnd250() % (totalNumAtoms));
}

double rand0_1() //generate a random number in [0,1]
{
    return ((double)rnd250() / (double)RND250_MAX);
}

double rand0_90(){
    std::cout << (rnd250() % 91) <<std::endl;
    return(rnd250() % 91);
}

void rand0_360(double *angles){
    double V[3];
    marsaglia(V);
    angles[0] = std::acos(V[0])*180.0/M_PI;
    angles[1] = std::acos(V[1])*180.0/M_PI;
    angles[2] = std::acos(V[2])*180.0/M_PI;
}


/* gauss.c - gaussian random numbers, using the Ziggurat method
 *
 * Copyright (C) 2005  Jochen Voss.
 *
 * For details see the following article.
 *
 *     George Marsaglia, Wai Wan Tsang
 *     The Ziggurat Method for Generating Random Variables
 *     Journal of Statistical Software, vol. 5 (2000), no. 8
 *     http://www.jstatsoft.org/v05/i08/
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 * $Id: gauss.c 6739 2005-11-12 02:47:20Z voss $
 */

//#include <math.h>
  //#include <assert.h>

//#include <gsl/gsl_rng.h>


/// position of right-most step
#define PARAM_R 3.44428647676

/// tabulated values for the heigt of the Ziggurat levels
static const double ytab[128] = {
  1, 0.963598623011, 0.936280813353, 0.913041104253,
  0.892278506696, 0.873239356919, 0.855496407634, 0.838778928349,
  0.822902083699, 0.807732738234, 0.793171045519, 0.779139726505,
  0.765577436082, 0.752434456248, 0.739669787677, 0.727249120285,
  0.715143377413, 0.703327646455, 0.691780377035, 0.68048276891,
  0.669418297233, 0.65857233912, 0.647931876189, 0.637485254896,
  0.62722199145, 0.617132611532, 0.607208517467, 0.597441877296,
  0.587825531465, 0.578352913803, 0.569017984198, 0.559815170911,
  0.550739320877, 0.541785656682, 0.532949739145, 0.524227434628,
  0.515614886373, 0.507108489253, 0.498704867478, 0.490400854812,
  0.482193476986, 0.47407993601, 0.466057596125, 0.458123971214,
  0.450276713467, 0.442513603171, 0.434832539473, 0.427231532022,
  0.419708693379, 0.41226223212, 0.404890446548, 0.397591718955,
  0.390364510382, 0.383207355816, 0.376118859788, 0.369097692334,
  0.362142585282, 0.355252328834, 0.348425768415, 0.341661801776,
  0.334959376311, 0.328317486588, 0.321735172063, 0.31521151497,
  0.308745638367, 0.302336704338, 0.29598391232, 0.289686497571,
  0.283443729739, 0.27725491156, 0.271119377649, 0.265036493387,
  0.259005653912, 0.253026283183, 0.247097833139, 0.241219782932,
  0.235391638239, 0.229612930649, 0.223883217122, 0.218202079518,
  0.212569124201, 0.206983981709, 0.201446306496, 0.195955776745,
  0.190512094256, 0.185114984406, 0.179764196185, 0.174459502324,
  0.169200699492, 0.1639876086, 0.158820075195, 0.153697969964,
  0.148621189348, 0.143589656295, 0.138603321143, 0.133662162669,
  0.128766189309, 0.123915440582, 0.119109988745, 0.114349940703,
  0.10963544023, 0.104966670533, 0.100343857232, 0.0957672718266,
  0.0912372357329, 0.0867541250127, 0.082318375932, 0.0779304915295,
  0.0735910494266, 0.0693007111742, 0.065060233529, 0.0608704821745,
  0.056732448584, 0.05264727098, 0.0486162607163, 0.0446409359769,
  0.0407230655415, 0.0368647267386, 0.0330683839378, 0.0293369977411,
  0.0256741818288, 0.0220844372634, 0.0185735200577, 0.0151490552854,
  0.0118216532614, 0.00860719483079, 0.00553245272614, 0.00265435214565
};

/// tabulated values for 2^24 times x[i]/x[i+1],
/// used to accept for U*x[i+1]<=x[i] without any floating point operations
static const unsigned long ktab[128] = {
  0, 12590644, 14272653, 14988939,
  15384584, 15635009, 15807561, 15933577,
  16029594, 16105155, 16166147, 16216399,
  16258508, 16294295, 16325078, 16351831,
  16375291, 16396026, 16414479, 16431002,
  16445880, 16459343, 16471578, 16482744,
  16492970, 16502368, 16511031, 16519039,
  16526459, 16533352, 16539769, 16545755,
  16551348, 16556584, 16561493, 16566101,
  16570433, 16574511, 16578353, 16581977,
  16585398, 16588629, 16591685, 16594575,
  16597311, 16599901, 16602354, 16604679,
  16606881, 16608968, 16610945, 16612818,
  16614592, 16616272, 16617861, 16619363,
  16620782, 16622121, 16623383, 16624570,
  16625685, 16626730, 16627708, 16628619,
  16629465, 16630248, 16630969, 16631628,
  16632228, 16632768, 16633248, 16633671,
  16634034, 16634340, 16634586, 16634774,
  16634903, 16634972, 16634980, 16634926,
  16634810, 16634628, 16634381, 16634066,
  16633680, 16633222, 16632688, 16632075,
  16631380, 16630598, 16629726, 16628757,
  16627686, 16626507, 16625212, 16623794,
  16622243, 16620548, 16618698, 16616679,
  16614476, 16612071, 16609444, 16606571,
  16603425, 16599973, 16596178, 16591995,
  16587369, 16582237, 16576520, 16570120,
  16562917, 16554758, 16545450, 16534739,
  16522287, 16507638, 16490152, 16468907,
  16442518, 16408804, 16364095, 16301683,
  16207738, 16047994, 15704248, 15472926
};

/// tabulated values of 2^{-24}*x[i]
static const double wtab[128] = {
  1.62318314817e-08, 2.16291505214e-08, 2.54246305087e-08, 2.84579525938e-08,
  3.10340022482e-08, 3.33011726243e-08, 3.53439060345e-08, 3.72152672658e-08,
  3.8950989572e-08, 4.05763964764e-08, 4.21101548915e-08, 4.35664624904e-08,
  4.49563968336e-08, 4.62887864029e-08, 4.75707945735e-08, 4.88083237257e-08,
  5.00063025384e-08, 5.11688950428e-08, 5.22996558616e-08, 5.34016475624e-08,
  5.44775307871e-08, 5.55296344581e-08, 5.65600111659e-08, 5.75704813695e-08,
  5.85626690412e-08, 5.95380306862e-08, 6.04978791776e-08, 6.14434034901e-08,
  6.23756851626e-08, 6.32957121259e-08, 6.42043903937e-08, 6.51025540077e-08,
  6.59909735447e-08, 6.68703634341e-08, 6.77413882848e-08, 6.8604668381e-08,
  6.94607844804e-08, 7.03102820203e-08, 7.11536748229e-08, 7.1991448372e-08,
  7.2824062723e-08, 7.36519550992e-08, 7.44755422158e-08, 7.52952223703e-08,
  7.61113773308e-08, 7.69243740467e-08, 7.77345662086e-08, 7.85422956743e-08,
  7.93478937793e-08, 8.01516825471e-08, 8.09539758128e-08, 8.17550802699e-08,
  8.25552964535e-08, 8.33549196661e-08, 8.41542408569e-08, 8.49535474601e-08,
  8.57531242006e-08, 8.65532538723e-08, 8.73542180955e-08, 8.8156298059e-08,
  8.89597752521e-08, 8.97649321908e-08, 9.05720531451e-08, 9.138142487e-08,
  9.21933373471e-08, 9.30080845407e-08, 9.38259651738e-08, 9.46472835298e-08,
  9.54723502847e-08, 9.63014833769e-08, 9.71350089201e-08, 9.79732621669e-08,
  9.88165885297e-08, 9.96653446693e-08, 1.00519899658e-07, 1.0138063623e-07,
  1.02247952126e-07, 1.03122261554e-07, 1.04003996769e-07, 1.04893609795e-07,
  1.05791574313e-07, 1.06698387725e-07, 1.07614573423e-07, 1.08540683296e-07,
  1.09477300508e-07, 1.1042504257e-07, 1.11384564771e-07, 1.12356564007e-07,
  1.13341783071e-07, 1.14341015475e-07, 1.15355110887e-07, 1.16384981291e-07,
  1.17431607977e-07, 1.18496049514e-07, 1.19579450872e-07, 1.20683053909e-07,
  1.21808209468e-07, 1.2295639141e-07, 1.24129212952e-07, 1.25328445797e-07,
  1.26556042658e-07, 1.27814163916e-07, 1.29105209375e-07, 1.30431856341e-07,
  1.31797105598e-07, 1.3320433736e-07, 1.34657379914e-07, 1.36160594606e-07,
  1.37718982103e-07, 1.39338316679e-07, 1.41025317971e-07, 1.42787873535e-07,
  1.44635331499e-07, 1.4657889173e-07, 1.48632138436e-07, 1.50811780719e-07,
  1.53138707402e-07, 1.55639532047e-07, 1.58348931426e-07, 1.61313325908e-07,
  1.64596952856e-07, 1.68292495203e-07, 1.72541128694e-07, 1.77574279496e-07,
  1.83813550477e-07, 1.92166040885e-07, 2.05295471952e-07, 2.22600839893e-07
};

double gaussian_ziggurat(){
  unsigned long  U, sign, i, j;
  double  x, y;

  while (1) {
    U = rnd250();
    i = U & 0x0000007F;         /* 7 bit to choose the step */
    sign = U & 0x00000080;     /* 1 bit for the sign */
    j = U>>8;             /* 24 bit for the x-value */

    x = j*wtab[i];
    if (j < ktab[i])  break;

    if (i<127) {
      double  y0, y1;
      y0 = ytab[i];
      y1 = ytab[i+1];
      y = y1+(y0-y1)*rand0_1();
    } else {
      x = PARAM_R - log(1.0-rand0_1())/PARAM_R;
      y = exp(-PARAM_R*(x-0.5*PARAM_R))*rand0_1();
    }
    if (y < exp(-0.5*x*x))  break;
  }
  return  sign ? x : -x;
}
```

C.10 ProgressBar.hpp

```
#ifndef ProgressBar_h
#define ProgressBar_h

#include <atomic>
#include <mutex>
#include <iostream>
#include <string>

class ProgressBar{
public:
    void set_progress(double value){
        //std::unique_lock<std::mutex> lock{mutex_};
        progress_ = value;
    }

    void set_bar_width(size_t width) {
        std::unique_lock<std::mutex> lock{mutex_};
        bar_width_ = width;
    }

    void fill_bar_progress_with(const std::string& chars) {
        std::unique_lock<std::mutex> lock{mutex_};
        fill_ = chars;
    }

    void fill_bar_remainder_with(const std::string& chars) {
        std::unique_lock<std::mutex> lock{mutex_};
        remainder_ = chars;
    }

    void set_status_text(const std::string& status) {
        std::unique_lock<std::mutex> lock{mutex_};
        status_text_ = status;
    }

    void update(float value, std::ostream &os = std::cout) {
        set_progress(value);
        write_progress(os);
    }

    void write_progress(std::ostream &os = std::cout) {
        std::unique_lock<std::mutex> lock{mutex_};

        // No need to write once progress is 100%
        if (progress_ > 100.0f) return;

```

```cpp
        // move up one line
        os << "\033[A";
        // erase current line
        os << "\33[2K";
        // Move cursor to the first position on the same line and flush
        os << "\r" << std::flush;

        // Start bar
        os << "[";

        const auto completed = static_cast<size_t>(progress_ * static_cast<float>(bar_width_));
        for (size_t i = 0; i < bar_width_; ++i) {
          if (i <= completed)
            os << fill_;
          else
            os << remainder_;
        }

        // End bar
        os << "]";

        // Write progress percentage
        os << " " << std::min(static_cast<size_t>(progress_*100.0), size_t(100)) << "%";

        // Write status text
        os << " " << status_text_;

        os << "\n";
    }

private:
    std::mutex mutex_;
    float progress_{0.0f};
    size_t bar_width_{60};
    std::string fill_{"#"}, remainder_{" "}, status_text_{""};
};

#endif /* ProgressBar_h */
```

C.11 constants.h

```cpp
#ifndef constants_h
#define constants_h

const double PI = 3.14159265358979323846264338;
const double MUB = 9.2740100783e-24; // J/T
const double KB = 1.380649e-23;  // J/K
const double MU0   = 1.25663706212e-6; // N/A^2
const long SEEED = 27052020;
const double MAGFE3 = 5 * MUB;
const double testRotationVectorLength = 2.0;
const double CUTOFFRADIUS = 1.0;

#endif /* constants_h */
```

C.12 generate_json_files.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys
import argparse

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('--measurement', type=str)
    parser.add_argument('--output', type=str)
    parser.add_argument('--structure_path', type=str)
    parser.add_argument('--num_particles', type=int)
    parser.add_argument('--particle_size', type=int)
    parser.add_argument('--meas_field', type=float)
    parser.add_argument('--temperature', type=float)
    parser.add_argument('--dipole', type=str)
    parser.add_argument('--steps', type=float)
    parser.add_argument('--av_steps', type=int)
    parser.add_argument('--num_or', type=int)
    parser.add_argument('--cool_field', type=float)
    parser.add_argument('--Tupper', type=float)
    parser.add_argument('--Tlower', type=float)
    parser.add_argument('--T_step', type=float)
    parser.add_argument('--alpha', type=float)
    parser.add_argument('--beta', type=float)
    parser.add_argument('--gamma', type=float)
    parser.add_argument('--macrocell_size', type=float)
    parser.add_argument('--anisotropy_constant', type=float)
    parser.add_argument('--ZFC', action='store_true')
    parser.add_argument('--FC', action='store_true')
    parser.add_argument('--APB', action='store_true')
    parser.add_argument('--Bupper', type=float)
    parser.add_argument('--Blower', type=float)
    parser.add_argument('--Bstep', type=float)
    parser.add_argument('--start_temperature', type=float)
    parser.add_argument('--cooling_steps', type=float)
    parser.add_argument('--sigma', type=float)
    parser.add_argument('--latticepar', type=str)
    parser.add_argument('--exchangeconstants', type=str)
    parser.add_argument('--APB_constant', type=float)

    args = parser.parse_args()

    measurement = args.measurement
    save_path = args.output
    structure_path = args.structure_path
    num_particles = args.num_particles
    particle_size = args.particle_size
    measurement_field = args.meas_field
    temperature = args.temperature
    dipole = args.dipole
    steps = args.steps
    averaging_steps = args.av_steps
    num_orientations = args.num_or
    cooling_field = args.cool_field
    T_upper = args.Tupper
    T_lower = args.Tlower
    T_stepsize = args.T_step
    ZFC = args.ZFC
    FC = args.FC
    APB = args.APB
    alpha = args.alpha
    beta = args.beta
    gamma = args.gamma
    macrocell_size = args.macrocell_size
    anisotropy_constant= args.anisotropy_constant
    sigma = args.sigma

    exchange_constants = (args.exchangeconstants).split(",")
    Fe_TT = float(exchange_constants[0])
    Fe_OO = float(exchange_constants[1])
    Fe_TO = float(exchange_constants[2])
    APB_constant = args.APB_constant

    latticepars = (args.latticepar).split(",")
    lattice_a = float(latticepars[0])
    lattice_b = float(latticepars[1])
    lattice_c = float(latticepars[2])

    Bupper = args.Bupper
    Blower = args.Blower
    Bstep = args.Bstep
    start_T = args.start_temperature
    cool_steps= args.cooling_steps


    if measurement == "MvB":
        print("\tStarting M vs B simulation\n")
        print("Output path: ",save_path)
        print("Number of particles: ",num_particles)
        print("Number of orientations per particle: ", num_orientations)
        print("Measurement field: ",measurement_field)
        print("Cooling field: ", cooling_field)
        print("Field range: %.1f - %.1f T (%.2f T steps)"%(Blower, Bupper, Bstep)
            )
        print("Monte Carlo steps per temperature step: ", steps)
        print("Averaging Monte Carlo steps: ", averaging_steps)
        print("Sigma (gaussian cone): ", sigma)
        print(" ")
        for i in range(1,num_particles+1):
            if APB:
                file_name = "%d_MvsB_simulation_APB_D%d.json"%(i, particle_size)
                structure_file = structure_path + "D%d_structure_APB_%d"%(
                    particle_size,i)
            else:
                file_name = "%d_MvsB_simulation_noAPB_D%d.json"%(i, particle_size
                    )
                structure_file = structure_path + "D%d_structure_noAPB_%d"%(
                    particle_size,i)

            completeName = save_path+file_name

            f = open(completeName, "w+")
            f.write("{\n")
            f.write('"Measurement": "M vs B",\n')
            f.write('"output_dir": "%s",\n' %(save_path))
            f.write('"structure_file": "%s",\n'%(structure_file))
            f.write('"dipole_interactions": "%s",\n'%(dipole))
            f.write('"steps": %d,\n'%(steps))
            f.write('"num_orientations": %d,\n'%(num_orientations))
            f.write('"temperature": %.4f,\n'%(temperature))
            f.write('"B_upper": %.4f,\n'%(Bupper))
            f.write('"B_lower": %.4f,\n'%(Blower))
            f.write('"B_step": %.4f,\n'%(Bstep))
            f.write('"cooling_field": %.4f,\n'%(cooling_field))
            f.write('"cooling_steps": %d,\n'%(cool_steps))
            f.write('"start_temperature": %.4f,\n'%(start_T))
            f.write('"temperature_step": %.4f,\n'%(T_stepsize))
            f.write('"FeTT": %.2f,\n'%(Fe_TT))
            f.write('"FeOO": %.2f,\n'%(Fe_OO))
            f.write('"FeTO": %.2f,\n'%(Fe_TO))
            if APB:
                f.write('"FeOO_APB": %.2f,\n'%(APB_constant))
            else:
                f.write('"FeOO_APB": 0.0,\n')
            f.write('"anisotropy constant": %.2e,\n'%(anisotropy_constant))
            f.write('"lattice_a": %.4f,\n'%(lattice_a))
            f.write('"lattice_b": %.4f,\n'%(lattice_b))
            f.write('"lattice_c": %.4f,\n'%(lattice_c))
            f.write('"sigma": %.2f,\n'%(sigma))
```

```python
            f.write('"particle center": %.2f\n}'%(particle_size/2.0+0.5))
            f.close()

    if measurement == "MvT":
        print("\tStarting M vs T simulation\n")
        print("Output path: ",save_path)
        print("Number of particles: ",num_particles)
        print("Number of orientations per particle: ", num_orientations)
        print("Measurement field: ",measurement_field)
        print("Cooling field: ", cooling_field)
        print("Temperature range: %.4f - %.4f K (%.4f K steps)"%(T_lower, T_upper, T_stepsize))
        print("Monte Carlo steps per temperature step: ", steps)
        print("Averaging Monte Carlo steps: ", averaging_steps)
        print("Sigma (gaussian cone): ", sigma)
        Meas_modes ="Measurements: "
        if ZFC:
            Meas_modes += "ZFC"
        if FC:
            Meas_modes += ", FC"
        print(Meas_modes)
        print(" ")

        for i in range(1,num_particles+1):
            if APB:
                file_name = "%d_MvsT_simulation_APB_D%d.json"%(i, particle_size)
                structure_file = structure_path + "D%d_structure_APB_%d"%(particle_size,i)
            else:
                file_name = "%d_MvsT_simulation_noAPB_D%d.json"%(i, particle_size)
                structure_file = structure_path + "D%d_structure_noAPB_%d"%(particle_size,i)

            completeName = save_path+file_name

            f = open(completeName, "w+")
            f.write("{\n")
            f.write('"Measurement": "M vs T",\n')
            f.write('"output_dir": "%s",\n' %(save_path))
            f.write('"structure_file": "%s",\n'%(structure_file))
            f.write('"dipole_interactions": "%s",\n'%(dipole))
            f.write('"steps": %.2f,\n'%(steps))
            f.write('"averaging steps": %d,\n'%(averaging_steps))
            f.write('"num_orientations": %d,\n'%(num_orientations))
            f.write('"measurement_field": %.4f,\n'%(measurement_field))
            f.write('"cooling_field": %.4f,\n'%(cooling_field))
            f.write('"TUpperLimit": %.4f,\n'%(T_upper))
            f.write('"TLowerLimit": %.4f,\n'%(T_lower))
            f.write('"TstepSize": %.4f,\n'%(T_stepsize))
            f.write('"FeTT": %.2f,\n'%(Fe_TT))
            f.write('"FeOO": %.2f,\n'%(Fe_OO))
            f.write('"FeTO": %.2f,\n'%(Fe_TO))
            if APB:
                f.write('"FeOO_APB": %.2f,\n'%(APB_constant))
            else:
                f.write('"FeOO_APB": 0.0,\n')
            f.write('"anisotropy constant": %.2e,\n'%(anisotropy_constant))
            if ZFC:
                f.write('"ZFC": true,\n')
            else:
                f.write('"ZFC": false,\n')
            if FC:
                f.write('"FC": true,\n')
            else:
                f.write('"FC": false,\n')
            f.write('"particle center": %.2f,\n'%(particle_size/2.0+0.5))
            f.write('"lattice_a": %.4f,\n'%(lattice_a))
            f.write('"lattice_b": %.4f,\n'%(lattice_b))
            f.write('"lattice_c": %.4f,\n'%(lattice_c))
            f.write('"sigma": %.2f\n}'%(sigma))
            f.close()

    if measurement == "spin_structure":
        print("\tStarting spin structure simulation\n")
        print("Output path: ",save_path)
        print("Number of particles: ",num_particles)
        print("Particle orientation angles: %.1f  , %.1f  , %.1f  "%(alpha, beta,
             gamma))
        print("Measurement field: ",measurement_field)
        print("Temperature: ", temperature)
        print("Monte Carlo steps: ", steps)
        print("Sigma (gaussian cone): ", sigma)
        print("Dipole interactions: ", dipole)
        print(" ")

        if APB:
            file_name = "spin_structure_APB_D%d_template.json"%(particle_size)
            structure_file = structure_path + "D%d_structure_APB_template"%(
                particle_size)
        else:
            file_name = "spin_structure_noAPB_D%d_template.json"%(particle_size)
            structure_file = structure_path + "D%d_structure_noAPB_template"%(
                particle_size)

        completeName = save_path+file_name

        f = open(completeName, "w+")
        f.write("{\n")
        f.write('"Measurement": "spin structure",\n')
        f.write('"output_dir": "%s",\n' %(save_path))
        f.write('"structure_file": "%s",\n'%(structure_file))
        f.write('"dipole_interactions": "None",\n')
        f.write('"steps": %d,\n'%(steps))
        f.write('"magnetic field": %.4f,\n'%(measurement_field))
        f.write('"temperature": %.4f,\n'%(temperature))
        f.write('"FeTT": %.2f,\n'%(Fe_TT))
        f.write('"FeOO": %.2f,\n'%(Fe_OO))
        f.write('"FeTO": %.2f,\n'%(Fe_TO))
        if APB:
            f.write('"FeOO_APB": %.2f,\n'%(APB_constant))
        else:
            f.write('"FeOO_APB": 0.0,\n')
        f.write('"anisotropy constant": %.2e,\n'%(anisotropy_constant))
        #f.write('"anisotropy constant": %.2e,\n'%(anisotropy_constant))
        f.write('"alpha": %.1f,\n'%(alpha))
        f.write('"beta": %.1f,\n'%(beta))
        f.write('"gamma": %.1f,\n'%(gamma))
        f.write('"macrocell_size": %.2f,\n'%(macrocell_size))
        f.write('"particle center": %.2f,\n'%(particle_size/2.0+0.5))
        f.write('"lattice_a": %.4f,\n'%(lattice_a))
        f.write('"lattice_b": %.4f,\n'%(lattice_b))
        f.write('"lattice_c": %.4f,\n'%(lattice_c))
        f.write('"sigma": %.2f\n}'%(sigma))
        f.close()

        for i in range(1,num_particles+1):
            if APB:
                file_name = "%d_spin_structure_APB_D%d.json"%(i, particle_size)
                structure_file = structure_path + "D%d_structure_APB_%d"%(
                    particle_size,i)
            else:
                file_name = "%d_spin_structure_noAPB_D%d.json"%(i, particle_size)
                structure_file = structure_path + "D%d_structure_noAPB_%d"%(
                    particle_size,i)

            completeName = save_path+file_name

            f = open(completeName, "w+")
            f.write("{\n")
            f.write('"Measurement": "spin structure",\n')
            f.write('"output_dir": "%s",\n' %(save_path))
            f.write('"structure_file": "%s",\n'%(structure_file))
            f.write('"dipole_interactions": "%s",\n'%(dipole))
            f.write('"steps": %d,\n'%(steps))
            f.write('"magnetic field": %.4f,\n'%(measurement_field))
            f.write('"temperature": %.4f,\n'%(temperature))
            f.write('"FeTT": %.2f,\n'%(Fe_TT))
            f.write('"FeOO": %.2f,\n'%(Fe_OO))
            f.write('"FeTO": %.2f,\n'%(Fe_TO))
            if APB:
                f.write('"FeOO_APB": %.2f,\n'%(APB_constant))
            else:
                f.write('"FeOO_APB": 0.0,\n')
            f.write('"anisotropy constant": %.2e,\n'%(anisotropy_constant))
            f.write('"alpha": %.1f,\n'%(alpha))
            f.write('"beta": %.1f,\n'%(beta))
            f.write('"gamma": %.1f,\n'%(gamma))
            f.write('"macrocell_size": %.2f,\n'%(macrocell_size))
            f.write('"particle center": %.2f,\n'%(particle_size/2.0+0.5))
            f.write('"lattice_a": %.4f,\n'%(lattice_a))
            f.write('"lattice_b": %.4f,\n'%(lattice_b))
            f.write('"lattice_c": %.4f,\n'%(lattice_c))
            f.write('"sigma": %.2f\n}'%(sigma))
            f.close()


if __name__ == "__main__":
    main()
```

C.13 generate_nanoparticle.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import crystal
import parsers
import numpy as np
import sys
import argparse

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('--output', type=str)
    parser.add_argument('--occ_oct', type=float)
    parser.add_argument('--occ_tet', type=float)
    parser.add_argument('--diameter', type=int)
    parser.add_argument('--number', type=int)
    parser.add_argument('--seed', type=int)
    parser.add_argument('--APB', action="store_true")
    parser.add_argument('--template', action="store_true")
    parser.add_argument('--ciffile', type=str)
    parser.add_argument('--atomlabels', type=str)
    parser.add_argument('--occupancies', type=str)
    parser.add_argument('--shape', type=str)
    parser.add_argument('--latticepar', type=str)

    args = parser.parse_args()

    cif_file = args.ciffile
    shape = "Sphere"
    shape = args.shape
    n_APBs = 1
    offset = np.array([0.5,0.5,0.5])
    gradient = None

    APB = args.APB
    template = args.template
    path = args.output
    occ_oct = args.occ_oct
    occ_tet = args.occ_tet
    diameter = args.diameter
    number = args.number
    seed = args.seed

    np.random.seed(seed)
    rands = np.random.randint(low=16430104, high=20210503, size=100)

    labels = (args.atomlabels).split(",")
    occs = (args.occupancies).split(",")
    occupancies = {}
    for l,o in zip(labels,occs):
        occupancies[l] = float(o)

    latticepars = (args.latticepar).split(",")

    if occupancies == {}:
        occ = False
    else:
        occ = True

    saveStructure = path + "D%d_structure_" %(diameter)

    if APB:
        saveStructure += "APB"
    else:
        saveStructure += "noAPB"


    if template:
        saveStructure += "_template"
        SEED = rands[0]
    else:
        saveStructure += "_%d"%(number)
        SEED = rands[number]

    unitcell = parsers.CifParser(cif_file)
    unitcell.lattice_a = float(latticepars[0])
    unitcell.lattice_b = float(latticepars[1])
    unitcell.lattice_c = float(latticepars[2])

    Crystal = crystal.Crystal(diameter, unitcell, occupancies, shape, n_APBs,
        offset)

    Crystal.build_nanoparticle(APB = APB,
                               occ = occ,
                               gradient = gradient,
                               plot = False,
                               SEED = SEED)

    Crystal.output_crystal_structure(saveStructure, oxygen=False)

    print("----------------------------\n")

if __name__ == "__main__":
    main()
```

C.14 3dplot_MC_Mayavi.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Jan  5 11:57:04 2021

@author: tobiaskohler

Running of this script requires MayaVi installed in python 3.7 environment.
The script has to be executed in this environment.
"""

import numpy as np
from mayavi import mlab
import argparse
import os

parser = argparse.ArgumentParser()

parser.add_argument('--input_dir', type=str)
parser.add_argument('--file', type=str)
parser.add_argument('--delimiter', type=str)

args = parser.parse_args()

file = args.file
input_dir = args.input_dir

if args.delimiter == None:
    delimiter=" "
else:
    delimiter=args.delimiter

sfile = input_dir+file

file_r = os.path.splitext(sfile)[0]

def plot_mayavi(spins, size, x,y,z,u,v,w,t,a):

    # setting up the data arrays
    # octahedral iron positions
    x_oct = np.array([xi for xi,ti,ai in zip(x,t,a) if ti == 0 and ai == 0])
    y_oct = np.array([yi for yi,ti,ai in zip(y,t,a) if ti == 0 and ai == 0])
    z_oct = np.array([zi for zi,ti,ai in zip(z,t,a) if ti == 0 and ai == 0])
    u_oct = np.array([ui for ui,ti,ai in zip(u,t,a) if ti == 0 and ai == 0])
    v_oct = np.array([vi for vi,ti,ai in zip(v,t,a) if ti == 0 and ai == 0])
    w_oct = np.array([wi for wi,ti,ai in zip(w,t,a) if ti == 0 and ai == 0])

    # tetrahedral iron positions
    x_tet = np.array([xi for xi,ti,ai in zip(x,t,a) if ti == 1 and ai == 0])
    y_tet = np.array([yi for yi,ti,ai in zip(y,t,a) if ti == 1 and ai == 0])
    z_tet = np.array([zi for zi,ti,ai in zip(z,t,a) if ti == 1 and ai == 0])
    u_tet = np.array([ui for ui,ti,ai in zip(u,t,a) if ti == 1 and ai == 0])
    v_tet = np.array([vi for vi,ti,ai in zip(v,t,a) if ti == 1 and ai == 0])
    w_tet = np.array([wi for wi,ti,ai in zip(w,t,a) if ti == 1 and ai == 0])

    # APB positions
    x_apb = np.array([xi for xi,ai in zip(x,a) if ai == 1])
    y_apb = np.array([yi for yi,ai in zip(y,a) if ai == 1])
    z_apb = np.array([zi for zi,ai in zip(z,a) if ai == 1])
    u_apb = np.array([ui for ui,ai in zip(u,a) if ai == 1])
    v_apb = np.array([vi for vi,ai in zip(v,a) if ai == 1])
    w_apb = np.array([wi for wi,ai in zip(w,a) if ai == 1])

    # plotting balls for the atomic positions and vectors for the spins with
        transparency
    # and color coding according to the spin orientation (Sx (u) component)

    transp = 0.4 # transparency setting for the balls
    sf = 0.2 # scale factor for spin vectors

    # APB atoms
    if x_apb.size != 0:
        apb_atom = mlab.points3d(x_apb, y_apb, z_apb, u_apb,
                                 scale_mode='none', scale_factor=size, resolution
                                     =30, figure=fig)
        apb_atom.glyph.color_mode = 'color_by_scalar'
        apb_atom.module_manager.scalar_lut_manager.data_range = np.array([-1.,
            1.])
        apb_atom.module_manager.scalar_lut_manager.lut.alpha_range = np.array([
            transp, transp])

        apb_spin = mlab.quiver3d(x_apb,y_apb,z_apb,
                                 u_apb,v_apb,w_apb,scalars=np.array(u_apb),mode='
                                     arrow',
                                 scale_factor=sf, resolution=30, figure=fig)
        apb_spin.glyph.color_mode = 'color_by_scalar'
        apb_spin.module_manager.scalar_lut_manager.data_range = np.array([-1.,
            1.])

    # tetrahedral positions
    tet_atom = mlab.points3d(x_tet,y_tet,z_tet,u_tet,
                             scale_mode='none', scale_factor=size, resolution=30,
                                  figure=fig)
    tet_atom.glyph.color_mode = 'color_by_scalar'
    tet_atom.module_manager.scalar_lut_manager.data_range = np.array([-1., 1.])
    tet_atom.module_manager.scalar_lut_manager.lut.alpha_range = np.array([transp
        , transp])

    tet_spin = mlab.quiver3d(x_tet,y_tet,z_tet,
                             u_tet,v_tet, w_tet,
                             scalars=np.array(u_tet),mode='arrow',
                             scale_factor=sf, resolution=30, figure=fig)
    tet_spin.glyph.color_mode = 'color_by_scalar'
    tet_spin.module_manager.scalar_lut_manager.data_range = np.array([-1., 1.])

    # octahedral positions
    oct_atom = mlab.points3d(x_oct,y_oct,z_oct,u_oct,
                             scale_mode='none', scale_factor=size, resolution=30,
                                  figure=fig)
    oct_atom.glyph.color_mode = 'color_by_scalar'
    oct_atom.module_manager.scalar_lut_manager.data_range = np.array([-1., 1.])
    oct_atom.module_manager.scalar_lut_manager.lut.alpha_range = np.array([transp
        , transp])

    oct_spin = mlab.quiver3d(x_oct,y_oct,z_oct,
                             u_oct,v_oct,w_oct,
                             scalars=np.array(u_oct),mode='arrow',
                             scale_factor=sf, resolution=30, figure=fig)
    oct_spin.glyph.color_mode = 'color_by_scalar'
    oct_spin.module_manager.scalar_lut_manager.data_range = np.array([-1., 1.])

    # coordinate axis
    mlab.quiver3d(0.5,0,1.0,1.5,0,0,color=(0.4,0.4,0.4),mode='arrow', resolution
        =40)
    mlab.quiver3d(0.5,0,1.0,0,1.5,0,color=(0.4,0.4,0.4),mode='arrow', resolution
        =40)
    mlab.quiver3d(0.5,0,1.0,0,0,1.5,color=(0.4,0.4,0.4),mode='arrow', resolution
        =40)

# setting up the figure (size, lighting, camera position etc.)
fig = mlab.figure(size=(1000,1000), bgcolor=(1,1,1))

# in case a colorbar is needed
# get the current lut manager
#lut_manager = mlab.colorbar(orientation='vertical')
# fix the range
#lut_manager.data_range = (-1, 1)

# these lines are needed for the lights to work properly, not sure what they do
    though..
from pyface.api import GUI
_gui = GUI()

while fig.scene.light_manager is None:
    _gui.process_events()

fig.scene.light_manager.lights[0].elevation = 32.37
fig.scene.light_manager.lights[0].azimuth = -5.0
fig.scene.light_manager.lights[0].intensity = 0.9322

fig.scene.light_manager.lights[1].elevation = -34.29
fig.scene.light_manager.lights[1].azimuth = -10.0
fig.scene.light_manager.lights[1].intensity = 0.1582

fig.scene.light_manager.lights[2].elevation = -41.43
fig.scene.light_manager.lights[2].azimuth = 15.48
fig.scene.light_manager.lights[2].intensity = 0.4915

fig.scene.light_manager.lights[3].elevation = 0.0
fig.scene.light_manager.lights[3].azimuth = 0.0
fig.scene.light_manager.lights[3].intensity = 0.0


x,y,z,u,v,w,t,a = np.loadtxt(sfile, usecols=(0,1,2,3,4,5,6,7), delimiter=
    delimiter, unpack=True)
plot_mayavi(True,0.15,x,y,z,u,v,w,t,a)

mlab.gcf().scene.parallel_projection = True

mlab.view(azimuth=90,elevation=180, distance=20)
#mlab.view(azimuth=90,elevation=160, distance=20)
#mlab.view(azimuth=10,elevation=10, distance=20)

fig.scene.anti_aliasing_frames = 20
mlab.draw()
fig.scene.camera.zoom(1.2)

mlab.savefig(file_r+".png", size=(2000,2000))
mlab.show()
```

C.15 average_spin_structure.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import numpy as np
import scipy.stats as stats
import argparse


parser = argparse.ArgumentParser()

parser.add_argument('--input_dir', type=str)
parser.add_argument('--template_file', type=str)
parser.add_argument('--particle_size', type=int)
parser.add_argument('--num_particles', type=int)
parser.add_argument('--dipole', type=str)

args = parser.parse_args()

template_file = args.template_file
particle_size = args.particle_size
num_particles = args.num_particles
dipole = args.dipole
center= particle_size/2.0+0.5
input_dir = args.input_dir

files = []
for i in range(1,num_particles+1):
    fname = template_file.replace("template","%d"%(i))
    fname = fname.replace("None", dipole)
    files.append(input_dir+fname)

outfile = input_dir+template_file.replace("template", "averaged")
outfile = outfile.replace("None", dipole)

def averaging_spins(files, template_file, filename, center):
    xt,yt,zt,tt,at = np.loadtxt(input_dir+template_file,delimiter=',',
                                usecols=(0,1,2,6,7), unpack = True)

    ut = np.zeros_like(xt)
    vt = np.zeros_like(xt)
    wt = np.zeros_like(xt)

    coord_spin_t = list(zip(xt,yt,zt,ut,vt,wt))
    dict1 = {(x,y,z): (u,v,w) for x,y,z,u,v,w in coord_spin_t}

    for file in files:

        x,y,z,u,v,w,t,a = np.loadtxt(file,delimiter=',', unpack=True)
        print(len(x))

        coord_spin = list(zip(x,y,z,u,v,w))
        dict2 = {(x,y,z): (u,v,w) for x,y,z,u,v,w in coord_spin}

        matchxyz = set(dict1) & set(dict2)

        for xyz in matchxyz:
            a = list(dict1[xyz])
            b = list(dict2[xyz])
            for n in range(3):
                a[n] += b[n]
            dict1[xyz] = a

    uf,vf,wf = [],[],[]
    xf,yf,zf = [],[],[]
    for key in dict1:
        x,y,z = key
        u,v,w = dict1[key]
        uf.append(u)
        vf.append(v)
        wf.append(w)
        xf.append(x)
        yf.append(y)
        zf.append(z)

    uf,vf,wf = np.array(uf), np.array(vf), np.array(wf)
    print(len(ut))
    print(len(uf))
    print(len(xf))

    lengths = np.sqrt(uf**2+vf**2+wf**2)
    uf /= lengths
    vf /= lengths
    wf /= lengths
    np.savetxt(filename, np.column_stack((xf,yf,zf,uf,vf,wf,tt,at)),
               fmt='%.5f')

averaging_spins(files, template_file, outfile, center)
```

C.16 measurement_script_MvT.sh

```
#!/bin/bash

# *-----------------------------------*
# |     shell script for M vs. T      |
# |           simulations             |
# |                                   |
# |  please check the paths to the    |
# |       files before running        |
# |                                   |
# *-----------------------------------*


measurement="MvT"
echo $measurement

# directory for outputs
output_dir="/user/specified/output/path/"


#--------------------------
# particle settings
#--------------------------
outer_loop=1
# inner loop determines how many processes are started in parallel
# do not use too many for larger particles --> the system might crash!
inner_loop=1
# total number of particles to be calculated
num_particles=$(($outer_loop*$inner_loop))

# cif-file for crystal structure (only Vesta cif supported)
# if needed import the cif-file in Vesta and save it again under a different
# name. This will generate the cif-file in the right format.
ciffile="/CIF_path/Fd-3m_perfect.cif"
# atom labels from cif-file used for occupancies (need to be comma separated)
atomlabels="Fe(oct),Fe(tet)"
occupancies="0.83,1.0"
latticepars="8.3965,8.3965,8.3965"

# particle shape (either "Sphere" or "Cube")
shape="Sphere"
```

```

# particle size (diameter) in unit cells
particle_size=6

# particle with or without APB
APB=false

# seed for particle generation
seed=20210503

# particle starting orientations
particle_orientations=9999

# magnetocrystalline anisotropy
anisotropy_constant=3.25e-25

# exchange constants: Fe_TT, Fe_OO, Fe_TO
exchange_constants="-21.0,-8.6,-28.1"
# exchange constant across APB in K
APB_constant=-106.28

#--------------------------
# field settings
#--------------------------
measurement_field=0.005
cooling_field=0.0

#--------------------------
# temperature settings
#--------------------------
temperature_upper=30.0
temperature_lower=0.001
temperature_step=0.5

#--------------------------
# Measurement settings
#--------------------------
# number of complete Monte Carlo steps = steps x totalNumAtoms
relaxation_steps=500.0
averaging_steps=0

# sigma parameter for opening of Gaussian cone in trial move
sigma=0.03

# setting for method of dipole interaction calculation
#dipole_interactions="brute_force"
dipole_interactions="None"
#dipole_interactions="macrocell_method"

# macrocell size only used if macrocell method is selected
macrocell_size=0.2

# select the measurements
ZFC=true
FC=true


#----------------------------------
# generate crystal structure files
#----------------------------------
# change path to "generate_nanoparticle.py" if necessary
#
for (( c=1; c<=$num_particles; c++ ))
do
    if [ "$APB" = true ] ; then
        python /program_file_path/generate_nanoparticle.py \
        --output=$output_dir --atomlabels=$atomlabels --occupancies=$occupancies
            \
        --diameter=$particle_size --number=$c --seed=$seed --ciffile=$ciffile \
        --shape=$shape --latticepar=$latticepars --APB
    else
        python /program_file_path/generate_nanoparticle.py \
        --output=$output_dir --atomlabels=$atomlabels --occupancies=$occupancies
            \
        --diameter=$particle_size --number=$c --seed=$seed --ciffile=$ciffile \
        --shape=$shape --latticepar=$latticepars
    fi
done

#--------------------------
# generate JSON files
#--------------------------
# these files are used as input parameter files for the simulation program
# change path to "generate_json_files.py" if necessary
#
if [ "$APB" == true ] ; then
    if [ "$ZFC" == true ] ; then
        if [ "$FC" == true ] ; then
            python /program_file_path/generate_json_files.py \
            --measurement=$measurement \
            --output=$output_dir \
            --structure_path=$output_dir \
            --num_particles=$num_particles \
            --particle_size=$particle_size \
            --meas_field=$measurement_field \
            --dipole=$dipole_interactions \
            --steps=$relaxation_steps \
            --av_steps=$averaging_steps \
            --num_or=$particle_orientations \
            --cool_field=$cooling_field \
            --Tupper=$temperature_upper \
            --Tlower=$temperature_lower \
            --T_step=$temperature_step \
            --anisotropy_constant=$anisotropy_constant \
            --sigma=$sigma --ZFC --FC --APB \
            --latticepar=$latticepars \
            --exchangeconstants=$exchange_constants \
            --APB_constant=$APB_constant
        else
            python /program_file_path/generate_json_files.py \
            --measurement=$measurement \
            --output=$output_dir \
            --structure_path=$output_dir \
            --num_particles=$num_particles \
            --particle_size=$particle_size \
            --meas_field=$measurement_field \
            --dipole=$dipole_interactions \
            --steps=$relaxation_steps \
            --av_steps=$averaging_steps \
            --num_or=$particle_orientations \
            --cool_field=$cooling_field \
            --Tupper=$temperature_upper \
            --Tlower=$temperature_lower \
            --T_step=$temperature_step \
            --anisotropy_constant=$anisotropy_constant \
            --sigma=$sigma --ZFC --APB \
            --latticepar=$latticepars \
            --exchangeconstants=$exchange_constants \
            --APB_constant=$APB_constant
        fi
    fi
else
    if [ "$ZFC" == true ] ; then
        if [ "$FC" == true ] ; then
            python /program_file_path/generate_json_files.py \
            --measurement=$measurement \
            --output=$output_dir \
            --structure_path=$output_dir \
            --num_particles=$num_particles \
            --particle_size=$particle_size \
            --meas_field=$measurement_field \
            --dipole=$dipole_interactions \
            --steps=$relaxation_steps \
            --av_steps=$averaging_steps \
            --num_or=$particle_orientations \
            --cool_field=$cooling_field \
            --Tupper=$temperature_upper \
            --Tlower=$temperature_lower \
            --T_step=$temperature_step \
            --anisotropy_constant=$anisotropy_constant \
            --sigma=$sigma --ZFC --FC \
            --latticepar=$latticepars \
            --exchangeconstants=$exchange_constants \
            --APB_constant=$APB_constant
        else
            python /program_file_path/generate_json_files.py \
            --measurement=$measurement \
            --output=$output_dir \
            --structure_path=$output_dir \
            --num_particles=$num_particles  \
            --particle_size=$particle_size \
            --meas_field=$measurement_field \
            --dipole=$dipole_interactions \
            --steps=$relaxation_steps \
            --av_steps=$averaging_steps \
            --num_or=$particle_orientations \
            --cool_field=$cooling_field \
            --Tupper=$temperature_upper \
            --Tlower=$temperature_lower \
            --T_step=$temperature_step \
            --anisotropy_constant=$anisotropy_constant \
            --sigma=$sigma --ZFC \
            --latticepar=$latticepars \
            --exchangeconstants=$exchange_constants \
            --APB_constant=$APB_constant \
        fi
    fi
fi

#---------------------------
# Run Simulation
#---------------------------
# change path to the compiled simulation program if necessary
#
trap "kill 0" EXIT

count=0
for (( i=1; i<=$outer_loop; i++ )); do
    for ((c=1; c<=$inner_loop; c++ )); do
        ((count=count+1))
        if [ "$APB" == true ] ; then
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_MvsT_simulation_APB_D${particle_size}.json &
        else
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_MvsT_simulation_noAPB_D${particle_size}.json &
        fi
    done
    wait
done
wait
```

C.17 measurement_script_MvB.sh

```
#!/bin/bash

# *-----------------------------------*
# | shell script for hysteresis loop  |
# |         simulations                |
# |                                    |
# |  please check the paths to the     |
# |       files before running         |
# |                                    |
# *-----------------------------------*


measurement="MvB"
echo $measurement

# directory for outputs
output_dir="/user/specified/output/path/"


#--------------------------
# particle settings
#--------------------------
outer_loop=1
# inner loop determines how many processes are started in parallel
# do not use too many for larger particles --> the system might crash!
inner_loop=2
# total number of particles to be calculated
num_particles=$(($outer_loop*$inner_loop))

# cif-file for crystal structure (only Vesta cif supported)
# if needed import the cif-file in Vesta and save it again under a different
# name. This will generate the cif-file in the right format.
ciffile="/CIF_path/P4_32_12_perfect.cif"

# atom labels from cif-file used for occupancies (need to be comma separated)
atomlabels="Fe1,Fe2,Fe3,Fe4"
occupancies="1.0,0.33,1.0,1.0"

# unitcell parameters in Angstrom
latticepars="8.3965,8.3965,8.3965"

# particle shape (either "Sphere" or "Cube")
shape="Sphere"

# particle size in unit cells
particle_size=6

# particle orientations per configuration used for averaging
particle_orientations=20

# particle with or without APB
APB=false

# seed for particle generation
seed=20210503

anisotropy_constant=3.25e-25

# exchange constants: Fe_TT, Fe_TO, Fe_OO
exchange_constants="-21.0,-8.6,-28.1"
# exchange constant across APB in K
APB_constant=-106.28

#--------------------------
# field (in tesla) and temperature (in Kelvin) settings
```

```
#-------------------------
# settings used for the field sweep
lower_field_limit=-1.5
upper_field_limit=1.5
field_step=0.1

# settings for zero field/field cooling
# starting temp. and final temp. have to be different
cooling_field=0.0
starting_temperature=301.0
temperature_step=1.0
final_temperature=5.0

#-------------------------
# Monte-Carlo settings
#-------------------------
# Number of Monte Carlo steps per temperature or field step
steps=8000
cooling_steps=1

sigma=0.03

# setting for method of dipole interaction calculation
#dipole_interactions="brute_force"
dipole_interactions="None"
#dipole_interactions="macrocell_method"

# macrocell size only used if macrocell method is selected
macrocell_size=0.2


#---------------------------------
# generate crystal structure files
#---------------------------------
# change path to "generate_nanoparticle.py" if necessary

for (( c=1; c<=$num_particles; c++ ))
do
    if [ "$APB" = true ] ; then
        python /program_file_path/generate_nanoparticle.py \
        --output=$output_dir --atomlabels=$atomlabels --occupancies=$occupancies
            \
        --diameter=$particle_size --number=$c --seed=$seed --ciffile=$ciffile \
        --shape=$shape --latticepar=$latticepars --APB
    else
        python /program_file_path/generate_nanoparticle.py \
        --output=$output_dir --atomlabels=$atomlabels --occupancies=$occupancies
            \
        --diameter=$particle_size --number=$c --seed=$seed --ciffile=$ciffile \
        --shape=$shape --latticepar=$latticepars
    fi
done

#-------------------------
# generate JSON files
#-------------------------
# these files are used as input parameter files for the simulation program
# change path to "generate_json_files.py" if necessary
#
if [ "$APB" == true ] ; then
    python /program_file_path/generate_json_files.py  \
    --measurement=$measurement --output=$output_dir --structure_path=$output_dir
        \
    --num_particles=$num_particles --particle_size=$particle_size --steps=$steps
        \
    --dipole=$dipole_interactions --macrocell_size=$macrocell_size \
    --num_or=$particle_orientations --Bupper=$upper_field_limit --Blower=
        $lower_field_limit \
    --Bstep=$field_step --start_temperature=$starting_temperature \
    --cooling_steps=$cooling_steps --temperature=$final_temperature \
    --T_step=$temperature_step --cool_field=$cooling_field \
    --anisotropy_constant=$anisotropy_constant --APB --sigma=$sigma \
    --latticepar=$latticepars --exchangeconstants=$exchange_constants \
    --APB_constant=$APB_constant
else
    python /program_file_path/generate_json_files.py \
    --measurement=$measurement --output=$output_dir --structure_path=$output_dir
        \
    --num_particles=$num_particles --particle_size=$particle_size --steps=$steps
        \
    --dipole=$dipole_interactions --macrocell_size=$macrocell_size \
    --num_or=$particle_orientations --Bupper=$upper_field_limit --Blower=
        $lower_field_limit \
    --Bstep=$field_step --start_temperature=$starting_temperature \
    --cooling_steps=$cooling_steps --temperature=$final_temperature \
    --T_step=$temperature_step --cool_field=$cooling_field \
    --anisotropy_constant=$anisotropy_constant --sigma=$sigma \
    --latticepar=$latticepars --exchangeconstants=$exchange_constants
fi

#--------------------------
# Run Simulation
#--------------------------
# change path to the compiled simulation program if necessary
#
trap "kill 0" EXIT

count=0
for (( i=1; i<=$outer_loop; i++ )); do
    for ((c=1; c<=$inner_loop; c++ )); do
        ((count=count+1))
        if [ "$APB" == true ] ; then
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_MvsB_simulation_APB_D${particle_size}.json &
        else
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_MvsB_simulation_noAPB_D${particle_size}.json &
        fi
    done
    wait
done
wait
```

C.18 measurement_script_spin_structure.sh

```
#!/bin/bash

# *------------------------------------*
# | shell script for spin structure    |
# |          simulations               |
# |                                    |
# |  please check the paths to the     |
# |        files before running        |
# |                                    |
# |    installation of Mayavi is       |
# |    required for the 3D plots       |
# |     (see description below)        |
# *------------------------------------*


measurement="spin_structure"
echo $measurement

```

```
# directory for outputs
output_dir="/user/specified/output/path/"


#--------------------------
# particle settings
#--------------------------
outer_loop=20
# inner loop determines how many processes are started in parallel
# do not use too many for larger particles --> the system might crash!
inner_loop=1
# total number of particles to be calculated
num_particles=$(($outer_loop*$inner_loop))

# cif-file for crystal structure (only Vesta cif supported)
# if needed import the cif-file in Vesta and save it again under a different
# name. This will generate the cif-file in the right format.
ciffile="/CIF_path/P4_32_12_perfect.cif"

# atom labels from cif-file used for occupancies (need to be comma separated)
atomlabels="Fe1,Fe2,Fe3,Fe4"
occupancies="1.0,0.83,0.83,0.83"

# unitcell parameters in Angstrom
latticepars="8.3965,8.3965,8.3965"

# particle shape (either "Sphere" or "Cube")
shape="Sphere"

# particle size in unit cells
particle_size=11

# particle orientation angles
alpha=0.0
beta=0.0
gamma=-45.0

# particle with or without APB
APB=true

# effective anisotropy constant: K[kJ/m^3] / N_atoms/particle_volume
anisotropy_constant=3.25e-25

# exchange constants: Fe_TT, Fe_OO, Fe_TO in K
exchange_constants="-21.0,-8.6,-28.1"
# exchange constant across APB in K
APB_constant=-106.28

# seed for particle generation
seed=20210503

#--------------------------
# field settings
#--------------------------
measurement_field=5.0

#--------------------------
# temperature settings
#--------------------------
temperature=0.01


#--------------------------
# Monte-Carlo settings
#--------------------------
# Number of Monte Carlo steps
# (statistical number of trial moves on one spin in the structure)
steps=5000
```

```

# sigma parameter for opening of gaussian cone in trial move
sigma=0.03

# setting for method of dipole interaction calculation
#dipole_interactions="brute_force"
dipole_interactions="None"
#dipole_interactions="macrocell_method"

# macrocell size only used if macrocell method is selected
macrocell_size=0.2

#----------------------------------
# generate crystal structure files
#----------------------------------
# change path to "generate_nanoparticle.py" if necessary
#
if [ "$APB" = true ] ; then
    python /program_file_path/generate_nanoparticle.py --output=$output_dir \
    --atomlabels=$atomlabels --occupancies=$occupancies --diameter=$particle_size \
    --seed=$seed --ciffile=$ciffile --shape=$shape --latticepar=$latticepars \
    --APB --template
else
    python /program_file_path/generate_nanoparticle.py --output=$output_dir \
    --atomlabels=$atomlabels --occupancies=$occupancies --diameter=$particle_size \
    --seed=$seed --ciffile=$ciffile --shape=$shape --latticepar=$latticepars\
    --template
fi


for (( c=1; c<=$num_particles; c++ ))
do
    if [ "$APB" = true ] ; then
        python /program_file_path/generate_nanoparticle.py --output=$output_dir \
        --atomlabels=$atomlabels --occupancies=$occupancies --diameter=$particle_size \
        --number=$c --seed=$seed --ciffile=$ciffile --shape=$shape \
        --latticepar=$latticepars --APB
    else
        python /program_file_path/generate_nanoparticle.py --output=$output_dir \
        --atomlabels=$atomlabels --occupancies=$occupancies --diameter=$particle_size \
        --number=$c --seed=$seed --ciffile=$ciffile --shape=$shape \
        --latticepar=$latticepars
    fi
done

#--------------------------
# generate JSON files
#--------------------------
# these files are used as input parameter files for the simulation program
# change path to "generate_json_files.py" if necessary
#
if [ "$APB" == true ] ; then
    python /program_file_path/generate_json_files.py --measurement=$measurement \
    --output=$output_dir --structure_path=$output_dir --num_particles=$num_particles \
    --particle_size=$particle_size --alpha=$alpha --beta=$beta --gamma=$gamma \
    --meas_field=$measurement_field --temperature=$temperature --steps=$steps \
    --dipole=$dipole_interactions --macrocell_size=$macrocell_size --sigma=$sigma \
    --latticepar=$latticepars --APB --anisotropy_constant=$anisotropy_constant \
    --exchangeconstants=$exchange_constants --APB_constant=$APB_constant
else
    python /program_file_path/generate_json_files.py --measurement=$measurement \
    --output=$output_dir --structure_path=$output_dir --num_particles=
        $num_particles \
    --particle_size=$particle_size --alpha=$alpha --beta=$beta --gamma=$gamma \
    --meas_field=$measurement_field --temperature=$temperature --steps=$steps \
    --dipole=$dipole_interactions --macrocell_size=$macrocell_size --sigma=$sigma
         \
    --latticepar=$latticepars --anisotropy_constant=$anisotropy_constant \
    --exchangeconstants=$exchange_constants
fi

#--------------------------
# Run Simulation
#--------------------------
# change path to the compiled simulation program if necessary
#
trap "kill 0" EXIT

if [ "$APB" == true ] ; then
    /compiled_program_path/MCS3P ${output_dir}/spin_structure_APB_D${
        particle_size}_template.json
else
    /compiled_program_path/MCS3P ${output_dir}/spin_structure_noAPB_D${
        particle_size}_template.json
fi

count=0
for (( i=1; i<=$outer_loop; i++ )); do
    for ((c=1; c<=$inner_loop; c++ )); do
        ((count=count+1))
        if [ "$APB" == true ] ; then
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_spin_structure_APB_D${particle_size}.json &
        else
            /compiled_program_path/MCS3P \
            ${output_dir}/${count}_spin_structure_noAPB_D${particle_size}.json &
        fi
    done
    wait
done
wait


#----------------------------------
# Calculate average spin structure
#----------------------------------

t=${temperature%.*}
mf=${measurement_field%.*}

if [ "$APB" == true ] ; then
    template_filename=D${particle_size}_structure_APB_\
template_spin_structure_${t}K_${steps}MCS_${mf}T_dipNone.txt
else
    template_filename=D${particle_size}_structure_noAPB\
_template_spin_structure_${t}K_${steps}MCS_${mf}T_dipNone.txt
fi

echo $template_filename

python /program_file_path/average_spin_structure.py --input_dir=$output_dir \
--template_file=$template_filename --particle_size=$particle_size --num_particles
    =$num_particles \
--dipole=$dipole_interactions

if [ "$APB" == true ] ; then
    averaged_file=D${particle_size}_structure_APB\
_averaged_spin_structure_${t}K_${steps}MCS_${mf}T_dip${dipole_interactions}.txt
else
```

```
    averaged_file=D${particle_size}_structure_noAPB\
_averaged_spin_structure_${t}K_${steps}MCS_${mf}T_dip${dipole_interactions}.txt
fi


#---------------------------------
# Generate 3D image with Mayavi
#---------------------------------
# Mayavi has to be installed
# use: pip install mayavi
# and: pip install PyQt5
# a conda environment with python version 3.7 is required
# create the environment with: conda create --name py37 python=3.7
source activate py37
python /program_file_path/3dplot_MC_Mayavi.py\
--input_dir=$output_dir --file=$averaged_file
conda deactivate
```

Appendix D

Sticky hard sphere structure factor implementation

In this appendix an implementation of the sticky hard sphere structure factor based on the implementation used in SasView is shown.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Jun 30 20:24:31 2021

Sticky hard sphere structure factor as implemented in SasView 5.0.
"""
import numpy as np

def stickyHardSphere(q, particle_R, hardsphere_R, perturb,stickiness, volfraction
    ):
        if volfraction == 0.0:
            return 1.0

        onemineps = 1.0-perturb
        eta = volfraction/onemineps/onemineps/onemineps
        radius_effective = particle_R+hardsphere_R

        sig = 2.0 * radius_effective
        aa = sig/onemineps
        etam1 = 1.0 - eta
        etam1sq = etam1**2

        qa = eta/6.0
        qb = stickiness + eta/etam1
        qc = (1.0 + eta/2.0)/etam1sq
        radic = qb*qb - 2.0*qa*qc

        if(radic<0):
            return(1.0)

        radic = np.sqrt(radic)
        lam = (qb-radic)/qa
        lam2 = (qb+radic)/qa
        if (lam2<lam):
            lam = lam2

        test = 1.0 + 2.0*eta
        mu = lam*eta*etam1
        if(mu>test):
            return(1.0)

        alpha = (1.0 + 2.0*eta - mu)/etam1sq
        beta = (mu - 3.0*eta)/(2.0*etam1sq)

        kk = q*aa
        k2 = kk*kk
        k3 = kk*k2

        ds = np.sin(kk)
        dc = np.cos(kk)

        aq1 = ((ds - kk*dc)*alpha)/k3
        aq2 = (beta*(1.0-dc))/k2
        aq3 = (lam*ds)/(12.0*kk)
        aq = 1.0 + 12.0*eta*(aq1+aq2-aq3)

        bq1 = alpha*(0.5/kk - ds/k2 + (1.0 - dc)/k3)
        bq2 = beta*(1.0/kk - ds/k2)
        bq3 = (lam/12.0) * ((1.0-dc)/kk)
        bq = 12.0*eta*(bq1+bq2-bq3)

        sq = 1.0/(aq*aq + bq*bq)

        return sq
```

Appendix E

Peak profile implementation

An implementation of the peak profile as described in eq. 4.3 of section 4.2. The implementation uses the *numexpr* package to speed up the calculations.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import numpy as np
import numexpr as ne

# APB component
def A_APB(L, hkl, a0, delta, D):
    h,k,l = hkl
    h,k,l = int(h), int(k), int(l)

    # unaffected peaks
    if (h+k)%4==0 and (h+l)%4==0 and (k+l)%4==0:
        fac = 0

    # a.1 (220), (422)
    elif (h)%2==0 and (k)%2==0 and (l)%2==0:
        fac = 2*(h+k+l)/(a0*np.sqrt(2*(h**2+k**2+l**2)))

    else:
        # a.2.1 (311)
        if (h+k)%4==0 and (h+l)%4==0:
            fac = (h+k+l)/(a0*np.sqrt((h**2+k**2+l**2)))

        # a.2.2 (511), (333)
        else:
            fac = (4*l)/(a0*np.sqrt(3*(h**2+k**2+l**2)))

    return (1-2*delta/D)**(L*fac)

# size component
def A_S(L,D):
    return 1-3/2*(L/D)+1/2*(L/D)**3

# instrumental component
def A_IP(L, nu, sig):
    k = 1/(1+(0.677622)*(1-nu)/nu)
    A = (1-k)*np.exp(-np.pi**2*sig**2*L**2/np.log(2))
    B = k * np.exp(-2*np.pi*sig*L)
    return A+B

def peak_profile(Q, pos, D, hkl, delta, a0, sig):
    N = 1000
    L = np.linspace(0,D,N)
    L = L[:, np.newaxis]

    Fourier_coeff = A_S(L,D) * A_APB(L, hkl, a0, delta, D) * A_IP(L, 0.5, sig)

    exponent = ne.evaluate("exp(1j*L*(Q-pos)).real")
    I = ne.evaluate("Fourier_coeff*exponent")
    I = ne.evaluate("sum((I), axis=0)")
    return I/N*D


if __name__ == "__main__":
    import matplotlib.pyplot as plt

    positions = np.array([2.1160, 2.4813, 2.5917, 2.9927,
                          3.6654, 3.8878, 3.8878, 4.2325])

    hkl = ['220', '311', '222', '004',
           '422', '333', '511', '440']

    intensities = [16.0, 49.6, 1.4, 10.6,
                   7.5, 19.5, 1.0, 28.5]

    Q = np.arange(1.0,5.0,0.01)

    def whole_pattern(Q, positions, intensitites, hkl, delta, D, a0):
        I = np.zeros_like(Q)
        for pos,hkl,i in zip(positions, hkl, intensities):
            I += i*peak_profile(Q, pos, D, hkl, delta, a0)
        return I

    pattern = whole_pattern(Q, positions, intensities, hkl, 0.0, 90, 8.3965)
    plt.plot(Q, pattern)
```