Computational Electron Microscopy Of Small Gold Nanoclusters

Richard Aveyard

Doctor of Philosophy

University of York
Physics
March 2014

Abstract

Scanning transmission electron microscopy is amongst the most valuable techniques for nanoscale structural characterization. It is capable of providing atomic resolution images with lesser sample damage than is typically incurred by other techniques of comparable resolution. Additionally, recent studies have found that the intensity in these images can be used to deduce the three dimensional structures of samples.

The atomic resolution, three-dimensional characterization of gold nanoclusters is particularly desirable, as it is expected to provide significant insights into their surprising catalytic activity. Unfortunately, the image formation process in scanning transmission electron microscopy is not straightforward, with many microscope and sample parameters affecting image intensities. Recently, there has been a concerted effort in the electron microscopy community to achieve more quantitative analyses of images to maximise the information which can be extracted from them. This is typically achieved through the comparison of experimental and simulated images of model structures. To apply such methods to nanoscale structures, the simulations should account for the large inhomogeneities expected in these structures. In particular, both the static structural disordering induced by strain, and the dynamic disordering caused by thermal motion, should be included. These effects are frequently overlooked in reports in the literature, principally because there is currently no means by which they can be accurately measured.

In the work presented here, molecular dynamics simulations are used to predict the structural relaxations and thermal motion in small gold nanoclusters, in order to produce more rigorous electron microscope simulations than any previously reported. This method is equally applicable to any system for which accurate molecular dynamics simulations can be performed. Images produced using this new method are compared with those produced using more conventional techniques and found to be sufficiently different to confirm the value of this approach. The results of the comparisons also prompt a systematic study into the effect of structural disorder on image intensities. It is found that electron channelling effects play a large role in image formation and cause a non-trivial relationship between thermal motion and image intensities. The results of this work show that the interrelated effects of the many factors affecting image formation in scanning transmission electron microscopy preclude parametrizations, so that the physical interpretation of images is expected to continue to rely upon rigorous computational simulations.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisor, Professor Jun Yuan, for his valuable contributions to my work.

I'd also like to thank Professor Riccardo Ferrando for sharing his expertise in molecular dynamics simulations and helping me to arrange a very enjoyable visit to the University of Genoa. Thanks must also go to Professors Ziyou Li and Roy Johnson and Dr. Andrew Logsdail from the University of Birmingham for many interesting discussions. Dr. Dongsheng He is deserving of particular thanks for contributing the experimental image used in chapter 6. I'd also like to thank Dr Richard Evans for his support as I began learning to program GPUs.

I'd also like to thank Laura Clark, for making office work considerably more enjoyable and for her fantastic meringue making skills.

Declaration

The research presented in this doctoral thesis is the work of the author, Richard Aveyard, except where explicitly acknowledged or referenced in the text, in accordance with the examination regulations of the University of York. This work has not previously been presented for any other award at this, or any other, University. A list of publications based on research in this thesis is included.

Papers

Manuscript under review: "Modelling nanoscale inhomogeneities for quantitative HAADF-STEM imaging" submitted May 2014.

Manuscript in preparation: "Channelling and thermal vibrations in HAADF STEM" to be submitted summer 2014.

Presentations

"Evaluation of thermal motion in individual nanoclusters using HAADF-STEM techniques" Oral presentation, Electron Microscopy and Analysis Group Conference, York 2013.

"Observation and correction of local environment effects in atom counting from HAADF-STEM images" Poster presentation Electron Microscopy and Analysis Group Conference, York 2013.

"HAADF-STEM simulations of gold nanoclusters utilizing site-dependent Debye-Waller

factors" Poster presentation, European Microscopy Congress, Manchester 2012.

"Site-dependent atomic vibration calculations for quantitative HAADF STEM analysis." Electron Microscopy and Analysis Group Conference, Birmingham 2011.

Chapter 1

Introduction

Electron microscopy is a staple tool for the characterization of nanoscale samples because it can routinely yield atomic resolution images with significantly less sample interaction than other techniques of comparable resolution. Scanning transmission electron microscopy using high-angle annular dark field detectors offers the highest achievable resolutions and produces directly interpretable two dimensional projected images. Recent studies have demonstrated that the intensity in these images is sensitive to the number of atoms along the optical axis, allowing additional three dimensional data to be extracted from the images via quantitative image analyses. The physical interpretation of this data is, however, non-trivial due to the contributions to image intensities of a number of other factors. Historically, images have been interpreted with the aid of computational simulations of the image formation process. This is difficult in the case of nanoscale structures because parameters such as the magnitude of thermal vibrations and strain-induced crystalline distortions cannot be measured with sufficient accuracy using independent techniques, and so they cannot be accurately accounted for in the simulations.

In this work, a new method is introduced to include inhomogeneous structures and realistic thermal motion into simulations of scanning transmission electron microscope simulations. This method is applied to the simulation of small gold nanoclusters with diameters of the order of a few nanometres.

Gold nanoclusters are of particular interest because they exhibit catalytic activity, in stark contrast to the noble metal behaviour of macroscopic samples. Whilst a number of possible theories for this unusual phenomenon have been suggested, a conclusive explanation has yet to be reached. The importance of the geometrical arrangement of the

constituent atoms is uncontested, prompting a desire to determine the three dimensional structures of nanoclusters with atomic precision.

Here, the well-established techniques of molecular dynamics simulations are used to directly import structural and dynamical properties of nanocluster models into electron microscopy simulations, thereby improving their accuracy. The value of this new method is determined through comparisons of the resulting images with those generated using conventional techniques. These comparisons also prompt a systematic investigation into the manner in which thermal vibrations in the sample effect the image formation process and the resulting image intensities.

In chapter 2, the workings of the scanning transmission electron microscope are introduced. This chapter begins with a description of the fundamental parts of an electron microscope and the image formation process, before detailing the parameters which can affect image quality and the scattering mechanisms involved in the beam-sample interaction. A review of the contemporary procedures for quantitative image analysis is also included.

In chapter 3, the means by which electron microscope image formation is computationally simulated are discussed. The multislice algorithm is explained with an emphasis on how the real-world imperfections described in chapter 2 are accounted for.

In chapter 4, the motivations for studying gold nanoclusters are provided in the form of a literature review of the current understanding of the causes for the unexpected catalytic activity. This chapter also includes discussions of pertinent structural and dynamic properties of gold nanoclusters.

In chapter 5, an overview of molecular dynamics simulations is given, including a more detailed discussion of the Gupta potential used in this work.

Chapter 6 begins with a discussion of the input parameter selection and convergence testing for the molecular dynamics and multislice simulations used throughout this work. A discussion is then given of the combination of the two computational methods to simulate a high-angle annular dark field scanning transmission electron microscope image of a gold nanocluster in a more rigorous manner than any previously reported in the literature. The image produced with this method is analysed and discussed with respect to those produced using more conventional methods. A comparison between a simulation of

a 309-atom gold nanocluster and an experimental image that prompted this work is also discussed. The effects of dechannelling due to tilting the sample away from crystallographic zone axes are then investigated. The chapter is concluded with a discussion of the implications of the results to the current quantitative analysis procedures used for three dimensional structural characterizations.

In chapter 7, the effects of thermally-induced structural disorder on high-angle annular dark field scanning transmission electron microscopy imaging are investigated. Systematic studies are introduced in which non-physical models are used to isolate thermal motion from other disordering. The insights into image formation provided by these simulations are discussed, and apparently contradictory reports in the literature are addressed and reconciled. The new method of combining molecular dynamics and multislice image simulations is then applied to a gold nanocluster over a range of temperatures, both cryogenic and beyond the melting point of the cluster.

Chapter 2

Scanning transmission electron microscopy

2.1 Introduction

The compound optical microscope was invented by Galileo Galilei in 1625. Most modern day optical microscopes share the same basic principles with the earliest microscope: light is transmitted through a sample, magnified by an objective lens and then focused by an ocular lens onto a detector, in Galileo's case, the human eye. Developments in the understanding of light propagation and improved fabrication techniques have provided improvements in the resolution of optical microscopes by reducing aberrations and increasing the magnifying abilities of objective lenses. These conventional optical microscopes are, however, fundamentally limited by the diffraction resulting from the finite size of the objective aperture. This limit was reported by Ernst Abbe in 1873 and is described by equation 2.1.

$$d = \frac{\lambda}{2(n\sin(\theta))} \tag{2.1}$$

Equation 2.1 gives the diameter, d, of the spot produced in the image plane due to an infinitesimal point source in the object plane emitting radiation of wavelength λ. The angular limit of the optical system, determined by the limited extent of the lens transverse to the optical axis, is given by θ, whilst n is the refractive index of the medium surrounding the lens. Two adjacent point sources in the object plane will produce spots in the image

plane which almost entirely overlap but for an infinitesimal portion at the edge. In order to resolve two point sources they must be sufficiently separated in the object plane so that the maxima in their corresponding spots in the image plane can be distinguished. This defines the resolution limit in optical microscopes. The size of the diffraction spots can be reduced, to improve the resolution, by increasing the refractive index of the propagation medium, n, and maximizing the convergence angle, θ. In practice, this gives a maximum product of $n\sin(\theta) \sim 1.4$ for oil immersed lenses. For visible light with wavelengths of around $\lambda = 500$ nm, this yields an optimum resolution of ~ 200 nm for contemporary optical microscopes [1]. To produce any further increases in resolution it is necessary to consider radiation of shorter wavelengths.

The wave-particle duality of matter was posited in 1924 by de Broglie who suggested that particles have a wavelength inversely proportional to their momentum, as given by equation 2.2 [2].

$$\lambda = \frac{h}{p} = \frac{h}{mv}\sqrt{1 - \frac{v^2}{c^2}} \tag{2.2}$$

Where h is Planck's constant and p is the momentum of the particle of mass m travelling with velocity v. The relativistic correction to the momentum introduces the speed of light term, c. Davisson and Germer experimentally confirmed this hypothesis by observing interference patterns in an electron beam reflected from a nickel sample [3]. By considering the observed maxima as Bragg peaks, they calculated the wavelength of electron beams with kinetic energies of 54 eV and 65 eV, and found them to have wavelengths of 1.67 Å and 1.5 Å respectively. These closely match the values of 1.65 Å and 1.52 Å predicted in de Broglie's hypothesis.

The use of magnetic fields to control electron beams was first reported by Plucker in 1858 and in 1926 Busch described how magnets could be constructed to act as lenses for electron beams [4, 5]. The first electron microscope was constructed in the early 1930s by Ruska and Knoll and in 1933 a resolution greater than any contemporary optical microscope was reported [6].

The early electron microscopes were directly analogous to optical microscopes and would now be classed as Conventional Transmission Electron Microscopes (CTEMs or TEMs) as the sample was illuminated by a parallel electron beam which is focussed by

an objective lens after transmission through the sample. The image was then recorded on a photographic plate. A schematic of a CTEM class of microscope is shown in figure 2.1a. More recently, Scanning Transmission Electron Microscopes (STEMs) have been developed, these differ from CTEMs in that the electron beam is focussed to a small point on the sample and raster scanned across the surface, the fraction of the beam which is transmitted can then be recorded by detectors to produce a map of the sample. Figure 2.1b shows a simplified schematic of a STEM microscope.

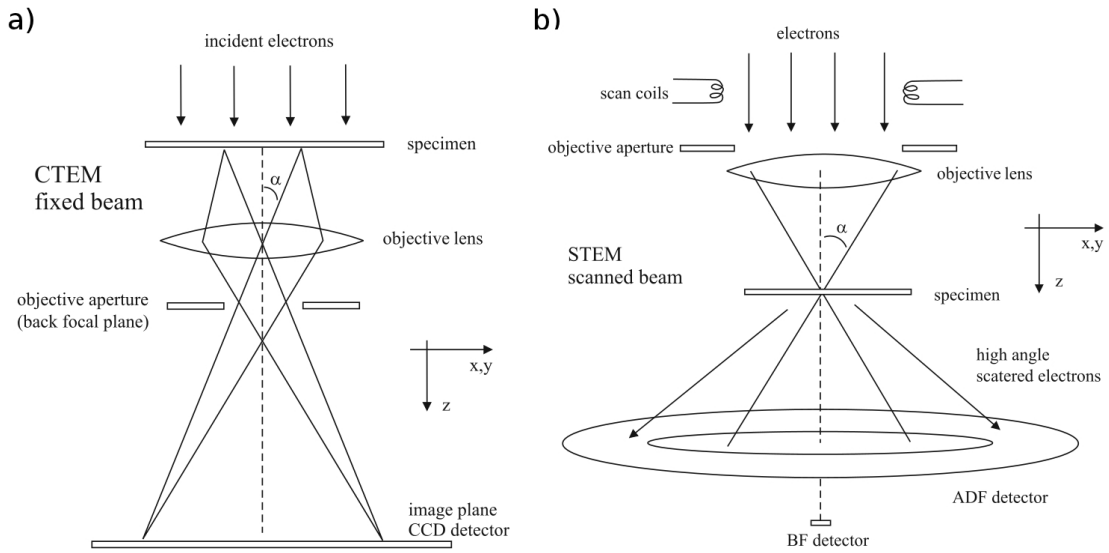

Figure 2.1: Schematic showing the fundamental optics of a) CTEM and b) STEM. Note that angles are exaggerated for visibility. Adapted from [7].

Only scanning transmission electron microscopes are considered in this work, in the following section the process of image formation in these microscopes is described in more detail.

2.2 Scanning transmission electron microscopy

The first scanning transmission electron microscope was produced by Ardenne in 1938 but had a lesser resolution than the CTEMs at the time and saw little further development [8]. The modern day STEM was developed by Crewe in 1969 [9]. A detailed schematic of a typical contemporary electron microscope (JEOL JEM-2100) is shown in figure 2.2.

Modern electron microscopes typically accelerate electrons to energies of 200 keV, resulting in wavelengths of the order of 2.5 pm, 200 thousand times smaller than visible light, though it should be noted that conventional electron microscopes are not currently operating close to the diffraction limit due to a number of technical challenges.

2.2.1 Electron source

Early microscopes developed from cathode ray oscilloscopes produced electron beams through thermionic emission, this approach is still used in most modern microscopes. A filament of tungsten or lanthanum hexaboride (LaB_6) is heated inside a bottomless Wehnelt cylinder to invoke thermionic emission. The Wehnelt cylinder is held at a small negative potential with respect to the filament whilst an anode with an annular aperture is positioned below the Wehnelt cylinder at a high positive potential. The potential difference between the anode and the Wehnelt cylinder drive the electrons down the microscope column towards the magnetic optics. Higher resolution modern microscopes employ field emission guns (FEGS) in which a high potential difference is generated between an anode and a sharp tungsten tip. This induces electron tunnelling from a small area of the tip resulting in a higher brightness and smaller source size. The finite size of the electron source can be a limiting factor in microscope resolution so must be minimized. Schottky FEGs are thermally assisted FEGs, offering more stable beam currents but slightly lesser brightnesses than cold FEGs. The anode used to direct the electrons into the column also accelerates the electrons to the desired energy of between 100 keV and 300 keV depending on their intended use.

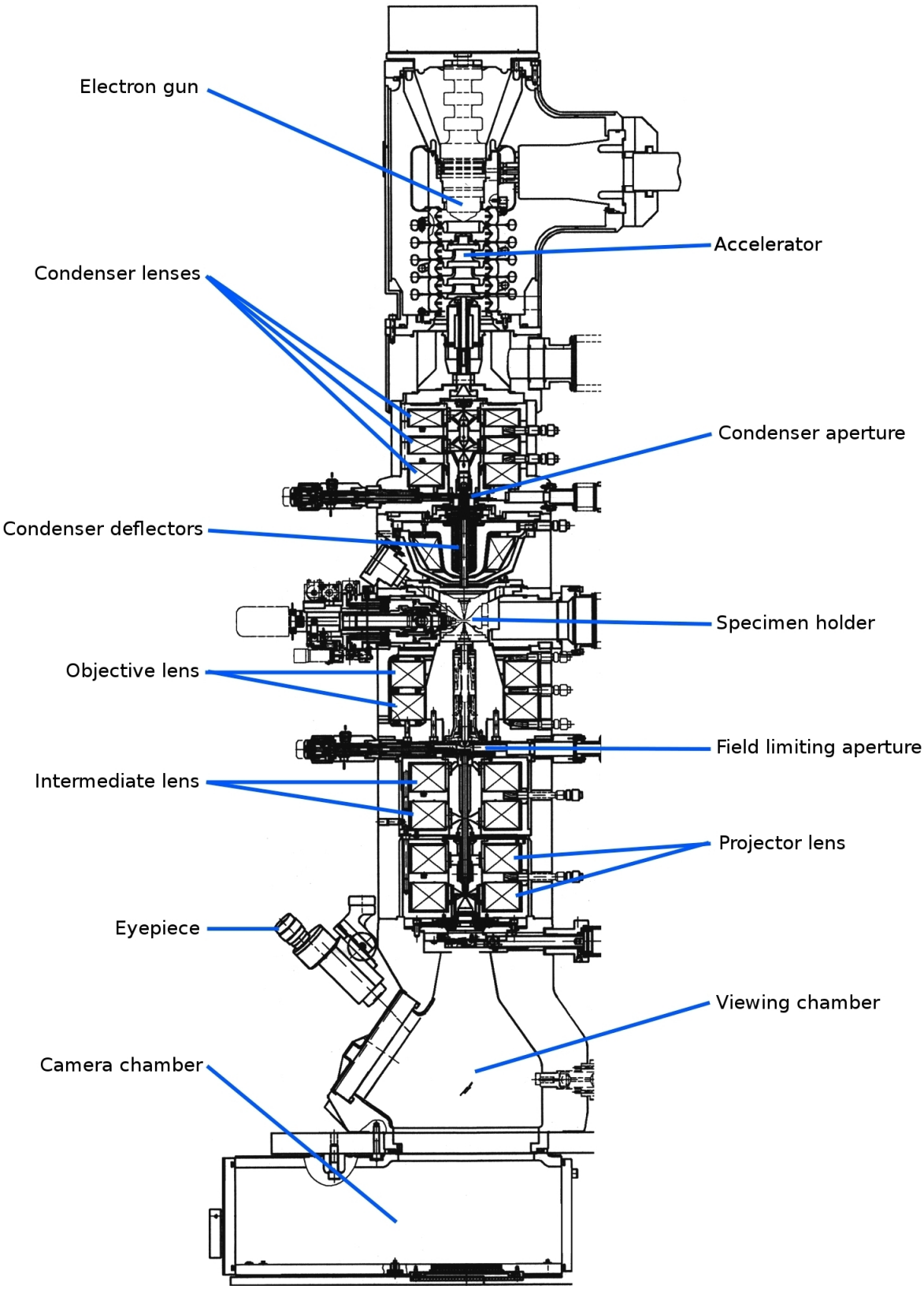

Figure 2.2: Schematic of JEM-2100 electron microscope. Image modified from [10]

2.2.2 Magnetic lenses

After production by the gun, the electron beam is directed down the centre of the microscope column by a series of magnetic lenses. Each magnetic lens consists of a cylindrically symmetric soft magnetic polepiece surrounded by a copper coil. The polepiece has an axial aperture through which the electron beam passes. A current is induced in the coils in order to produce a magnetic field which magnetizes the polepiece. This produces an inhomogeneous magnetic field within the aperture which can be adjusted by altering the current in the coils. As the electrons enter the magnetic field they are subjected to a Lorentz force, given by equation 2.3.

$$\mathbf{F} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right) \tag{2.3}$$

In which $\mathbf{F}$ is the force acting on the particle of charge ,q, moving with velocity ,$\mathbf{v}$, in electric and magnetic fields given by $\mathbf{E}$ and $\mathbf{B}$, respectively. An electron travelling exactly down the axis of the pole piece, antiparallel to the magnetic field, will not be subjected to any force due to the magnet. Electrons travelling at an angle to the axis will undergo a force with components along the optical axis and perpendicular to it. These electrons will thus follow a helical trajectory through the magnetic field. If only electrons close to the optical axis are considered (this is the paraxial approximation), the motion of electrons in the magnetic field can be described by equations 2.4 and 2.5 [11].

$$\frac{d^2r}{dz^2} + \frac{e}{2m_0c^2}\frac{\mathbf{B}^2r}{2\mathbf{v}^{1/2}} = 0 \tag{2.4}$$

$$\frac{d\theta}{dz} = \left(\frac{e}{2m_0c^2}\right)^{1/2}\frac{\mathbf{B}}{2\mathbf{v}^{1/2}} \tag{2.5}$$

Where r is the radial distance from the optical axis, z is the displacement along the optical axis, e is the charge of an electron, m_0 is the electron rest mass, $\mathbf{v}$ is the accelerating voltage and θ is the azimuthal angle perpendicular to the optical axis. From equations 2.4

it can be seen that the radius of the helical trajectory can be reduced by increasing the field strength to focus the electrons towards the optical axis.

The exact configuration of magnetic lenses used for STEM varies between microscopes but usually consists of a series of condenser lenses followed by an objective lens. The purpose of the condenser lenses is to collimate the electrons into a narrow parallel beam at the objective lens. Conventionally TEMs use two condenser lenses, labelled C_1 and C_2. The C_1 lens produces a demagnified image of the gun source to narrow the beam allowing for the production of probe sizes which are smaller than the gun source size. The C_2 lens is usually not engaged in STEM mode, however it has an adjustable aperture which is used to control the convergence angle of the beam at the objective lens.

The objective lens is frequently an immersion lens wherein the sample holder sits inside the objective polepiece. In STEM the objective lens is used to focus the parallel beam produced by the condensers into a small spot on the sample surface. The size of the spot in the sample plane defines the resolution of the STEM. Before the objective lens there are additional scan coils which are used to deflect the focused probe away from the optical axis so that it can be scanned across the sample surface, ideally these should not affect the beam profile, just displace it transversely to the optical axis.

2.2.3 Scattering mechanisms

When the electron beam enters the sample, its constituent electrons can be scattered by coulomb interactions with the positive charge due to protons in the atomic nuclei and the negative charge of bound electrons in the sample. These interactions can be elastic, in which case the beam electrons undergo a change in direction but no change in kinetic energy, or they can be inelastic, wherein the beam electrons lose or gain energy. The majority of the electron beam usually passes through the sample with little or no scattering, although this may not be the case if the sample is particularly thick. This beam is of relatively little interest as it has not interacted with the sample and so does not contain any information pertaining to it, though it is possible to infer some information from the fraction of the incident beam which is transmitted this way.

Elastic scattering is dominated by electron-nucleus interactions caused by coulomb attraction and is similar in nature to the backscattering of alpha particles observed by Geiger and Marsden and explained by Rutherford [12]. In his study of alpha particle scattering

Rutherford developed the single-atom differential cross-section given by equation 2.6.

$$d\sigma_R(\theta) = \frac{e^4 Z^2}{16(4\pi\epsilon_0 E_0)^2} \frac{d\Omega}{\sin^4 \frac{\theta}{2}} \tag{2.6}$$

The cross-section, σ_R gives an effective area within which an electron beam accelerated to an energy of E_0 will be scattered through an angle, θ into a solid angle Ω by a nucleus of atomic number, Z. Here, ϵ_0 is the permittivity of free space.

The bound electrons in the sample counteract some of the positive charge of protons in the nucleus which effectively reduces the strength and range of the positive electric field, this is known as the screening effect. Rutherfords equation does not account for this but it can be corrected using a screening parameter, θ_0, given by equation 2.7. Relativistic effects can also be corrected, by adjusting the electron beam wavelength using equation 2.8. This leads to equation 2.9 which is a good approximation for elastic electron scattering by atoms of low atomic mass.

$$\theta_0 = \frac{0.117 Z^{1/3}}{E_0^{1/2}} \tag{2.7}$$

$$\lambda_R = \frac{h}{\left[2m_0 eV\left(1 + \frac{eV}{2m_0 c^2}\right)\right]^{1/2}} \tag{2.8}$$

$$d\sigma_R(\theta) = \frac{\lambda_R^4 Z^2}{64\pi^4 a_0^2} \frac{d\Omega}{\left[sin^2(\frac{\theta}{2}) + \frac{\theta_0^2}{4}\right]^2} \tag{2.9}$$

Where a_0 is the Bohr radius given by equation 2.10.

$$a_0 = \frac{h^2 \epsilon_0}{\pi m_0 e^2} \tag{2.10}$$

The more involved differential cross-section of Mott is required to accurately describe scattering by heavier species [11], but is not discussed in more detail here as the closely

related scattering factor is more relevent. As mentioned in section 2.1, the beam in electron microscopes has been observed to behave like a wave, exhibiting superposition effects [3]. It is thus necessary to consider coherence effects by using the wave analogue to the differential cross-section, the atomic scattering factor. The atomic scattering factor gives the amplitude of an electron wave scattered by a single atom, the Mott-Bethe formulation of the atomic scattering factor is given in equation 2.11 [13].

$$f_e(q) = \frac{8\pi^2 m_0 e^2}{h^2}\left(\frac{Z - f_x(q)}{q^2}\right) \tag{2.11}$$

The term $f_x(q)$ is the x-ray scattering factor which accounts for elastic electron-electron scattering and is tabulated for most elements [11]. The Mott-Bethe formula (equation 2.11) contains a singularity at $q = 0$ where $f_x(q) = Z$. Ibers identified this problem and proposed equation 2.12 as a solution at $q = 0$, in which $< r^2 >$ is the mean square radius of the atom [14].

$$f_e(0) = \frac{4\pi^2 m_0 e^2}{3h^2} Z < r^2 > \tag{2.12}$$

The Mott-Bethe forumula with Iber's correction give a good approximation for elastic sacttering by a single atom.

Inelastic scattering of the beam electrons can arise from several interactions:

1. The beam electrons can excite plasmons in the bound electrons of the sample.
2. The beam electrons can induce shell transitions or ionisations of the bound electrons.
3. The beam electrons can excite phonons in the sample crystal lattice.
4. The beam electrons can generate photons whilst decelerating in the sample

All of these inelastic scattering events impart information to the beam which can be used to measure properties of the sample with spectroscopic techniques but they do not contribute significantly to HAADF images so will not be discussed further here.

2.2.4 Electron detection

A number of different detectors have been developed to extract information from the STEM, these can be split into those used for imaging and those for chemical analysis. In the imaging modes, a detector measures the intensity of some portion of the transmitted beam as the focussed electron beam is scanned over the sample surface. By measuring the intensity from each point on the sample, an image can be constructed. In bright field (BF) imaging techniques, a detector is positioned below the sample such that it collects those electrons which undergo little or no diffraction. This is known as bright field imaging because the majority of the sample will usually appear in bright contrast due to the high direct transmission count. At probe positions in the sample containing large numbers of atoms there will be less transmission, resulting in dark regions in the image. Figure 2.3 shows a schematic with a BF detector and an example image. Dark field (DF) imaging makes use of an annular detector which detects only scattered electrons, whilst the directly transmitted beam passes through a hole in the centre of the detector. The DF detector produces a dark image with bright spots in regions where greater scattering occurs. The example images in Figure 2.3 show that the BF and DF modes produce complimentary images. The scattering angles which are covered by these detectors can be adjusted with the use of post-sample lenses which effectively alter the path length between the sample and detector.

The BF and DF detectors rely largely on elastic scattering to produce contrast as it is elastic interactions with nuclei which produce higher scattering angles. The probability of elastic scattering through a certain angle is thus dependent on the atomic mass of the nuclei causing the scattering and the number of atoms in the path of the beam. As discussed in section 2.2.3, electron beams exhibit wave behaviour, with Bragg diffraction observed in crystalline structures. This is useful in that it allows the production of diffraction patterns, however, the interference of coherently scattered electrons can produce misleading contrast in the predominantly mass-thickness images of BF and DF STEM. In 1979, Howie proposed that High-angle annular dark field (HAADF) detectors similar to the annular dark field detectors, but with collection angles even further from the beam axis, should minimise the detection of coherently scattered electrons and so avoid misleading contrast [15]. This mode of microscopy has a cross-section close to Rutherfords predictions (equation 2.6) and gives contrast which is highly dependent on atomic mass and so it is also known as Z-contrast microscopy. This mode of microscopy offers the highest resolutions

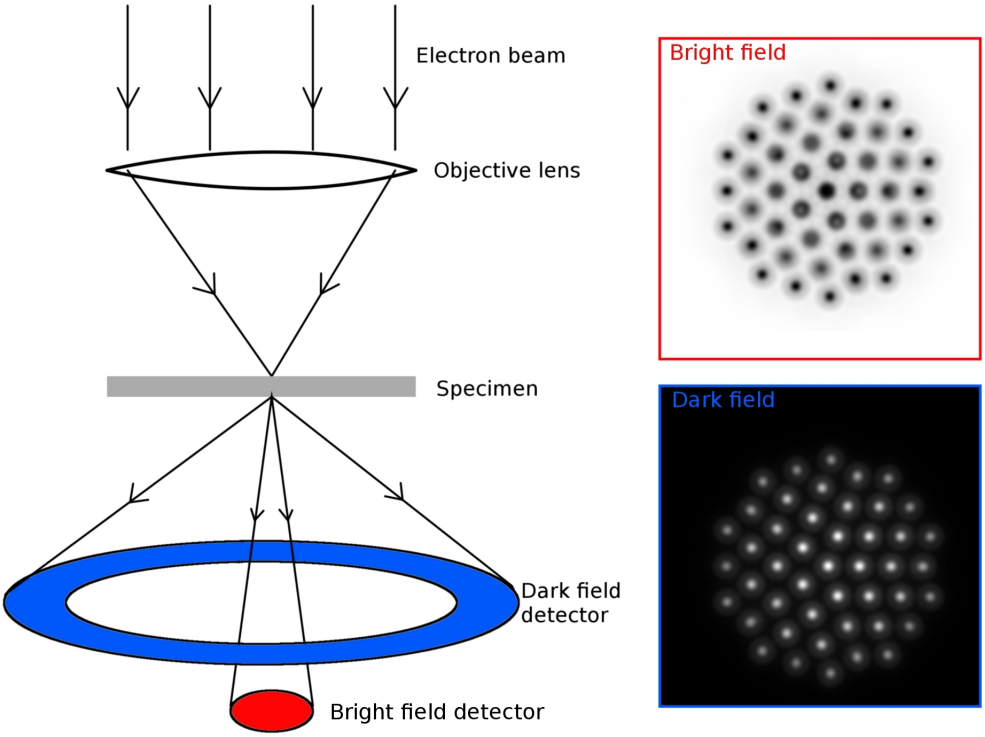

Figure 2.3: Schematic of bright and dark field detector geometries with simulated example images inset. Atomic columns in the beam path scatter electrons out of the bright field detector range into the dark field range so that the intensity at column positions is reduced in the bright field causing dark spots and increased in the dark field causing bright spots.

and is the focus of this work. A more detailed discussion of the contrast in HAADF-STEM is given is section 2.3.

The most common type of detector used in HAADF-STEM is a coupled scintillator and photomultiplier, a schematic of which can be seen in figure 2.4. In a scintillator-

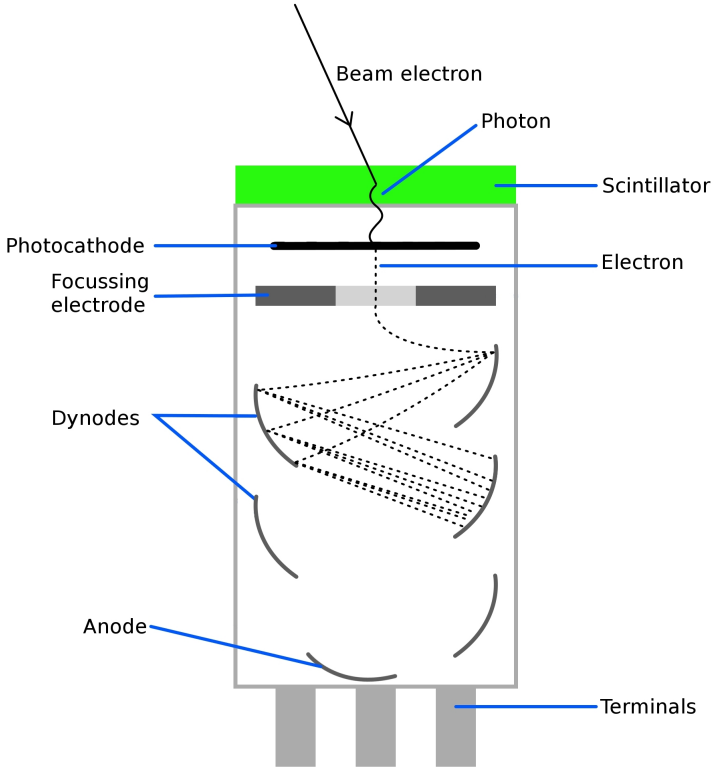

Figure 2.4: Scintillator and photomultiplier tube. The scintillator converts incoming high-energy electrons into photons which, in turn, produce a number of lower energy electrons at the photocathode. The low energy electrons are directed onto a dynode in the photomultiplier tube. As each electron impinges on a dynode it releases multiple electrons so that the electron flux grows exponentially down the tube. The amplified signal is then collected at the anode.

photomultiplier detector system the high-energy beam electrons transmitted through the microscope are directed onto a scintillator which is cathodoluminescent, that is, it emits photons when bombarded with electrons. The decay time of these scintillators is ~ 30 ns, which allows for fast image acquisition without any overlap in signals between adjacent scanning positions. The photons are, in turn, directed by a light pipe onto a photocathode, which emits lower-energy electrons via the photoelectric effect. The electrons produced at the photocathode are then accelerated towards a dynode by a potential difference. The electrons colliding with the dynode induce secondary electrons with a net gain in free electrons. A series of subsequent dynodes, each at a positive potential to the previous

one, repeat this process resulting in a gain of approximately 10^n, where n is the number of dynodes [11]. After the final dynode the electrons are collected by an anode mesh so that the current can be measured to quantify the beam count incident on the scintillator. Semiconductor detectors and charge-coupled devices (CCD) can also be used but the high gain, low noise level, and high acquisition rate make the scintillator-photomultiplier the preferred option for most applications.

2.3 Quantitative High-angle annular dark field scanning transmission electron microscopy

High-angle annular dark field scanning transmission electron microscopy offers amongst the highest resolutions of any conventional imaging technique and so it has become a staple in nanoscale analysis. When Howie first proposed the use of high-angle annular dark field (HAADF) detectors, one of its major benefits was that images could be directly interpreted in a qualitative manner, as the contrast should be caused by incoherent thermal diffuse scattering with negligible misleading interference effects. More recently, however, it has been found that detailed quantitative analyses allow more information to be extracted from the images. In order to correctly interpret the images it is necessary to have a detailed understanding of the factors affecting contrast in experimental HAADF-STEM images. These can be split into two classifications: contrast due to the properties of the sample and contrast caused by artefacts of the imaging system. The microscope parameters affecting the contrast are:

1. The accelerating voltage of the incident beam,
2. The effective size of the electron beam source,
3. The convergence angle of the incident beam,
4. The defocus of the incident beam,
5. The optical quality of the magnetic lens system,
6. The angular range of the annular detector,
7. The response of the detector to the incident beam,

The specimen characteristics which affect the contrast in HAADF-STEM images are:

1. The number of atoms in the path of the beam,
2. The proton numbers of the atoms,
3. The geometrical arrangements of the atoms,
4. The thermal motion of the atoms in the specimen,

These numerous microscope and specimen parameters combine in a non-trivial fashion so that extracting quantitative data from HAADF-STEM images is difficult in many cases. In this section each of the above parameters will be discussed in terms of their origins, their effects on image contrast and how they can be isolated and measured.

2.3.1 Accelerating voltage of the incident beam

The energy of the beam electrons is defined by the accelerating voltage applied to the acceleration coils in the microscope. Instabilities in the current supply to these coils results in a small spread of intensities, resulting in a spread of wavelengths in the beam electrons, this leads to a transverse spreading of the beam due to chromatic aberrations of the lens system as described in section 2.3.5. The extent of the beam spreading due to chromatic aberrations, d_c is given by equation 2.13 [16].

$$d_c = C_c \frac{dE}{E_0} \alpha \tag{2.13}$$

In which C_c is the coefficient of chromatic aberration, α is the convergence semi-angle, and dE is the energy variation from the nominal value, E_0. In addition, the energy of the beam defines its wavelength in accordance with equation 2.2 so a higher energy beam has a shorter wavelength and is more readily focused (see equation 2.1). The beam voltage thus contributes to determining the size of the scanning probe spot on the sample surface, the larger the voltage, the smaller the spot and the better the resolution. Increasing the accelerating voltage consequently reduces the cross-section for inelastic scattering and so reduces specimen heating [11]. The downside to a high energy beam is that the electrons have greater momentum and so are more likely to damage the sample. This is particularly important for unstable samples such as nanoclusters, and has prompted research into low voltage experiments [17]. The effects of beam irradiation on samples are not well characterised and have not been included in computational simulations as of yet.

2.3.2 The effective size of the electron beam source

As discussed in section 2.2.1, electron sources usually consist of a filament which is heated and subjected to a potential, in order to extract electrons. The size of the filament which

contributes electrons to the beam is usually of the order of a few Ångstroms. This introduces transverse incoherence into the electron beam. The effective size of the source also includes contributions from Coulomb interactions within the beam, demagnification by the column optics, and instabilities in the column optics. In modern aberration corrected microscopes, the effective source size is one of the key limiting factors in determining the size and shape of the electron beam impinging on the sample [18]. The effective source size manifests itself in a broadening and reduction in intensities in HAADF-STEM images. In the past the contribution of the effective source has been estimated by considering it an adjustable parameter when comparing experimental and simulated images [19]. This is a hazardous approach as the difference between experimental and simulated images could be caused by a number of other factors such as strain and thermal motion in the specimen, which can be overlooked if this method is employed. Several methods have since been developed to measure the effective source, by measuring the intensities of interference fringes in convergent beam electron diffraction (CBED) patterns. A discussion of the methods and their relative merits can be found in reference [18]. Since the effective size is sensitive to the electron optics before the sample, it is necessary to measure it under similar conditions to those in which experiments will be performed. Furthermore, the ageing of the electron source alters the source size so that it should be measured as near as possible to when images are recorded [19].

2.3.3 The convergence angle of the incident beam

The beam convergence angle is the angle at which the beam focuses onto the sample from the objective aperture. Its size is determined by the radius of the aperture and the convergence of the beam prior to passing through the aperture. The aperture size is selected from a number of different sized holes on a slide in the objective plane (in the case of combined CTEM/STEM microscopes the condenser aperture is the effective objective aperture for STEM) so is limited to discrete values. The convergence angle of the beam incident on the aperture is defined by the electron optics and can be adjusted continuously. The convergence angle affects the resolution of the microscope in STEM by altering the footprint of the scanning beam on the specimen. A larger convergence angle reduces diffraction and so reduces the size of the Airy disk footprint of the beam. The relationship was determined by Rayleigh and is given by equation, which determines the radius of the first maxima, r_{df} [20].

$$r_{df} = 0.61\frac{\lambda}{\alpha} \tag{2.14}$$

As the convergence angle increases, the effective aperture allows transmission further from the optical axis. As discussed in section 2.3.5, electron optics are susceptible to spherical aberrations which cause over-focusing of beams further from the optical axis, causing a spread in focus. Thus, increasing the convergence angle increases the broadening due to spherical aberrations given by equation 2.16. The optimum convergence angle is thus a compromise given by equation 2.15.

$$\alpha = 0.77\frac{\lambda^{\frac{1}{4}}}{C_s^{\frac{1}{4}}} \tag{2.15}$$

The effects of altering the convergence angle on the beam profile are illustrated in figure 2.5, which shows 6 simulated beam profiles at the sample plane with different convergence angles for an accelerating voltage of 200 keV and a spherical aberration of 0.3 mm. The 5 mrad convergence angle has the lowest spreading due to spherical aberrations but is broadened by diffraction from the small effective aperture. The 10 mrad convergence is a good compromise with a sharp central peak. The 15, 20, 25 and 30 mrad convergence angles are progressively less sensitive to diffraction spreading but exhibit increasing spreading due to spherical aberrations.

For modern probe aberration corrected microscopes the spherical aberration is reduced so that large convergence angles are preferred to reduce diffractive spreading and convergence angles of 20-30 mrad are typical for 200 keV beams.

2.3.4 Defocus

When performing experiments the electron beam is focused onto the sample by adjusting the strength of the magnetic lenses. The focal length chosen is thus subjective and does not necessarily match the theoretically ideal focal length. The difference between the position of the sample and the theoretical focal plane is the defocus. The sample will rarely be flat at electron wavelength scales so the defocus will vary over the surface of the

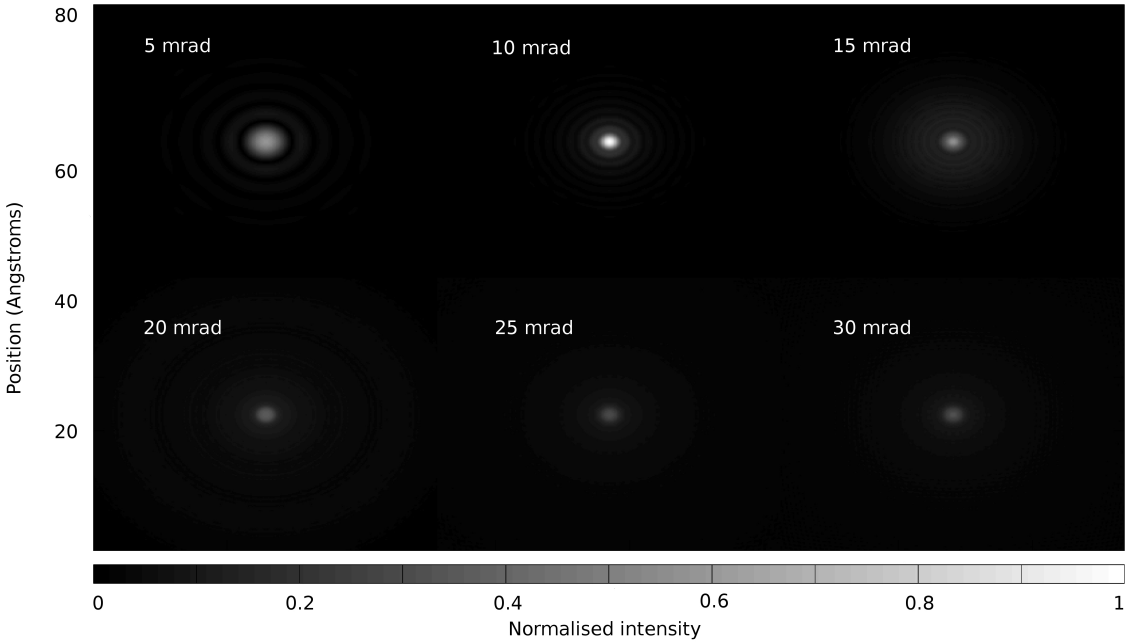

Figure 2.5: Simulated beam profiles at the sample plane for an accelerating voltage of 200 keV, a spherical aberration of 0.3 mm and varying convergence angles.

sample and the user must judge the best focal length based on their interests. Figure 2.6 shows the electron beam at the sample plane with differing defocus values from -10 to 10 nm. Negative defocii occur where the focal plane is beyond the sample, this can also be described as underfocused. The effect of defocus is to reduce the microscope resolution, smearing points in the image into disks. It should be noted that for microscopes with significant spherical aberrations, the optimal sample plane has some underfocus so that the sample sits close to the plane of least confusion, as shown in figure 2.7. It should also be noted that instabilities in the current supplies to the magnetic lenses lead to fluctuations in the defocus so that experimental images consist of a time-averaged set of images at slightly different defocii.

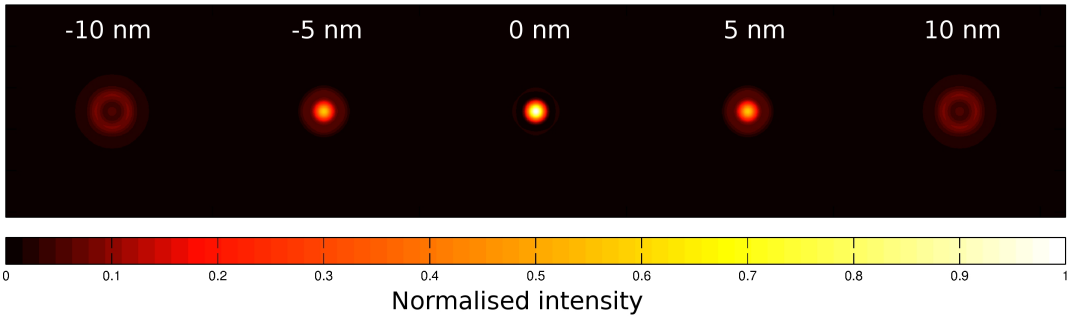

Figure 2.6: Simulated beam profiles at the sample plane for a range of defocus values.

2.3.5 Lens defects

The magnets used in electron microscopes do not constitute ideal lenses, they have defects that lead to systematic imperfections in the beam. The major contributions to imperfect behaviour are spherical aberration, chromatic aberration, astigmatism and coma.

Spherical aberration in magnetic lenses occurs due to the greater deflection of electrons at greater distances from the column axis, as described by equation 2.4. The effect of this is shown in the ray diagram in figure 2.7. Spherical aberration results in a shorter focal

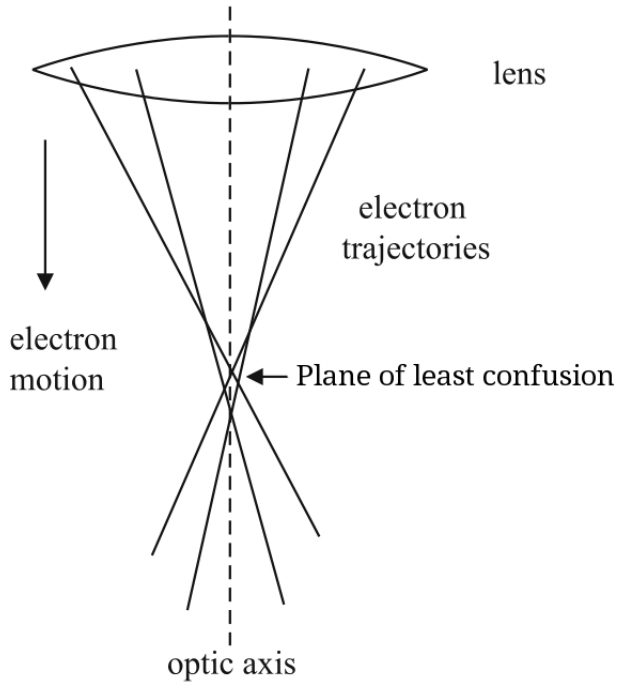

Figure 2.7: Ray diagram showing axial spread of focal points due to spherical aberration. Adapted from [7].

lengths for rays farther from the optical axis so that they cross the optical axis before rays which enter the lens closer to the optical axis. This means that the narrowest point in the beam is a disk rather than a point. The plane in which the smallest possible disk is produced is known as the plane of least confusion. The extent to which spherical aberration affects a lens is quantified by the spherical aberration constant, C_s defined by equation 2.16.

$$r_{sph} = C_s \alpha^3 \tag{2.16}$$

Where: r_{sph} is the radius of the beam in the Gaussian image plane, C_s is the spherical aberration coefficient and α is the convergence semi-angle.

In 1936 Scherzer published a paper in which he derived a mathematical description of the spherical aberration of cylindrically symmetric magnetic lenses [21]. This paper suggested that the problem could not be overcome. However, in a later paper, he suggested a solution making use of non-cylindrically symmetric corrective lenses [22]. Proof of principle experiments were conducted by Seeliger [23] and Möllenstedt [24] in which a combination of cylindrically symmetrical and octopole lenses produced a negative coefficient of spherical aberration, which could potentially counter the positive aberration inherent to cylindrically symmetrical lenses. In 1964 Deltrap successfully used a combination of quadrupole and octopole lenses to correct the spherical aberration of an electron beam on an optical bench, but not in a useful microscope device [25]. The first corrected microscope was a scanning electron microscope (SEM) with quadrupole and octopole correctors, developed by Zach and Haider [26]. Subsequently, with the assistance of Rose and Urban, they produced a sextupole corrector for use in a conventional transmission electron microscope [27]. As Haider *et al.* developed their sextupole correctors, Krivanek and Dellby continued to work on combined quadrupole-octopole systems, capitalizing on newly available computational power to control the currents to a number of corrector coils and thereby automate the correction system in a scanning transmission electron microscope. They demonstrated correction in 1997 [28] and achieved a resolution of 1.4 Å in 2000 [29] the first sub-Ångstrom resolution images were produced using the same microscope in 2002 [30].

Chromatic aberration is the second-most significant aberration effect in magnetic lenses and has become more important with the correction of the previously dominant spherical aberration. Chromatic aberration occurs because the magnetic lenses diffract lower energy electrons more strongly, as seen in equation 2.4. This energy dependence causes a further spreading of the focal point in the microscope resulting in a point source in the object appearing as a disk at optimum focus. This is illustrated in the ray diagram in figure 2.8. The chromatic aberration can be considered to exist as a result of the variation in the energies of electrons in the beam. This spread of energies can be caused by the inherent spread of energies produced by the source gun and by inelastic scattering in the sample.

Chromatic aberration can be physically corrected in two ways [11]. The beam can be energy filtered to narrow its energy distribution, but this reduces the beam intensity.

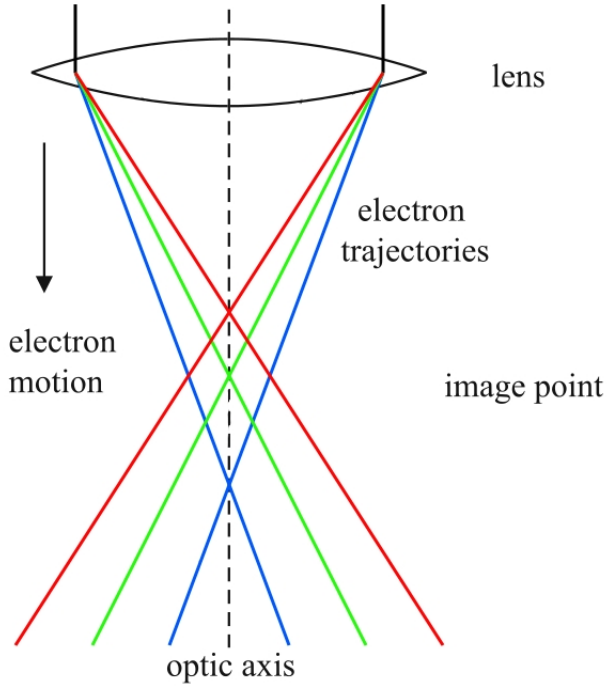

Figure 2.8: Chromatic aberrations result in the overfocusing of lower energy electrons in the beam.

Alternatively, a monochromator can be used which separates out low energy electrons, accelerates them to the desired energy and then returns them to the beam. Monochromators are expensive and so are used mainly in electron energy loss spectroscopy (EELS) whilst filtering is employed for other modes. In comparing the CTEM and STEM schematics in figure 2.1 it can be seen that, unlike CTEM, in STEM there are no lenses after the sample. The result of this is that in STEM modes there is no post-sample chromatic aberration. This is important because scattering in the sample introduces a far greater range of electron energies than there are in the source. Post-sample chromatic aberrations are therefore more significant than pre-sample. The avoidance of post-sample chromatic aberrations allows for greater resolutions in STEM than is possible in CTEM.

Astigmatism occurs in magnetic lenses due to the limitations of manufacturing processes: polepieces which aren't perfectly homogeneous and cylindrically symmetric produce fields with defects. In addition, slightly misaligned polepieces and charging of contaminants can introduce asymmetries. Astigmatism can be corrected with octopole stigmators, which are additional low-symmetry magnetic elements that the user controls to compensate for defects in the main lenses.

2.3.6 High-angle annular detector range

The detector range is the angular range, measured with respect to the beam axis, over which the electron intensity is measured. Howie first proposed the use of a high-angle annular dark field detector (HAADF) in 1979 [15]. The annular detector allows the unscattered beam to pass through a central hole without detection. The majority of the electron beam passes through a TEM sample with little angular deviation so using an annular detector severely limits signal strength. However, it does give a signal which is almost entirely incoherent, by minimising contributions from Bragg scattering which dominate at low angles. This is important because the superposition of coherently scattered electrons can produce interference patterns at the detector so that the signal strength is not proportional to the probability of scattering. By minimising the coherent contribution, the signal strength at the detector becomes proportional to the scattering probability and images are directly interpretable. When using annular detectors a compromise must be found to optimise the detector geometry so that a sufficiently large signal is detected whilst the signal remains predominantly incoherent.

There are two sources of coherent scattering in STEM, longitudinal coherence and lateral coherence, these are caused by interference between the electrons scattered by atoms transverse and parallel to the optical axis, respectively. This is illustrated in figure 2.9a). Jesson and Pennycook have conducted a thorough survey of coherence in HAADF-STEM, leading them to suggest the lower limit for the inner angle of the HAADF detector, θ_i, given in equation 2.17, to minimise lateral coherence between atoms seperated by ΔR [31].

$$\theta_i = \frac{1.22\lambda}{\Delta R} \tag{2.17}$$

For 100 keV electrons and $\Delta R = 1.5$ Å this limit is $\theta i = 30$ mrad and results in a deviation from perfect incoherence of 5% due to laterally coherent contributions from stationary neighbouring atoms [31]. Thermal vibrations of the atoms spatially offset the scattering sources and further reduce this value to 3%. The thermal vibrations result in inelastic, incoherent scattering, rather than the coherent scattering that would occur if the atoms were static. This is known as thermal diffuse scattering (TDS). Whilst lateral coherence is highly sensitive to detector angle and can be significantly attenuated by selection

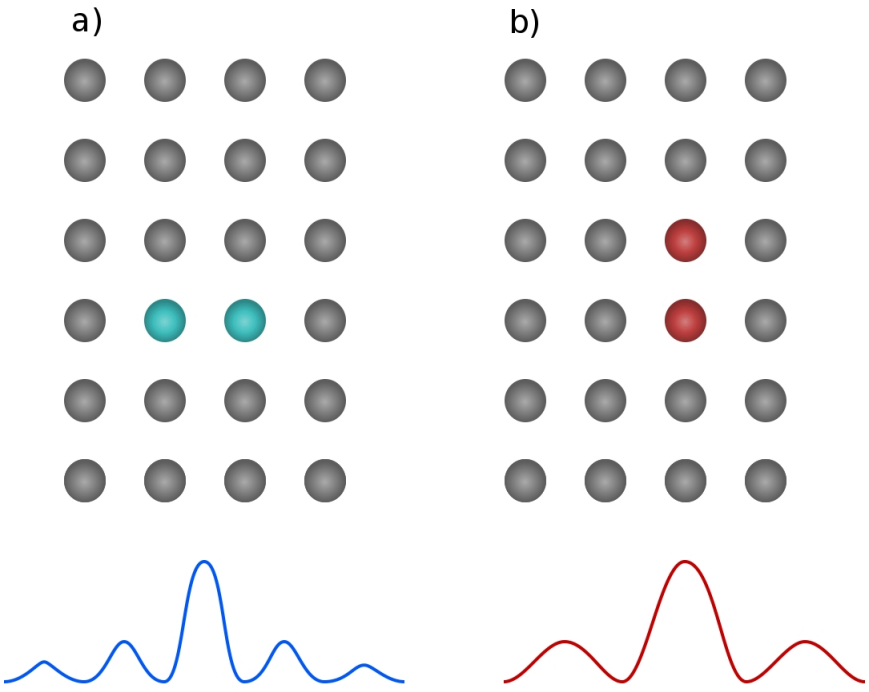

Figure 2.9: Image illustrating the sources of lateral and longitudinal coherence in HAADF-STEM.

of large inner angles, longitudinal coherence remains a problem. Figure 2.9b) shows a schematic illustrating the source of longitudinal coherence. A simple geometrical analysis can be used to derive equation 2.18.

$$\Theta_1 = arccos\left(1 - \frac{\lambda}{\Delta R}\right) \tag{2.18}$$

The angle between the optical axis and the first maxima for typical wavelengths of 2.5 pm and atomic spacing of 3 Å this gives a maxima at $\Theta_1 = 129$ mrad. Because this maxima is due to longitudinal interference, it occurs in all directions transverse to the optical axis resulting in a concentric interference pattern.

In their report, Jesson and Pennycook employ the Warren approximation from x-ray diffraction theory to account for the effects of the thermal motion of atoms on scattering coherency [31]. The Warren approximation assumes correlated motion between near-neighbour atoms but does not give a full consideration of phononic behaviour. Figures 2.10 a), b) and c) show the thickness dependence of HAADF-STEM imaging predicted for a column of stationary atoms separated by 2 Å for two detector geometries.

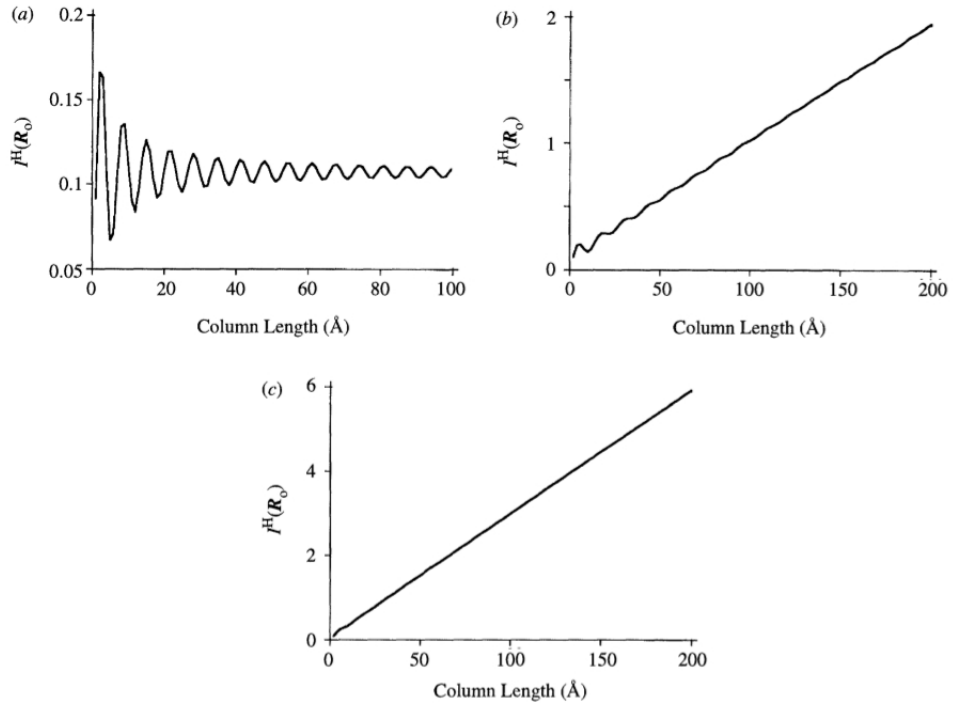

Figure 2.10: The depth dependence of a static column with atomic spacing of 2 Å for various detector ranges: a) $75 - 150$ mrad b) $75 - 200$ mrad. c) Depth dependence with thermal vibrations included for a $75 - 150$ mrad detector [31].

These results suggest significant non-linearities occur for small column lengths due to coherent interference effects. The longitudinal coherence was found to be reduced by thermal vibrations as seen in figure 2.10 but some interference effects between neighbouring atoms remains. Jesson and Pennycook suggest this could modify column intensities in typical HAADF-STEM imaging by as much as 20%.

In contrast to the analytical calculations of Jesson and Pennycook, multislice simulations by Hillyard and Silcox found that increasing the detector inner angle from 40 mrad to 120 mrad reduced the signal strength but had little effect on relative image intensities, suggesting that the signal was already incoherent at 40 mrad [32]. They also note that, in accordance with the Rutherford cross-section, equation 2.6, the scattering to high-angles is greater for atoms of larger mass. This means that selecting greater inner angles reduces the relative contrast of the lighter atoms.

2.3.7 The response of the annular detector

In order to analyse intensities in HAADF-STEM images quantitatively, it is necessary for the electrical output of the detector system to be proportional to the electron beam flux

reaching the detector throughout the experiment. The scintillating material used in typical scintillator-photomultiplier detectors has a short decay time so there should be minimal overlap of the signal from the detector as the scanning probe moves from one pixel to the next. The scintillators are also not susceptible to radiation damage so should have a constant response throughout experiments.

The output signal from the photomultiplier is typically passed to a preamplifier to convert the current signal into a voltage signal, often with a constant additional voltage offset. This voltage is then passed to an analog-to-digital converter which, in conventional commercial microscopes, produces 16-bit quantisation. Lebeau *et al.* tested the response of the annular detector on their STEM by focussing the entire electron beam onto the HAADF detector and varying the extraction voltage of the gun source to alter the beam intensity [33]. They found that the beam brightness must be selected carefully to avoid saturation of the detector output before the maximum intensity of the beam was reached, but that if this were ensured, the expected relationship between extraction voltage and beam intensity, $ln(I) \propto U_E^{1/2}$, was achieved at the preamplifier output. This is shown in figure 2.11a). There are slight deviations from the simple relationship but these are expected due to the extended Schottky emission regime at high extraction voltages and a background intensity level due to thermionic emission alone at low extraction voltages. The proportionality between detector signal and beam intensity allows images to be quantitatively analysed in a straightforward manner, and the beam flux at the detector can be determined as a fraction of the incident beam.

By scanning the beam across the annular detector a map of the response was produced, as shown in figure 2.11b). It was also found that the response is not uniform over the surface of the detector. This was ascribed to the fact that the detector does not lie in the plane normal to the optical axis, it is tilted in order to fit in the microscope. This result is important as the intensity landing on the detector is not necessarily cylindrically symmetric. Mapping this intensity variation allows it to be accounted for in quantitative simulations by multiplying the intensity at each pixel on the detector by a correction factor.

2.3.8 The number of atoms in the path of the beam

As the beam propagates through the sample it is scattered by Coulomb interactions with any nuclei within whose cross-section it passes. Thus the overall scattering intensity

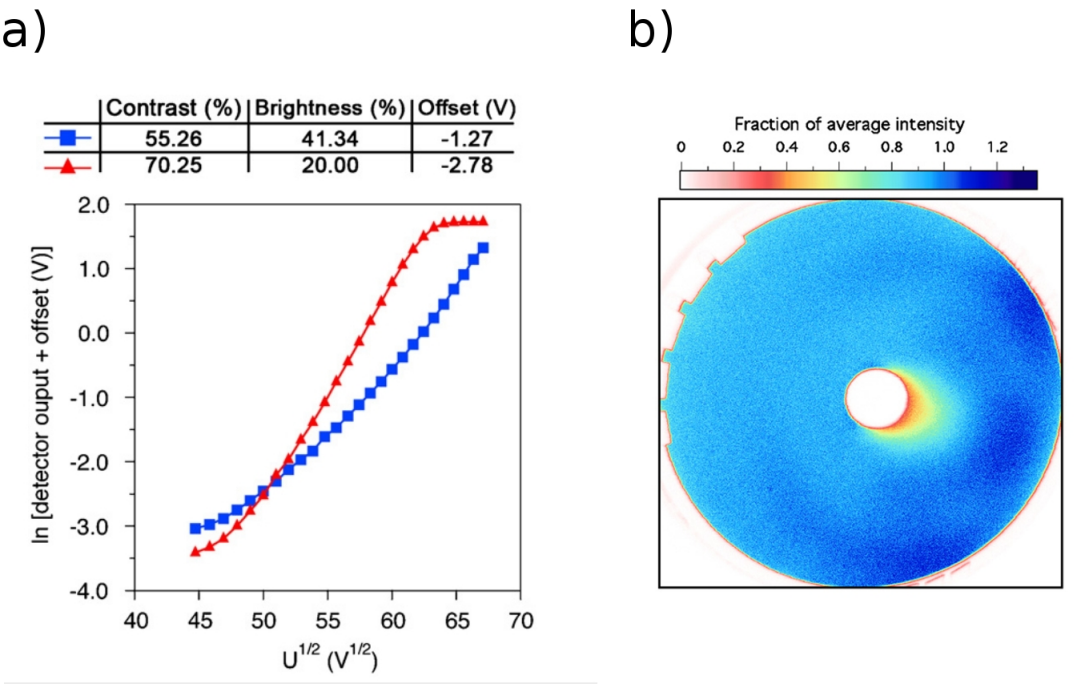

Figure 2.11: a) Response of the output voltage on the annular detector of an FEI Titan microscope with increasing gun extraction voltage. b) Map of annular detector response, non-uniformity is attributed to a tilt of the detector relative to the normal to the optical axis. Figures taken from reference [33].

should be related to the number of atoms. In HAADF-STEM, only the electrons scattered to high-angles are measured, so greater intensities are observed for thicker samples. The variation of the intensity with the number of atoms in the beam path is not trivial because the form and intensity of the beam changes as it propagates through the sample. Young *et al.* confirmed the correlation between HAADF intensity and atom count by taking images of gold nanoclusters that had been size-selected by a time-of-flight mass spectrometer with an accuracy of $\sim \pm 4\%$ [34]. Their results are shown in figure 2.12.

When HAADF-STEM images are taken with the incident beam parallel to a crystallographic zone axis the sample presents a series of columns of atoms to the scanning beam. Li *et al.* found that comparisons between the intensities of zone axis images of nanoclusters and simulated images of model clusters could be used to characterise the clusters in three dimensions with atomic resolution [35]. A good match between the simulated and experimental images suggests that the model was an accurate depiction of the real cluster, however, there were some discrepancies, as shown in figure 2.13.

This work was conducted using a time-of-flight mass spectrometer to yield clusters of 309 ± 6 atoms. This is a magic number for gold clusters as it is the number of atoms re-

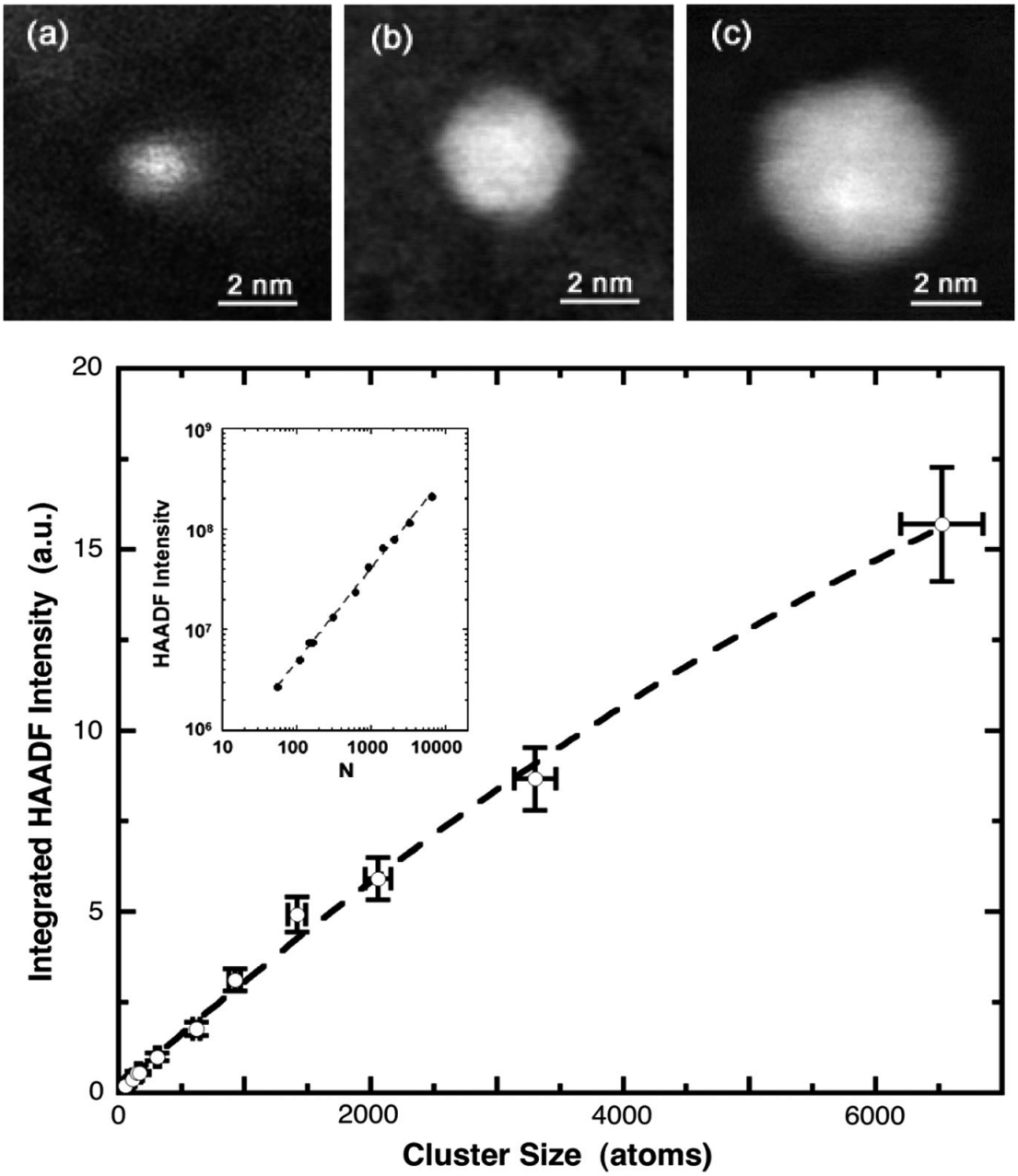

Figure 2.12: The relationship between HAADF intensity and number of atoms in beam path confirmed using a size-selected cluster source [34]

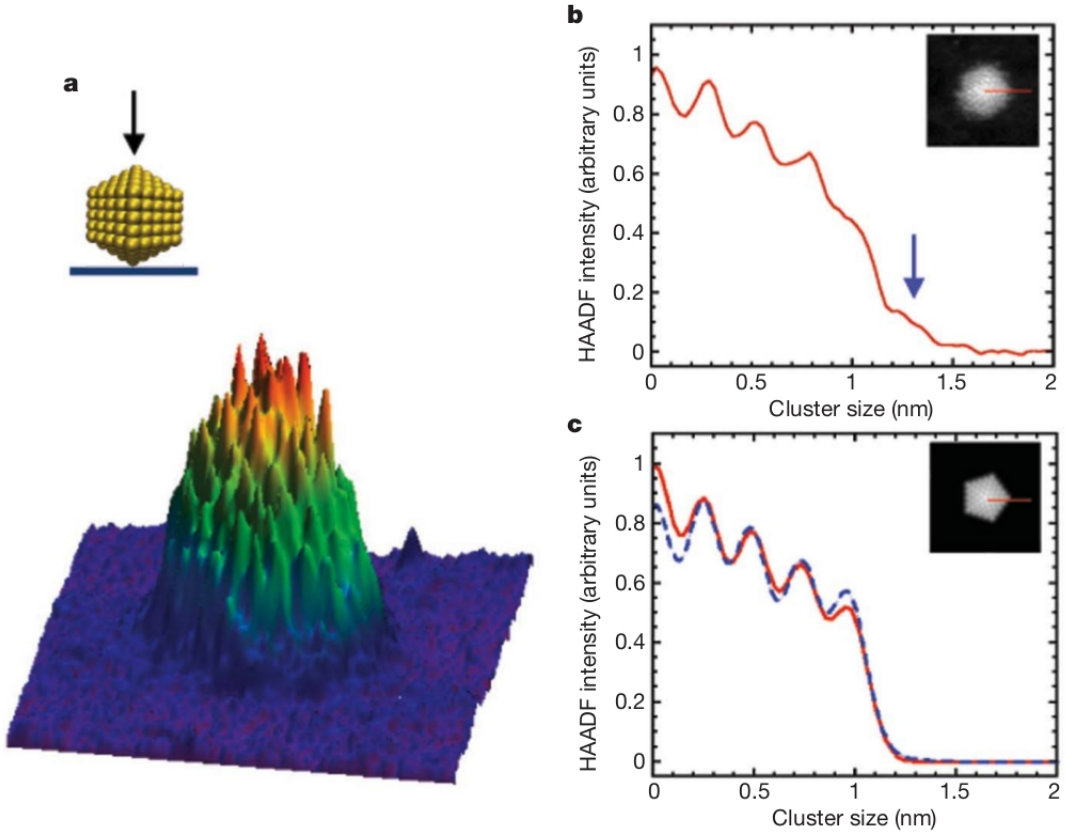

Figure 2.13: a) Model ino-decahedral nanocluster with arrow indicating the direction of the electron beam incidence. b) 3D plot of experimental image intensity. b) Experimental intensity profile along the red line indicated in the inset image exhibits a blurred outer column with a shoulder. c) Simulated intensity profile exhibits a well-defined outer column with no shoulder in both kinematical (red line) and multislice (blue line) simulations. Taken from [35].

quired to produce a perfect regular ino-decahedral cluster with five layers of atoms about the central core, as shown in figure 2.13a). The experimental intensity profile along one of the five-fold twinning boundaries, figure 2.13b), shows four intensity peaks corresponding to columns from the centre towards the surface of the cluster. In the position where the fifth peak is expected there is a broad shoulder which itself has an additional shoulder indicated by the blue arrow. The simulations, shown in figure 2.13c) do not reproduce the shoulders, rather, they have five well-defined peaks. It is also interesting to note that, whilst the line profile produced using a kinematic (single-slice) simulation yields a monotonic relationship between the peak height and the number of atoms in the column, the more rigorous multislice simulation produces a lower peak intensity for the central 9 atom column than the neighbouring 8 atom column. The difference in the shape of the surface column was attributed to the effects of thermal vibrations. The simulations were produced using a bulk mean square displacement to describe the thermal motion of all atoms in the cluster without taking into account the enhanced thermal motion at the surface (discussed in section 4.3) as MSD values were not available. The greater range of motion of the surface atoms could be expected to smear out the peak into the broader shoulder feature. In section 6.7, the experimental image is compared with a new simulation, which accounts for inhomogeneous thermal motion.

Lebeau *et al.* have further developed the atom counting methodology by introducing a technique to calibrate the annular detector so that image intensities can be expressed as a fraction of the incident beam intensity [33]. This facilitates the quantitative comparison of simulations and experiments on an absolute scale, whereas previously scaling factors were applied so that only the relative contrast of features were considered. This allows images to be interpreted without recourse to calibration standards and so offers greater insights into the image formation process, as real-world artefacts are not lost to arbitrary scaling factors. This new development was applied to counting the number of atoms in the columns of a zone-axis image of a tapered gold foil, figure 2.14a) shows the experimental image superimposed with the atom counts determined by comparison with simulations [36]. The intensity-atom count relationship is shown in figure 2.14b) .

It is noted that in the thicker region of the foil in figure 2.14a), the atom count of adjacent columns in the image plane varies smoothly with large regions of equal height and single atom steps, whilst in the thinner region many larger steps are observed. This feature is attributed to vacancies and enhanced thermal motion at the tapered edge which

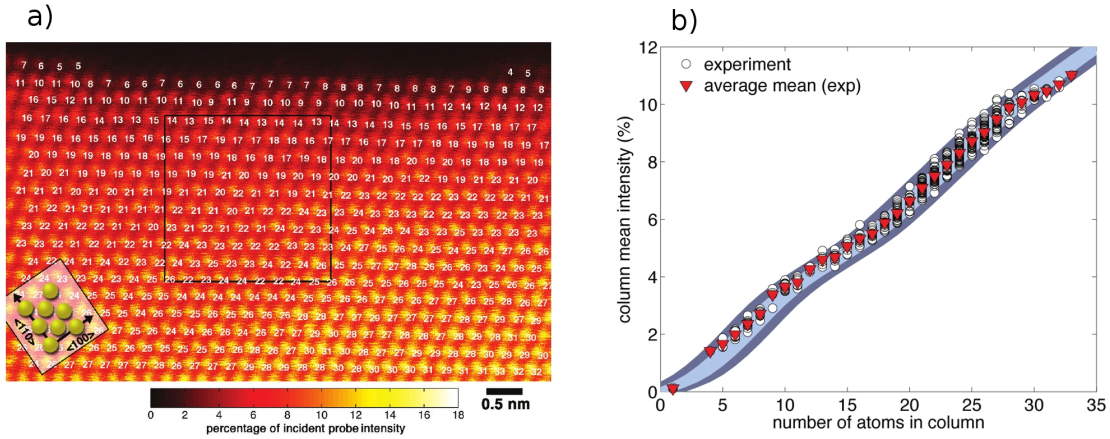

Figure 2.14: Relationship between intensity and atom count for a tapered gold foil. Experimental points and simulated trends are compared on an absolute scale due to detector calibration [36].

were not accounted for in the simulation.

An alternative method to count the number of atoms in a sample using HAADF-STEM intensities has been developed by Van Aert *et al.* [37]. In their work, the intensities of columns in zone axis images are measured to produce a statistical dataset of values which are analysed without considerations of the image formation process. Since the intensities are sensitive to atom counts which are inherently discrete, the dataset should ideally have corresponding discrete intensities. In reality, the intensity peaks are continuously distributed about mean values, due to detection noise and experimental instabilities. Van Aert uses a finite mixture modelling process to fit the intensity dataset with Gaussian peaks, each of which should correspond to a certain atom count and should ideally contain all the intensities due to columns of that size. Figure 2.15a) shows an example silver cluster embedded in an aluminium matrix. Figure 2.15b) shows a histogram of the intensity dataset from that image, fitted with Gaussian distributions.

The dataset could be fitted by any number of Gaussians but should ideally be fitted by a number, G, corresponding to G different atom count values present in the structure. In order to determine the true value of G, Van Aert tries a range of values and employs an Integrated Classification Likelihood (ICL) criterion to determine the best. The ICL criterion rewards better fit quality but penalizes increasing the number of components, it produces a minima at the optimum G value, as seen in figure 2.15c). Atom counts can then be assigned to each of the Gaussian components. If the intensity of the lowest component is close to the intensity difference between adjacent components, it can reasonably

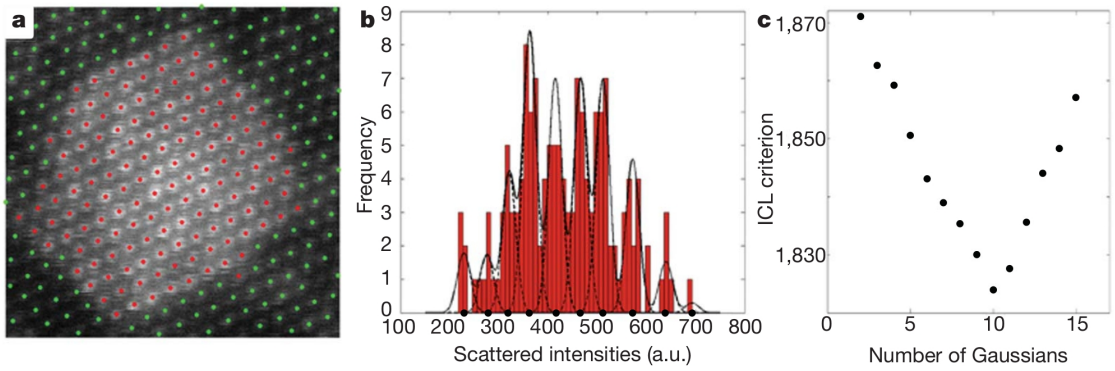

Figure 2.15: a) HAADF-STEM image of a single-crystalline silver cluster embedded in an aluminium matrix. b) Histogram of peak intensities from a) fitted with Gaussian distributions.

be assumed that the first component corresponds to a single atom. Atom counts are then assigned to the rest of the components, assuming a monotonic relationship between atom count and intensity. Once a model has been produced it can be used in multislice simulations to assess its validity. This method reduces the requirement for multiple multislice simulations which are computationally expensive. The method has been applied to a number of single-crystalline samples [37, 38, 39, 40, 41]and an ultrasmall agglomerate of $<$20 atoms [42]. The applicability of this approach to structural characterization of small multiply twinned nanoclusters is discussed in chapter 6.

2.3.9 The proton numbers of the atoms

Intensities in HAADF-STEM images are produced only by electrons which undergo high-angle scattering resulting from Rutherford-like interactions with nuclei in the sample. As the scattering is caused by Coulomb interactions, the extent to which it occurs is related to the number of protons in the nuclei, the Z number. This sensitivity has been used to produce element maps [43, 44]. Whilst these studies were qualitatively successful, the quantitative interpretation has proved troublesome. Rutherfords analysis of alpha particles suggests a scattering cross-section proportional to Z^2, however, this is a simplified interpretation which negates electronic effects. Quantitative measurements have determined a variety of Z dependencies such as $Z^{1.64}$ [45] and $Z^{1.47}$ [46], the difficulty in isolating the Z dependence from other effects has prevented a definitive determination of the relationship.

2.3.10 Thermal motion

The thermal motion of atoms in the sample is important to imaging in HAADF-STEM as it is the primary cause responsible for the incoherent nature of the measured signal. The period over which a beam electron is propagating through the sample is several orders of magnitude smaller than the frequency of thermal vibrations, so the positions of atoms in the specimen are essentially frozen throughout an individual electron's propagation. A typical beam current of 70 pA corresponds to one electron every 2.28 ns, this is approximately three orders larger of magnitude than the period of thermal vibrations. Each electron thus impinges on an effectively frozen sample with atomic coordinates which are uncorrelated with those experienced by the previous electron. This results in a predominantly incoherent intensity at the annular detector so interference effects are avoided and contrast is directly interpretable. In addition to causing incoherence, the thermal vibrations effect the intensity of high-angle scattering. The relative displacements of atoms in a crystalline column due to thermal vibrations produce a broadened scattering cross-section as illustrated in figure 2.16.

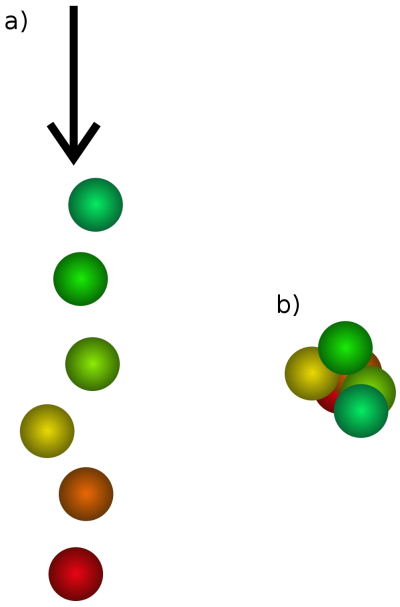

Figure 2.16: Thermal displacements broaden the total scattering cross-section of an atomic column. a) A column of atomic potentials with electron beam alignment indicated by arrow. b) Plan view illustrates enlarged cross-section.

The broader projected scattering cross-section is expected to increase the intensity at the high-angle detector [47]. This is observed in Bloch wave simulations [31] and multi-slice simulations of silicon [46]. Abe *et al.* have also reported experimental observation

of this effect in quasicrystalline decagonal $Al_{72}Ni_{20}Co_8$. In contrast, studies using multislice simulations have observed reduced intensities as the amplitude of thermal vibrations increase [19, 7]. On observing this in multislice simulations of an FCC gold sample, Kirkland notes that this is "a little unexpected" before attributing it to Bragg peaks near the detector inner-angle. The effects of thermal motion on image contrast are studied in this thesis and discussed in chapter 7. An explanation is provided to reconcile the seemingly contradictory results referenced here.

2.3.11 The geometrical arrangements of the atoms

The importance of the geometrical arrangement of atoms in crystalline structures arises from the channelling effect. Electron channelling occurs when the focused electron beam is directed onto a crystalline specimen, orientated with a zone-axis parallel to the optical axis of the microscope. The specimen then presents a series of atomic columns aligned with the incident beam. Each column has a periodically oscillating positive potential along the optical axis which will draw in the electron beam, focussing it along the column [48]. With a well focused beam, most of the electrons will be drawn into a single column, although some will channel down neighbouring columns, particularly as the beam propagates deeper into the specimen where electrons scattered by the column closest to the beam can be scattered again by neighbouring columns.

Electron channelling results in an increase in high-angle scattering as a greater proportion of the beam is brought close to the nuclei where the high-angle scattering cross-section is greatest. In experimental microscopy this effect is readily observed as a large increase in image contrast as the specimen is rotated into a zone-axis. Zone-axis images are vital to extracting 3 dimensional data from HAADF-STEM as they produce images consisting of discrete spots corresponding to atomic columns with intensities related to the number of atoms in the column.

The focussing and subsequent scattering of a channelled electron beam results in a transversely oscillating intensity as the beam propagates along the column. This effect was reproduced in multislice simulations by Voyles *et al.* [49], as can be seen in figure 2.17. In this case the beam was focused onto one of the silicon columns of a silicon dumbbell pair in crystalline bulk silicon. The beam propagates down a $<110>$ column with intensity maxima at approximately 100 and 300 Å. It should be noted that it is not

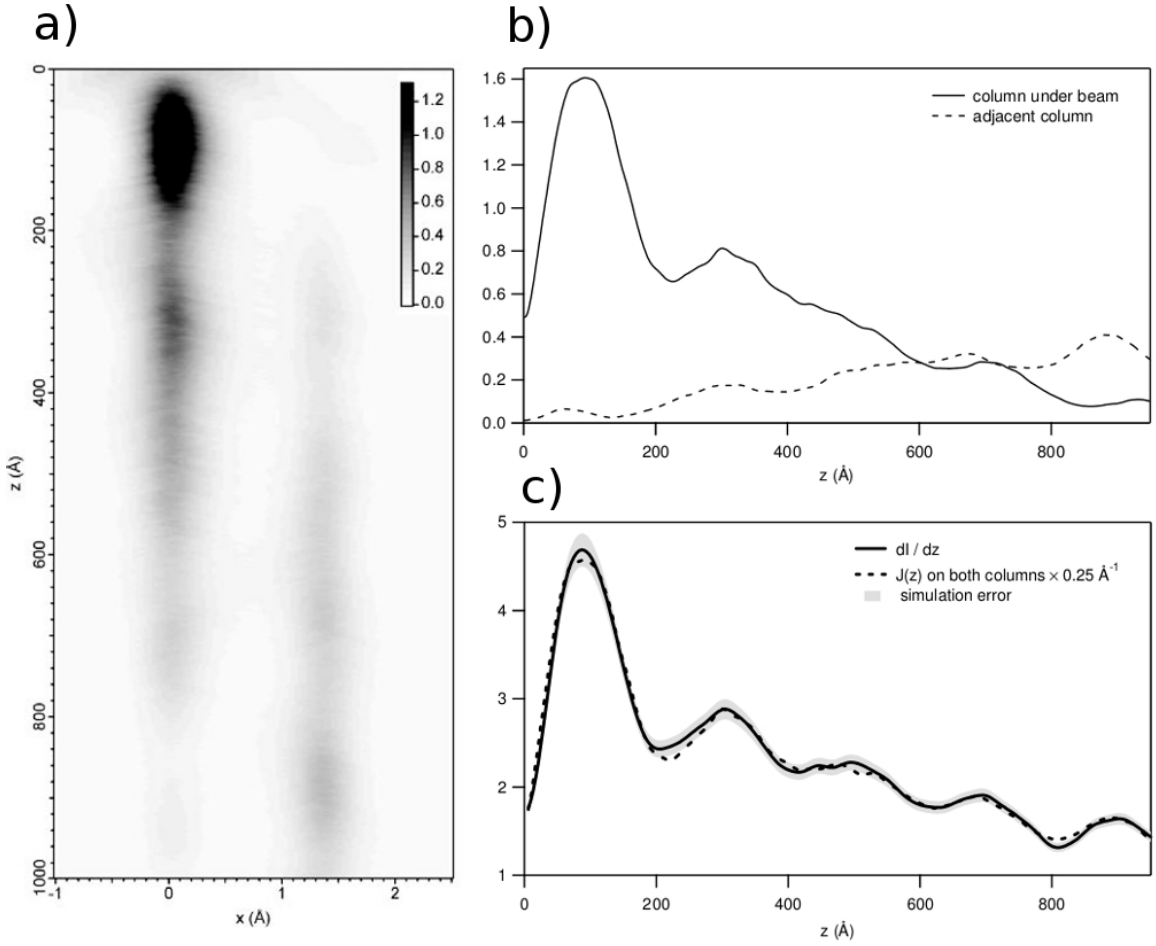

Figure 2.17: a) Simulated beam propagation in bulk crystalline silicon for a beam with accelerating voltage of 200 keV, C_s=-1.0 mm, aperture semiangle of 10 mrad and 450 Å defocus. b) Integrated wavefunction over dark-field detector angles of the aforementioned beam during propagation. c) Oscillating rate of increase in intensity at the HAADF detector as the probe propagates through the sample due to channelling. Edited from [49].

unusual for a channelled beam to partially migrate from its initial column onto adjacent parallel columns as seen in figures 2.17 a) and b). This can result in scattering from columns away from the incident beam position in thick samples. The transverse oscillation of the channelling beam can result in a non-linear relationship between scattering intensity and sample thickness as indicated in figure 2.17 c). The extent to which channelling occurs is sensitive to the form of the incident beam and the alignment of the atoms constituting the column with the beam axis. Thus, static strains in the specimen and mistilts from the zone axis of a column result in a reduction in HAADF intensities. Experimental evidence for this has been observed in HAADF-STEM images of GaAs with InGaAs quantum wells [50]. The intensity of the GaAs was observed to drop close to the interface with the quantum wells, a region in which strain is expected. By tilting the sample it was found that the intensity of the dim region to one side of the well would reduce whilst on the other side it increased. This behaviour cannot be explained by mixing at the interface or other symmetrical phenomena but is readily explained by strains with mirror symmetry for which the effect of tilting is to better align the foremost atoms of the strained columns on one side of the defect and misalign those on the other side. Multislice simulations of structures with such mirror symmetry were found to reproduce both the intensity dip and the asymmetrical tilt dependence.

Chapter 3

Computational scanning transmission electron microscopy

3.1 Introduction

In the previous section, a number of factors affecting image intensities in HAADF-STEM were described. Several of these effects are difficult to parametrize and combine in a non-trivial fashion. Consequently, it has become commonplace to employ computational simulations to aid the interpretation of experimental images. In this section, the means by which this is accomplished are discussed. There are currently two distinct simulation algorithms: the Bloch wave method introduced by Fujimoto and Kambe, and the multislice method developed by Cowley and Moodie [51, 52, 53]. In Bloch wave simulations it is approximated that beam propagation in the sample is dominated by a number of Bloch waves which match the crystalline periodicity. This method is well suited to studying crystalline structures and is less computationally expensive than multislice simulations, however, even with the inclusion of some correction terms, it is less well suited to irregular structures such as those observed at the nanoscale. The multislice algorithm is thus used for all the simulations in this work.

In this section a brief overview of the entire multislice algorithm is given, before a more detailed discussion of each step including a thorough mathematical description. The multislice simulation of HAADF-STEM images begins by mathematically reproducing the form of the focused electron probe beam incident at a point on the sample surface. The sample is split into a number of slices perpendicular to the incident beam, as indicated in figure 3.1.

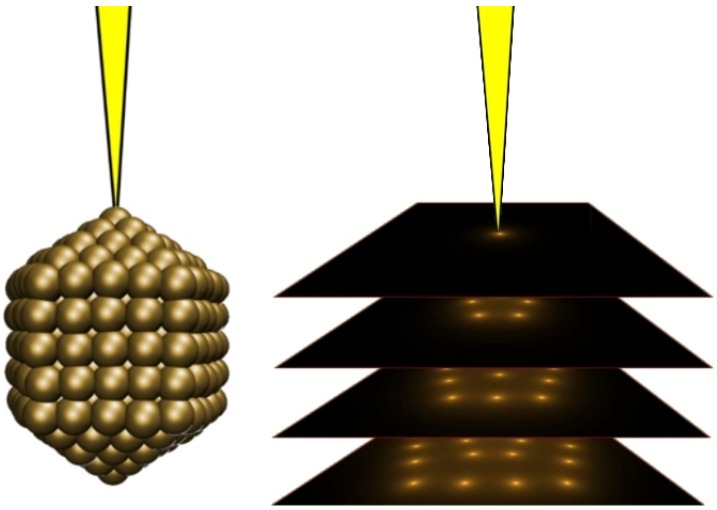

Figure 3.1: Schematic illustrating the principle of the multislice algorithm in which the potential of each atom in the sample is projected onto a 2D slice with which the electron beam interacts before propagating to the subsequent slice until it has passed through the sample whereupon it is propagated to the far-field.

In splitting the sample into discrete slices, each atom is shifted into the plane of the nearest slice. The effective Coulomb potential of each atom in a slice is determined and combined with the others in that slice. This results in a series of slices containing the projected 2D potential of those atoms nearest each particular slice. The incident probe is then modified to account for its interaction with the foremost potential slice in the sample, this takes the form of a phase change in accordance with the weak phase object approximation (discussed in section 3.2.2). After interaction with the first slice, the beam is propagated the short distance to the next slice before interacting with that slice. This process is repeated until the beam has propagated through the entire specimen, whereupon it is propagated into the far field. The intensity of the beam wavefunction in the far field is then calculated and summed over the range of the detector. A new incident wavefunction is then generated for the next position of the scanning beam and this process is repeated until calculations have been performed at all the incident scanning positions to produce an intensity map. For reference, a flow-diagram of the multislice algorithm is shown in figure 3.2.

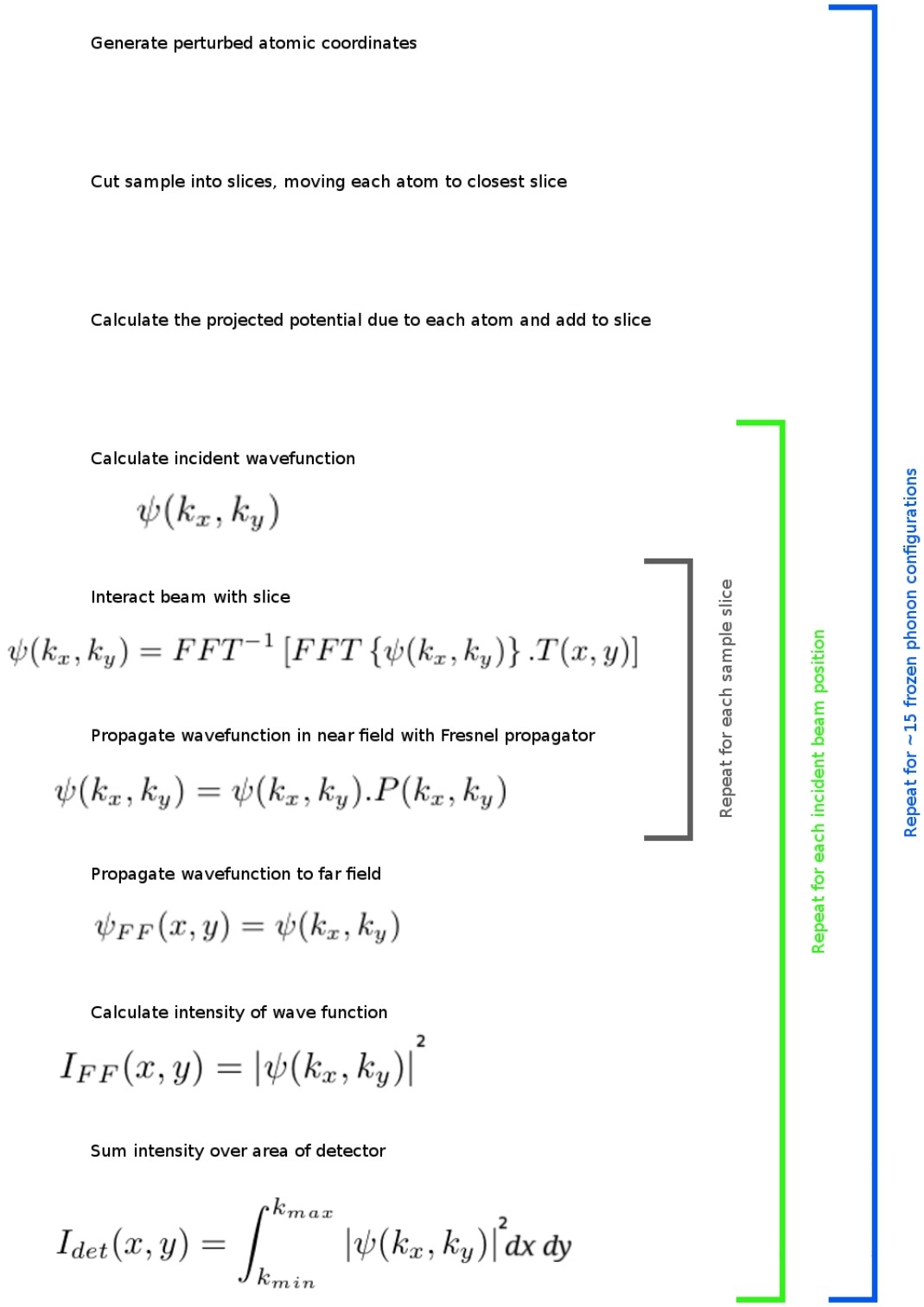

Figure 3.2: Flow diagram explaining the algorithm of multislice simulations.

3.2 The Multislice algorithm

3.2.1 Calculating the incident probe wavefunction

The incident probe is generated taking into account the acceleration voltage, defocus, aperture and aberrations of the microscope. The accelerating voltage determines the wavelength of the electron beam as given by equation 3.1. This equation includes relativistic effects, as the electron beams used in most measurements are travelling at a significant fraction of the speed of light.

$$\lambda = \frac{hc}{\sqrt{eV(2m_0c^2 + eV)}} \tag{3.1}$$

The wavefunction at a position x_p on the top surface of the sample is given by a superposition of plane waves as in equation 3.2.

$$\psi_p = A_p \int_0^{k_{max}} e^{i2\pi k(x - x_p)} \tag{3.2}$$

In which each plane wave is indexed by it's wavevector, k, up to k_{max} which is the largest wavevector allowed by the convergence aperture. The wavefunction is normalised to unity intensity with the normalization constant, A_p, before interacting with the sample so that any intensity component measured is expressed as a fraction of the incident intensity. The wave function is constructed in reciprocal space for convenience, as the plane waves constituting the focused wavefunction are spatially separated in reciprocal space, that is, each position in a reciprocal space array corresponds to a different k-vector. The array positions are thus given complex numerical values in accordance with equation 3.3.

$$\psi_k = cos(2\pi k) + isin(2\pi k) \tag{3.3}$$

The range of k values included in the array is limited by the width of the objective aperture, α, setting the limit given by equation 3.2.1.

$$k_{max} = \frac{\alpha}{\lambda} \tag{3.4}$$

Equation 3.3 produces an ideal focussed wavefunction which must be modified to account for defocus and lens defects. This is achieved by the addition of a phase term, χ, to yield the final wavefunction, equation 3.5.

$$\psi_p = A_p \int_0^{k_{max}} e^{i2\pi k \cdot (x - x_p)} e^{-i\chi} \tag{3.5}$$

Any defocus in the incident beam is simulated by an addition to the phase of each of the convergent plane waves in reciprocal space. The change to the phase increases with the spatial frequency of the plane wave, so that the beam converges in a shorter distance. The phase change due to defocus is given by equation 3.6.

$$Df = \Delta F \lambda k^2 \tag{3.6}$$

Where ΔF is the distance between the top plane of the sample and the focal point of the incident beam. The effect of spherical aberration is to make beams at the outer extent of lenses converge in a shorter distance than those closer to the optical axis. This is simulated by adding increasingly large offsets to the phase of plane waves further from the optical axis i.e. those with greater wavevectors, as in equation 3.7.

$$Sph = -\frac{1}{2}\pi C_{S3} k^4 \lambda^2 \tag{3.7}$$

Chromatic aberrations are not accounted for in these multislice simulations. The spread of energies in the electron beam before interaction with the sample is small and so it is not particularly susceptible to chromatic aberrations. Additionally, in HAADF-STEM

the beam is not focussed after the sample so the energy spread accrued in propagating through the sample is not converted into a chromatic aberration.

Two and three fold astigmatisms lead to an azimuthal phase variation in the beam. They are accounted for by including the phase corrections in equations 3.8 and 3.9.

$$Ast1 = -\lambda k^2 \Delta F_{a2} sin(2(\phi - \phi_{a2})) \tag{3.8}$$

$$Ast2 = -\frac{2}{3}\lambda^2 k^3 \Delta F_{a3} sin(3(\phi - \phi_{a3})) \tag{3.9}$$

In which ΔF_{a2} and ΔF_{a3} are the magnitudes of the two and three fold astigmatisms, respectively, and ϕ_{a2} and ϕ_{a3} are their azimuthal directions. Combining lens defects and defocus as in equation 3.10 gives the total phase modification terms,χ.

$$\chi = Sph + Chr + Df + Ast1 + Ast2 \tag{3.10}$$

The finite effective size of the electron source causes a broadening of the beam in the image plane. This is conventionally accounted for in simulations by a Gaussian convolution of the final image. There is, however, no justification in assuming that the broadening due to the incoherent source should be have a Gaussian distribution. Indeed, Verbeeck et al. found that a combination of a Gaussian and bivariate Cauchy distribution was better suited, providing a better match to the extended tails of the beam profile [54].

3.2.2 Constructing sample slices

To construct the potential slices, each atom in the sample is shifted into the plane of the nearest slice. The scattering factor of each atom is interpolated from a look up table of values derived from quantum mechanical calculations of various kinds, depending on the atom. A description of these methods is outside the scope of this work, but a brief discussion with references is available in reference [7]. The scattering factors of all the atoms

are superimposed on a real space array known as the trasmission array. The scattering factors used in this method are for isolated atoms and so do not account for bonding effects, however, bonding should have a negligible effect on the potential as interactions with the nuclei are the dominant cause of high-angle scattering.

3.2.3 Beam-sample interaction

The dominant interaction of the sample with the electron beam is the Coulomb interaction between the beam electrons and the sample nuclei. For thin samples, or thin slices of a sample, this results in a small deviation in the beam path which is well approximated by an adjustment to the magnitude of the k-vector of each plane wave making up the focused wavefunction as in equation 3.11.

$$\frac{1}{\lambda_s} = \frac{[(eV + eV_s)(2m_0c^2 + eV + eV_s)]^{1/2}}{hc} \tag{3.11}$$

In which λ_s is the wavelength accounting for the increase to the electrostatic potential energy of the beam whilst in the specimen, given by eV_s. This corresponds to a phase change in the total wavefunction as per equation 3.12.

$$\psi(\mathbf{x}) \sim exp(2\pi i k_z z) exp(i\sigma V_s z) \tag{3.12}$$

Where σ is the interaction parameter given in equation 3.13.

$$\sigma = \frac{2\pi}{\lambda V}\left(\frac{m_0c^2 + eV}{2m_0c^2 + eV}\right) \tag{3.13}$$

The approximation that the effect of the beam-sample interaction is solely a phase change to the wavefunction is called the weak phase object (WPO) approximation. This approximation is reasonable so long as the specimen potential is much smaller than the beam energy, which is true of almost all TEM systems [7]. The second exponential term in

equation 3.12 is also known as the transmission function, as it describes the phase change due to transmission through a slice of the sample. When the slice is sufficiently thin that the WPO approximation can be used, the z dependence in the transmission function is negligible and can be removed as in equation 3.14.

$$t(\mathbf{x}) = e^{(i\sigma v_z(x))} \tag{3.14}$$

The beam-sample interaction is thus calculated by multiplying the transmission function array by the beam wavefunction in real space. This causes a phase shift in the wavefunction corresponding to elastic scattering of the beam.

The thermal vibrations of the sample are of great importance to HAADF-STEM imaging as they are responsible for minimising coherent scattering and producing the dominant thermal diffuse scattering (TDS) contribution to the images. Thermal vibrations are accounted for in multislice simulations using the frozen phonon algorithm developed by Loane and Silcox [48]. In this algorithm, the final image is produced by taking an average over a series of multi-slice images, each with slightly distorted atomic coordinates. The high velocities of electrons in the incident probe beam result in a short interaction period between each incident electron and the sample. It is thus a reasonable approximation that the coordinates of atoms in the sample are frozen during the propagation period. The relatively large period between incident electrons ensures that the configurations observed by subsequent electrons are uncorrelated. Thus the use of averaging over time-independent interactions gives a reasonably good approximation of the time-dependent system. The position shifts applied to the atoms are determined using pseudo-random selections weighted by a Gaussian distribution. The width of the distribution is given by the mean square displacement of the atoms, which are conventionally determined from X-ray diffraction experiments. In the conventional frozen phonon algorithm, the perturbation to the atom positions are calculated independently with no correlated phonons, this is the Einstein oscillator approximation. The credibility of this approximation has been evaluated by Muller *et al.* by using the phonon dispersion curves of silicon to produce more accurate, but still random, atomic displacements [55]. The difference between the correlated and independent oscillator simulations for HAADF-STEM images was found to be negligible, though it was significant for convergent beam electron diffraction (CBED) patterns. In the literature the effects of thermal motion are often referred to in terms of the Debye-Waller

factor, a term taken from the field of x-ray diffraction. This can lead to some confusion, as discussed in appendix 1.

A major deficiency of the frozen phonon algorithm is that it does not account for momentum or energy transfer between the sample atoms and the electron beam. Forbes *et al.* have reported on the use of a rigorous many-body quantum mechanical approach to address this issue [56]. Their work used a model similar to the Born-Oppenheimer model to take into account inelastic interactions. This was found to produce numerically similar results to the frozen phonon algorithm, providing justification for the use of the simpler semi-classical model. In this work, the applicability of the conventional frozen phonon algorithm to studies of nanoclusters at atomic resolution has been assessed and is discussed in chapter 6.

3.2.4 Beam propagation

Between each slice in the multislice algorithm the beam must be propagated a short distance, of the order of several angstroms, to the next slice. This small propagation falls within the regime of Fresnel optics and so it is described by the Fresnel propagator function derived from the Fresnel-Kirchhoff diffraction integral, equation 3.15.

$$\psi(x, y, z + \Delta z) = \frac{1}{2i\lambda} \int \psi(x', y', z) \frac{e^{(2\pi i R/\lambda)}}{R} (1 + \cos\theta) dx' dy' \tag{3.15}$$

This gives the wavefunction at a position $\psi(x, y, z+\Delta z)$ based on the sum of the wavefunctions at the previous slice, $\psi(x', y', z)$. This integral works in an analogous manner to Huygen's principle of spherical wavelets, with θ being the angle between the plane of the wavefront before propagation and the outgoing wavefront at any position $(x, y, z + \Delta z)$ and R being the distance between them. For the electron beams typically used in scanning transmission electron microscopes, this angle is small as the beam does not deflect far from the optical axis. This allows for the use of the paraxial approximation given by equation 3.16.

$$1 + \cos(\theta) \approx 2R \tag{3.16}$$

If the distance between slices is sufficiently small, the distance between the origin of a wavelet and any position of the propagated wave, R, must also be small. This allows for the further approximation given by equation 3.17.

$$\begin{aligned} 2R &= \Delta z\sqrt{1+(x-x')^2/\Delta z^2+(y-y')^2/\Delta z^2} \\ &\approx \Delta z\left[1+0.5(x-x')^2/\Delta z^2+0.5(y-y')^2/\Delta z^2+...\right] \end{aligned} \tag{3.17}$$

These approximations give the approximated Fresnel-Kirchhoff integral used in multislice calculations, equation 3.18.

$$\begin{aligned} \psi(x,y,z+\Delta z) =& \frac{1}{i\lambda}\frac{\exp\left(2\pi i\Delta z/\lambda\right)}{\Delta z}\times ... \\ & \int \psi(x',y',z)\exp\left(\frac{i\pi}{\lambda\Delta z}\left[(x-x')^2+(y-y')^2\right]\right)dx'dy' \end{aligned} \tag{3.18}$$

For computation, this can be expressed as a convolution of the wavefunction with a propagator function, as in equations 3.19.

$$\begin{aligned} \psi(x,y,z+\Delta z) &= \exp\left(2\pi i\Delta z/\lambda\right)\left[\psi(x,y,z)\otimes p(x,y,\Delta z)\right] \\ &\text{where} \\ p(x,y,\Delta z) &= \frac{1}{i\lambda\Delta z}\exp\left[\frac{i\pi}{\lambda\Delta z}(x^2+y^2)\right] \end{aligned} \tag{3.19}$$

This convolution can be efficiently calculated by exploiting well optimised fast Fourier transform libraries as described in section 3.3.1.

After propagating through the sample to the exit surface, the beam must be propagated to the detector plane in the far field. This is achieved by performing a fast Fourier transform on the exit wavefunction in accordance with Abbe's theory.

Electron channeling occurs when the beam is focussed down a column of atoms. This increases the intensity of high-angle scattering by increasing the beam intensity close to

the nuclei. This effect is automatically included in multislice simulations as the beam electrons can be scattered once by each slice. Often more simple simulations are used in which the entire potential of the specimen is projected onto one slice. These kinematic simulations are sometimes used as they are less computationally expensive. This can prove useful in studies where sacrificing accuracy for computationally efficiency is not important, see for example reference [57]. However, for accurate quantitative analysis, multislice simulations must be used to include this effect.

3.2.5 Beam detection

Once the beam has been propagated into the far-field it is necessary to determine the magnitude of the signal at the HAADF detector. This is achieved by taking the complex magnitude of the wavefunction and summing it over all pixels within the range of the HAADF detector. This corresponds to limiting the wavevector values. To allow for absolute comparisons between experiments and simulations the intensities at the detector should be calculated as a fraction of the incident beam intensity.

Any non-uniformity in the experimental detector can be recreated in the simulation by multiplying the magnitude of the wavefunction at the detector by a correction function array. The correction function array must be determined by measuring the response of the detector as discussed in section 2.3.7.

3.3 Computational details

3.3.1 Fast Fourier transforms

Fourier transforms play a key role in the multislice simulation technique, they are used to transform a function between real-space and reciprocal space and so provide a computationally efficient means to perform convolutions and to simulate the propagation of the beam to the far-field. The forward Fourier transform is given by equation 3.20, the inverse transform is calculated by applying the forward transform and then applying conjugation and scaling as shown in equation 3.21.

$$FT[f(x)] = F(k) = \sum_{x,y} exp(2\pi ikx) f(x) dx \tag{3.20}$$

$$FT^{-1}[F(k)] = f(x) = \sum_{x,y} exp(2\pi ikx) F(k) dk \tag{3.21}$$

In this work, discrete Fourier transforms are performed using fast Fourier transform (FFT) algorithms. Two dimensional transforms are calculated through a succession of 1D transforms given by equation 3.22.

$$F_n = \sum_j f_j exp[2\pi i(nj/N)] \tag{3.22}$$

The number of operations involved in performing such a transformation scales with array size as N^2 because N sums of N terms must be calculated. A simple radix-2 FFT such as that of Cooley and Tukey [58] improves the efficiency of transforms of sizes factorizable by 2. This is achieved by avoiding repeated calculations by storing results which will be needed again. The summation in the discrete Fourier transform can be split into two, one for the even terms and another for the odd terms, as shown in equation 3.23

$$F_n = \sum_{m=0}^{N/2-1} f_{(2m)} exp(-\frac{2\pi i}{N}(2m)n) + \sum_{m=0}^{N/2-1} f_{(2m+1)} exp(-\frac{2\pi i}{N}(2m+1)n) \tag{3.23}$$

The twiddle factor, $W_N = e^{-2\pi in/N}$, is key to minimising the calculations needed to solve a Fourier transform as it has both symmetrical and periodic properties, as shown in equations 3.24 and 3.25.

$$W_N^{n+N/2} = -W_N^n \tag{3.24}$$

$$W_N^{n+N} = W_N^n \tag{3.25}$$

The twiddle factor can be factorised out of the odd term summation and $W_N^2 = W_{N/2}$ can be substituted to yield equation 3.26.

$$F_n = \sum_{m=0}^{N/2-1} f_{(2m)} W_{N/2}^{mn} + W_N^n \sum_{m=0}^{N/2-1} f_{(2m+1)} W_{N/2}^{mn} \tag{3.26}$$

Exploiting the periodic nature of $f_{(2m)}$ and $f_{(2m+1)}$ with period N/2 and substituting equation 3.24 gives equations 3.27 and 3.28.

$$F_n = F_e(n) + W_N^n F_o(n), n = 0, 1..., \frac{N}{2} - 1 F_{n+N/2} \tag{3.27}$$

$$F_n = F_e(n) - W_N^n F_o(n), n = 0, 1..., \frac{N}{2} - 1 \tag{3.28}$$

This approach to optimization is called the decimation-in-time algorithm, an N point Fourier transform has been determined from two N/2 point calculations with a reduction in the total number of calculations from N^2 to $N^2/2 + N/2$. Higher orders of decimation can be used, for example, if N/4 point DFTs are first computed, N/2 point DFT can be derived from them. The decimation can be continued, so that for an array of $N = 2^v$ points, a $v = log_2 N$ order decimation is used. In this case the number of operations is reduced from N^2 to $Nlog_2 N$. In this work, N=4096 size transforms are frequently used, a radix-2 FFT reduces the number of operations to perform this task by a factor of $341\frac{1}{3}$.

3.3.2 Parallel processing and general purpose graphics programming

In parallel processing, two or more processing cores are used simultaneously to perform independent calculations. In principle this could reduce the real-world time needed to

perform N_c operations on N_p processors by a factor of N_c/N_p. In reality, the overheads of memory allocations and data transfers reduce the efficiency of parallel programming. In recent years CPUs containing a number of parallel cores have become commonplace in consumer computing. On inspecting the multislice algorithm, it is clear that there are a number of independent calculations that are suitable for parallel processing, for instance, the propagation of the electron beam from each incident beam position is independent of the others. This can be exploited to reduce the computation time of multislice simulations making them accessible to users lacking access to supercomputers.

The computer gaming industry is at the forefront of developing highly parallel processing architectures in the form of graphical processing units (GPUs). These provide the processing power to yield high frame rates in increasingly complex computer games. In 2007 the GPU manufacturer Nvidia released a CUDA software development kit allowing programs to be written and compiled to run on their graphics cards. This allows developers access to levels of parallel processing which were previously only available in large supercomputers.

For this work, two computers were designed to accommodate Nvidia Tesla C2075 GPUs. Each GPU has 448 processing cores allowing a peak dual precision performance of 5.15×10^{11}) floating-point operations per second. Each of the GPUs have 6GB of RAM, enough to store the large arrays used in multislice simulations of non-periodic structures. A multislice program based on Kirklands method has been produced to make use of the GPUs as well as any available parallel CPU cores [7]. The GPUs are used to perform fast Fourier transforms as the 2D transforms can be split into independent 1D transforms and calculated in parallel before being combined to yield the 2D transform. This is implemented using the CUFFT library which is a library of highly optimised Fourier transform functions designed for GPUs [59]. The multiplication of the wavefunctions with propagator functions and transmission functions are also performed on the GPUs.

A number of simulations have been conducted to compare the speed of the program using the CPU and GPU with that of the CPU alone. In these simulations, a test sample made up of 31 slices is used with varying probe and transmission array sizes. In all cases the output image size is 64 pixels. Both simulations scale approximately linearly with the number of slices and the number of output pixels but differently with probe and transmission array size. The relationship between probe and transmission array size, and calculation time for the two programs can be seen in figure 3.3.

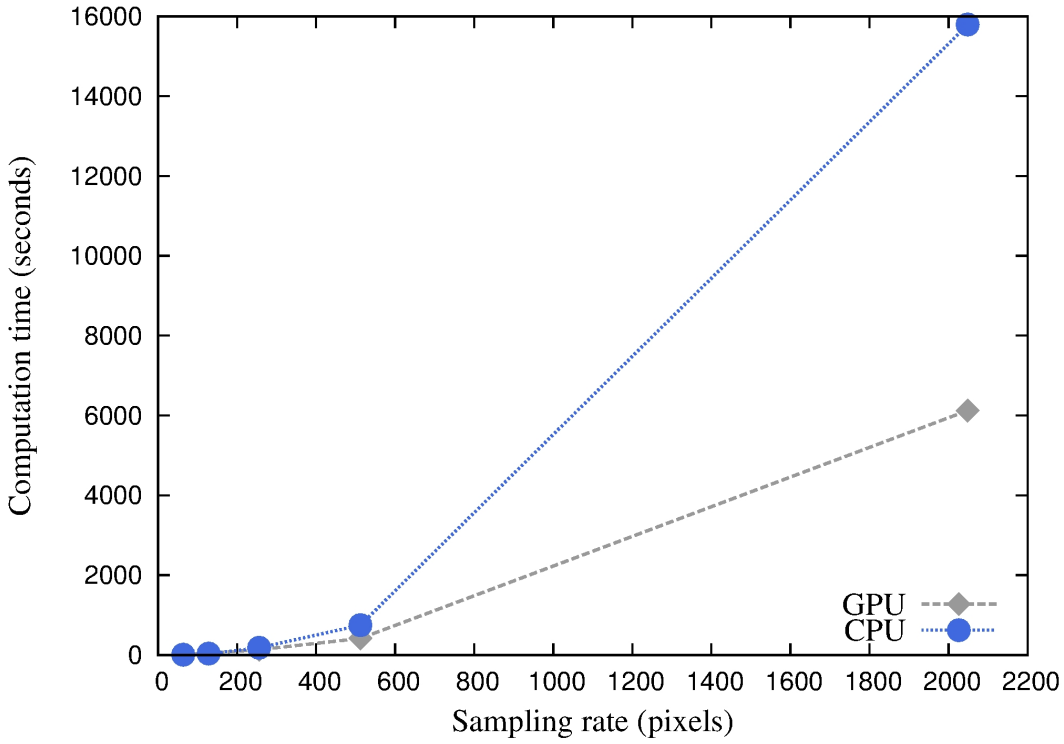

Figure 3.3: Processing time for multislice simulations using conventional CPU only and CPU + GPU.

For the array sizes of 64^2 and 128^2, the overheads of copying data from the motherboard memory to the memory on the GPU outweigh the benefits of parallel processing. At array sizes above 256^2 pixels, the GPU program becomes quicker. For 2048^2 arrays, the GPU program is 2.6 times faster than the CPU program.

3.3.3 CuSTEM

The GPU enabled multislice simulation program used throughout this work, CuSTEM.cu, is included in appendix 2, and is available on the included CD. Two function libraries are required, Tiffsubs.c and Slicelib.cu. These are included on the CD but are not listed in the appendix because, with the exception of some data types, they remain the same as those published by Kirkland [7]. The program includes hard-coded values for the number of Cuda threads per block, in this case 16, which is suitable for the Nvidia Tesla C2075 used for this work. This value may need to be adjusted for alternative GPUs. The program includes a Cuda kernel to perform the convolution between the wavefunction and the transmission function that describes the sample potential, a bandwidth-limiting kernel, and a propagation kernel. These kernels, combined with the discrete fast Fourier transfrom func-

tions from the CUFFT library allow the *'transmission-propagation-transmission-...'* cycle of the multislice algorithm to be performed without transferring information between the motherboard (host) and GPU (device) memories [59]. The transmission function arrays and initial wavefunctions are generated in host memory before being copied to the GPU as these processes do not lend themselves to parallelism, particularly on the GPU, due to the large overhead of parameter data transfers. After the GPU portion of the program is complete, the exit wavefunctions are copied to the host memory to calculate the complex magnitudes in order to determine the HAADF signal intensity.

Chapter 4

Gold nanoclusters

4.1 Introduction

Metallic nanoclusters are interesting subjects for study both commercially and for the insights they offer into fundamental physics. Their primary commercial use is in the catalyst industry where they offer greater economy than bulk materials, and in some cases only function in nanocluster form. The small dimensions of nanoclusters place them in the regime where quantum mechanical phenomena become important, so they facilitate the observation of exotic behaviours unknown to our everyday macroscopic experiences. In this work, gold nanoclusters will be used as a prototypical sample, the techniques developed should be equally applicable to any heterogeneous samples for which molecular dynamics potentials of good quality are available.

The earliest known nanoclusters are thought to have been produced by dissolving the bulk metal in glass at high temperatures which, under cooling, precipitated into colloidal nanoclusters. This was achieved by the Romans in the 4th century to produce the dichroic glass Lycurgus cup seen in figure 4.1.

More recently, Faraday used phosphorus to reduce gold chloride, yielding colloidal nanoclusters [61]. Modern synthesis concentrates on producing nanoclusters with large yields, small size distributions, and high stabilities. A large variety of production methods have been developed, a review of these techniques can be found in reference [62]. Further discussion of this subject is beyond the scope of this work. In the following section a description of the pertinent properties of gold nanoclusters is given to explain the motivations for conducting this work and provide the background knowledge necessary to understand

Figure 4.1: 4th Century dichroic glass cup. Gold nanoclusters exhibit surface plasmon resonances resulting in differing colours of transmitted (red) and reflected (green) light . Taken from [60].

the research chapters which follow.

4.2 Morphologies

In bulk, gold forms a face-centred cubic crystalline structure. In the small nanocluster size range, of the order of a few nanometers, the low coordination of surface atoms induces a shift to pseudo-spherical morphologies which minimize the surface to volume ratio. The three major morphologies observed in this size range are the icosahedron, the Ino decahedron and the cuboctahedron as shown in figures 4.2 a), b) and c) respectively.

Of these morphologies, the icosahedral offers the minimum surface to bulk ratio so it is favoured at small sizes. However, this structure deviates from the preferred FCC crystalline configuration of gold. As the cluster size increases this internal strain begins to dominate the surface energy and Ino decahedral clusters become energetically favourable [64]. Decahedral structures can be constructed from a combination of five platonic tetrahedra, each of which is compatible with the FCC crystal structure, so that the internal strain is lower than in icosahedra. The five tetrahedral segments each have a dihedral angle of 70.53^o, so there is a small missing section of 7.35^o as shown in figure 4.3 introducing a strain energy.

For clusters of $\sim$ 2-3 nm diameter the truncated Ino decahedron offers the best compromise between surface energies and crystallographic strain. A mutated form of the Ino decahedral morphology also occurs in this size range: The Marks decahdedral structure.

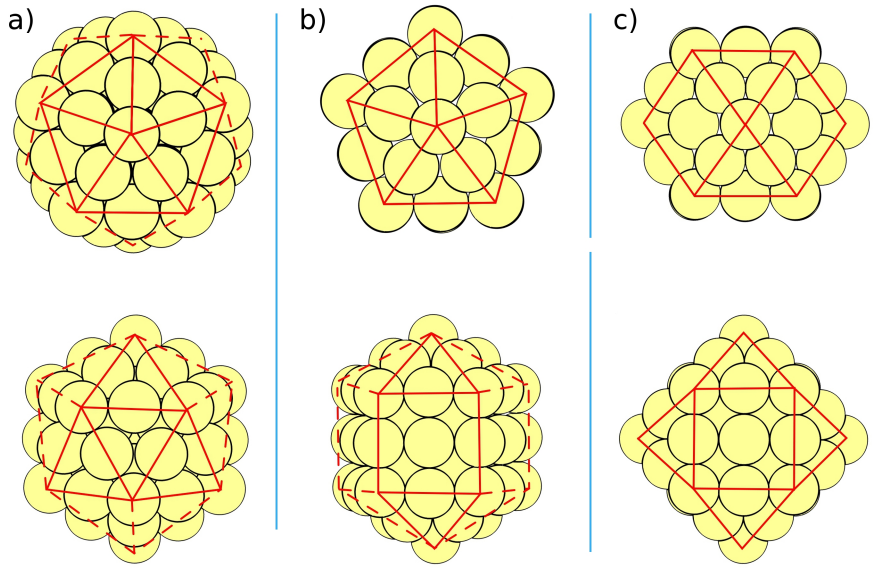

Figure 4.2: Regular pseudo-spherical cluster morphologies a) Icosahedron, b) Ino decahedron, c) Octahedron. Adapted from [63].

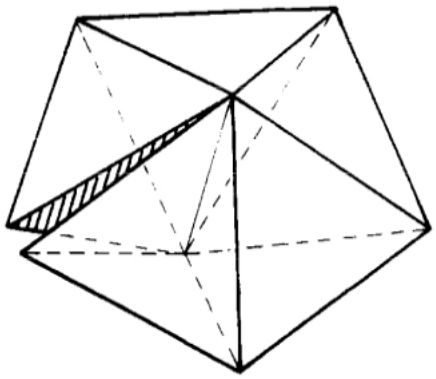

Figure 4.3: Decahedron constructed from five tetrahedra with a small missing section. Adapted from [65].

A geometrical model of a Marks decahedral structure is shown in figures 4.4a) and b) alongside a HAADF-STEM image of a real gold cluster, figure 4.4c). This structure was first observed by Marks in annealed silver clusters [66]. During aggregate synthesis, additional atoms join the cluster at the most favourable nucleation sites, however, the annealing process allows the cluster to reconstruct to the optimum energy structures. The Marks decahedral structure is favoured in this case because it reduces the size of (100) surfaces in favour of lower energy (111) surfaces. This is consistent with molecular dynamics simulations which predict that the extent to which the reentrant facets occur depends on the energy difference between the (100) and (111) surfaces [64]. For gold, the (100)/(111) relaxed energy ratio was found to be 1.15 using a Gupta potential, thus Marks decahedrals with reentrant facets are expected in the small, multiply-twinned size regime.

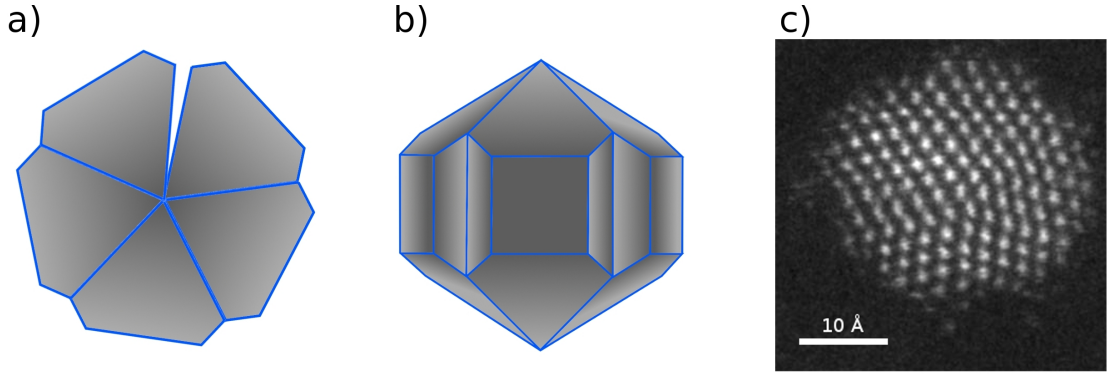

Figure 4.4: Marks decahedron with reentrant faceting plan view along a) [110] zone axis, b) [001] axis. c) HAADF-STEM image of Marks decahedral gold nanocluster produced by Dr. D He, University of Birmingham.

At larger sizes the internal strain due to the missing angle overcomes the surface strain and octahedral morphologies are favoured as they are truncated single crystalline FCC structures with little internal strain. Barnard, *et al.* have used DFT simulations to produce a phase map showing the favoured structural motifs at various temperatures and cluster sizes, this is shown in figure 4.5 [67].

As mentioned previously, clusters are typically synthesised in an aggregative process in which atoms join the cluster at the most favourable site when they aggregate. This does not necessarily lead to optimum energy structures. Koga *et al.* have surveyed the morphologies present in cluster samples before and after annealing at a range of temperatures, their results are shown in figure 4.6 [68]. The initial sample is almost exclusively populated by icosahedra (Ih), with very few decahedra (Dh), at diameters below 5 nm. The Ih:Dh ratio decreases to 3:2 at $\sim$ 18 nm. This is consistent with the theoretical predic-

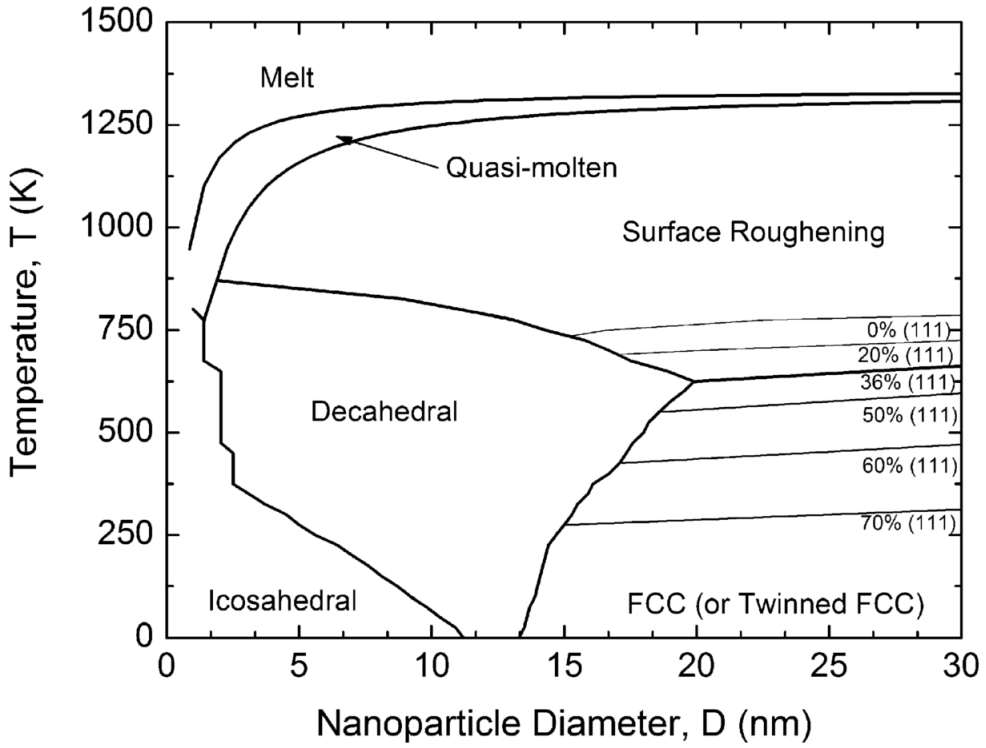

Figure 4.5: Phase map showing favoured gold cluster morphologies predicted by DFT simulations. Taken from [67].

tions that the reduced crystallographic strain overcomes the surface energy at larger sizes. After annealing at 1173 K (figure 4.6b)), the majority of the clusters smaller than 6 nm had transitioned from Ih to the more favourable Dh structures, between 6 nm and 18 nm the clusters are predominantly Ih as the annealing temperature did not provide enough energy for the larger clusters to transform, though the size effect is responsible for a gradual increase in the population of Dh at larger sizes. After annealing at 1223 K and 1273 K, figures 4.6c) and d), larger clusters have acquired sufficient energy to transform from Ih to Dh resulting in a cross-over at 7.5 nm. For diameters less than 5 nm, the population is a closer balance of Ih and Dh morphologies which suggests that the annealing temperature was high enough to melt some of the smaller clusters so that they transformed back into Ih from Dh. After annealing at 1373 K, approximately 35 K above the bulk melting point, there is an almost equal population of Ih and Dh at diameters < 5 nm, the population of Ih drops off steeply to almost zero at 7.5 nm whilst the population of Dh peaks at this size, making up ~ 0.9 of the population with the rest being twinned and octahedral FCC clusters. These population graphs show that, whilst calculations can determine the most energetically favourable structures, the make-up of real-world samples depends strongly on the preparation method.

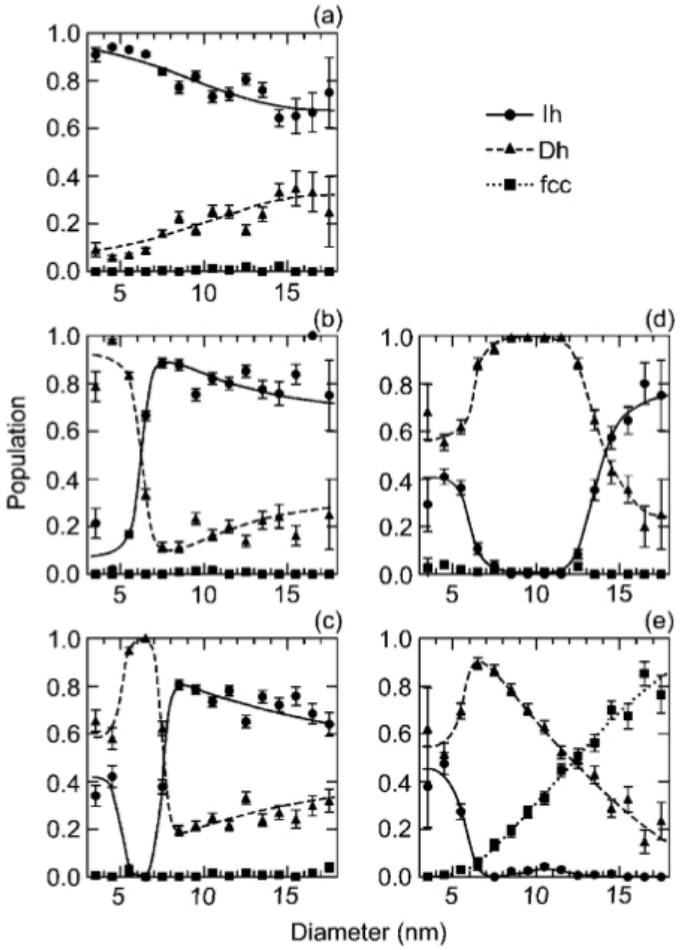

Figure 4.6: Fractional populations of icosahedral, decahedral and FCC clusters. a) Before annealing, b) after annealing at 1173 K, c) after annealing at 1223 K, d) after annealing at 1273K and e) after annealing at 1373 K [68].

The missing angle in decahedral clusters causes a disclination strain which has been experimentally measured by applying geometrical phase analysis to a $\sim$ 15 nm diameter decahedral gold cluster [69]. In this work it was found that the 7.35^o missing angle was accounted for by a combination of internal lattice rotation (4.3^o) and shear strains. It should be noted that decahedral clusters of this size may not be representative of the more common decahedral clusters at diameters $<$ 5 nm. More recently, Walsh *et al.* used aberration-corrected TEM to produce atomic resolution zone axis images of $\sim$ 3 nm decahedral nanoclusters in order to quantify the strain in the projected image plane [70]. They measured a compression relative to the bulk lattice constant at the very centre of the cluster but a relatively unstrained interior. The surface of the cluster exhibited expansions of $\sim 6\%$.

4.3 Thermal motion

As discussed in section 2.3, thermal motion has a significant effect on the intensity in high-angle annular dark field scanning transmission electron microscopy. If the intensity of HAADF-STEM images is to be used to deduce the number of atoms in the sample perpendicular to the plane of the projected image, the effect of thermal motion must be accounted for. In bulk samples the thermal motion is approximately uniform throughout the sample and the effects of elevated values at surfaces is negligible. The thermal motion can then be measured readily using x-ray diffraction [71]. It can also be deduced with good accuracy using phonon density of states measurements obtained from neutron scattering experiments [72]. This work, however, is conducted at sub-Ångstrom resolutions where surface columns are distinguished from interior columns and the effects of atoms at the upper and lower surfaces of the thin samples are not negligible. It is thus necessary to determine or estimate the thermal motion with a comparable resolution in order to establish the effects it has on image intensities.

Molecular dynamics simulations predict depressed melting points for clusters of smaller sizes using Quantum Sutton-Chen (QSC) [73], Embedded Atom Model (EAM) [74], Baskes' modified EAM [75], Glue [76], and Gupta [77] potentials. Figure 4.7 shows a compilation of melting point values taken from the literature.

All the simulations predict a similar trend between cluster size and melting point, the causes of the discrepancies between the values cannot be identified as the simulations

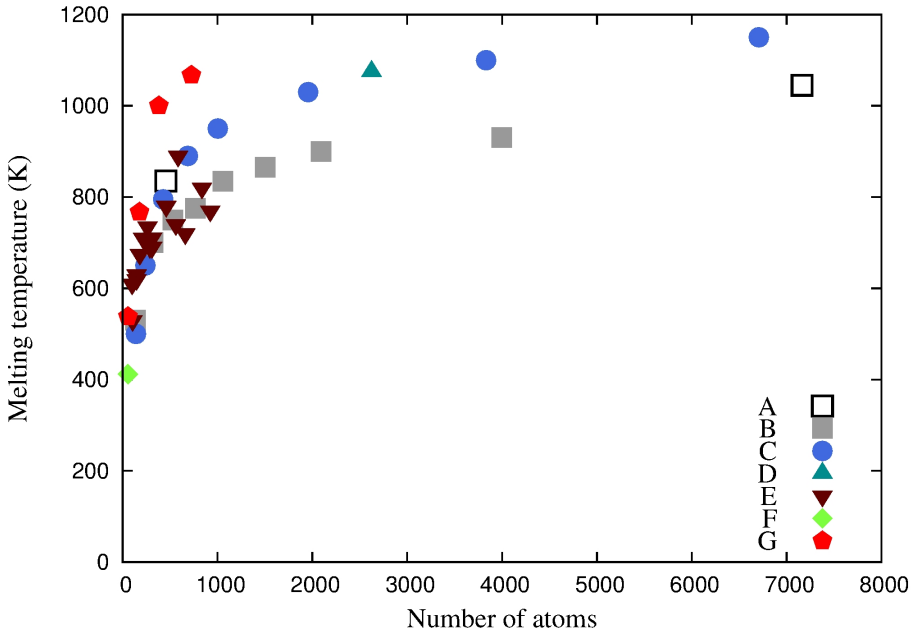

Figure 4.7: Melting points of Au nanoclusters of varying sizes and morphologies as predicted by molecular dynamics simulations. A-[73], B-[74], C-[75], D-[78], E-[77], F-[77], G-[79].

used both different potentials and a variety of different cluster morphologies. In Li's work alone, five different morphologies were used, resulting in the large spread of melting points for clusters of similar atom count [77]. In several of these studies the dynamics of different layers in the cluster were inspected [80, 74, 73, 75, 78]. In all cases, this revealed that surface pre-melting occurs prior to the global phase transition. Figure 4.8 shows the average mean square displacements of atoms in different layers of a 2624 atom icosahedral gold nanocluster [78]. Note that the scale of the MSD axis varies between the plots. At 600 K the cluster is in the solid phase, the outer two surfaces are substantially more mobile than the interior but their movement is smaller than the nearest neighbour distance of ~ 2.88 Å . At 900 K the outer two layers have pre-melted whilst the atoms in the interior show little net motion. By 1100 K the global phase transition has occured with saturated MSDs for all atoms. An interesting feature of the 600 K plot is the seemingly larger thermal motions of the first sub layer compared to the surface. This is attributed to the averaging process; the first sub layer contains a greater proportion of edge and vertex atoms than the outer surface and so has a greater mean MSD. It is expected that the mean MSD of the surface, discounting edges and vertices, would be greater for the outer surface.

The predictions of depressed melting points and enhanced thermal motion at cluster surfaces are supported by experimental evidence. Buffat measured the intensity of elec-

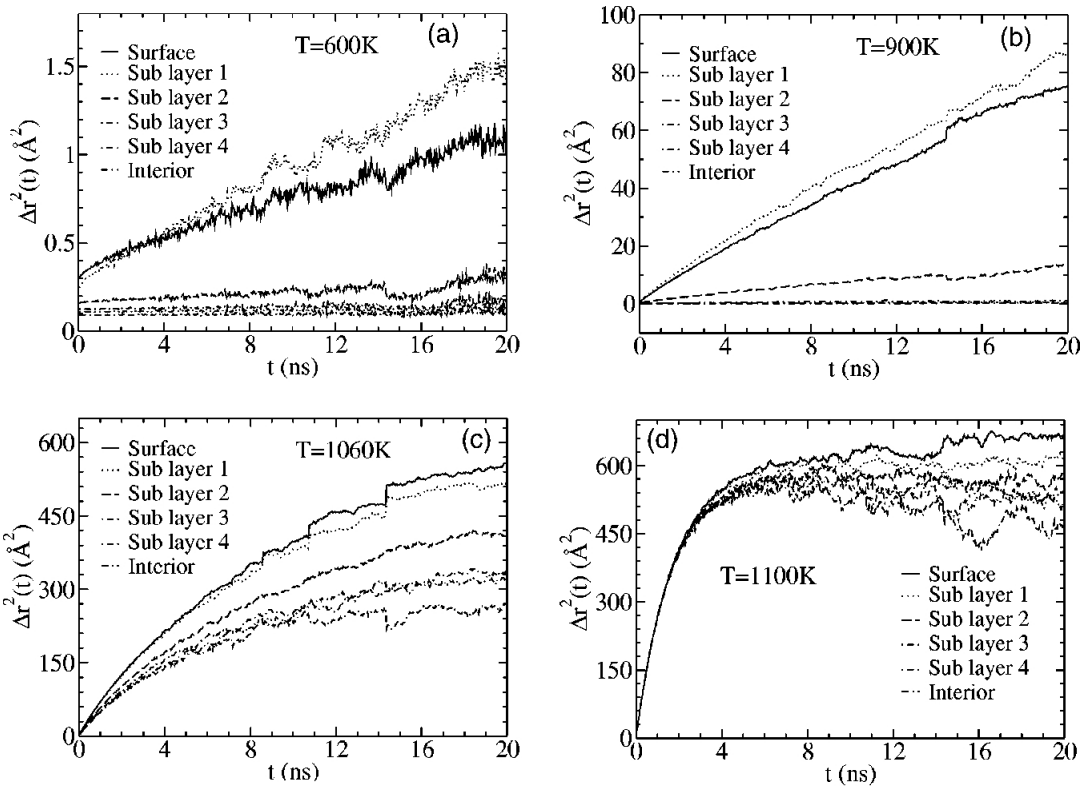

Figure 4.8: Average mean square displacements of atoms in different layers from the surface of 2624 atom gold icosahedral cluster. Taken from [78].

tron diffraction patterns whilst increasing the sample temperature in order to observe the melting point, the results are shown in figure 4.9 [81]. In this work Buffat also reported observing surface reconstructions and sintering in samples of gold nanoclusters at room temperature. The images in this work were produced using high-resolution transmission electron microscopy so it is likely that the electron beam irradiation will have contributed to the thermal energy of the clusters during in this experiment, enabling these transformations.

4.4 Catalysis

The use of nanoclusters as industrial and domestic catalysts has been an important factor in driving nanocluster research. The most obvious benefit of using nanoclusters in catalysis is that smaller particles provide a greater surface to volume ratio. Assuming approximately spherical particles, the surface area of a particle is proportional to the radius squared whilst the volume is proportional to the cube of the radius, thus the surface area to volume ratio is proportional to $1/r$. Only the surface of the catalyst is active in the

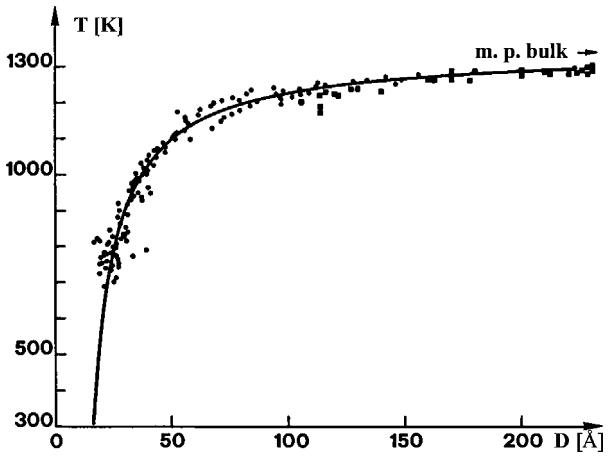

Figure 4.9: Experimental observation of depressed melting point in gold nanoclusters. Taken from [81].

reaction so it is economically beneficial to minimise the volume of catalyst needed. In studying nanoscale catalysts it has become apparent that the improved efficiency of small nanoclusters is greater than can be accounted for by the increase in surface area alone. Indeed, it has been observed that gold, which is almost chemically inert in bulk form, is an effective catalyst for various reactions in nanocluster form [82, 83, 84]. The relationship between the size and catalytic activity of gold clusters on a titanium dioxide support was investigated by Lai *et al.* yielding the results in figure 4.10 [85].

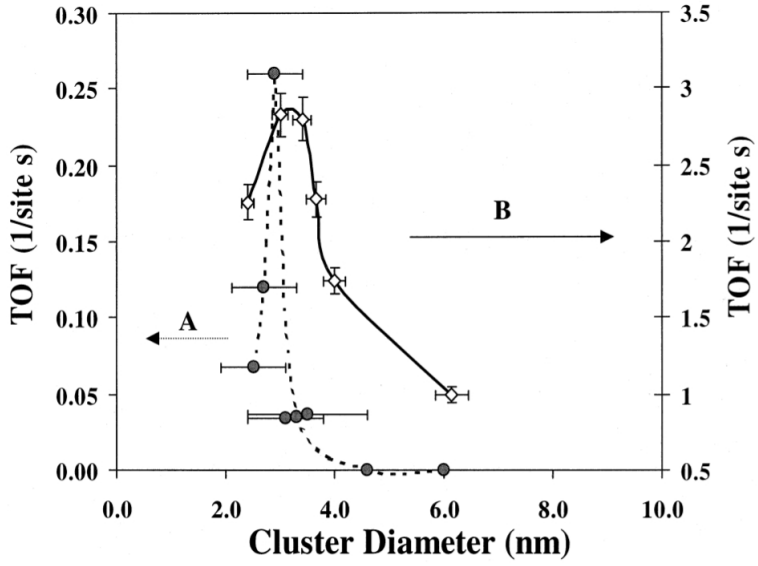

Figure 4.10: CO oxidation turnover frequency of gold nanoclusters of varying sizes on titanium dioxide supports at ~ 350 K, 40 Torr. Taken from [85].

This observation has prompted a plethora of investigations into the possible causes of the unexpected catalytic activity, however, a definitive answer has not yet been found.

Lopez *et al.* consider surface roughness to be the primary contribution to the catalytic activity of nanoclusters [86]. It is established that surface roughness due to steps and surface add-atoms increases activity in bulk catalysts, and nanoclusters are, by nature, very rough with many low coordination atoms and only small clean crystalline surfaces. Lopez *et al.* discount the possibility of any contribution from the support based on a review of the literature, which suggests that carbon monoxide oxidation turnover increases with decreasing particle size at a similar rate irrespective of the composition of the catalyst support, as shown in figure 4.11a). They claim further support for the importance of low coordination atoms based on density functional theory calculations which predict reduced oxygen and carbon monoxide binding energies for structures with lower bond coordination, as shown in figure 4.11b). As the binding of the reactant to the catalyst is a vital step in catalysis, it follows that a reduced barrier to binding should increase catalytic turnover.

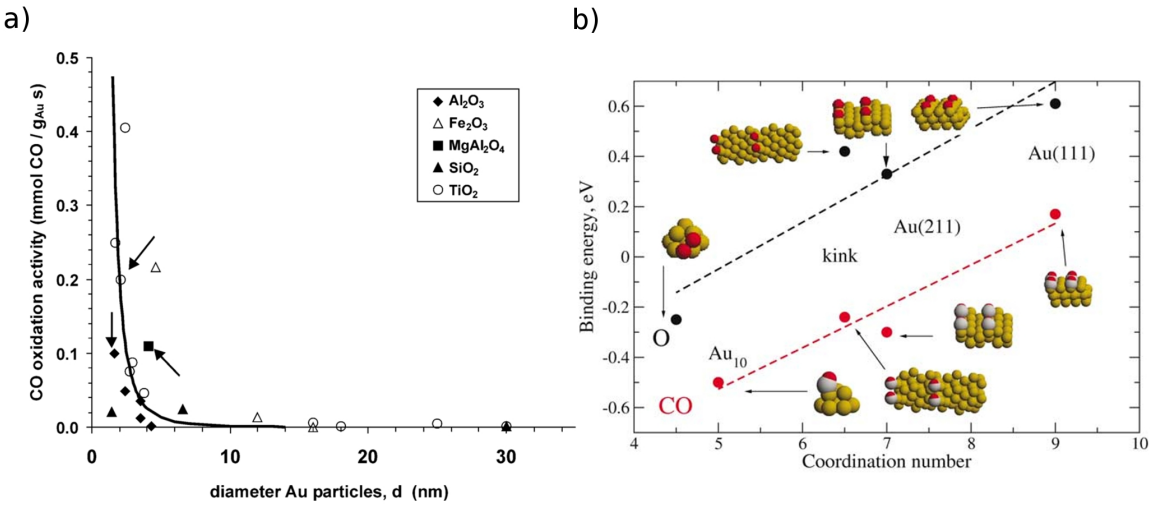

Figure 4.11: a) A collection of CO oxidation frequencies in the presence of gold nanoclusters on varying supports suggests that the particle-support interaction is not a major influence on catalytic activity. b) Reduced binding energies for low-coordination atoms intimates that this is the cause cause of catalytic activity in otherwise noble metals. Taken from [86].

As a counterpoint, Chen *et al.* have observed catalysis in gold mono- and bi-layer samples deposited on titanium oxide with turnover frequencies similar to the highest observed for pseudo-spherical clusters. This prompted them to suggest that the height of the cluster from the metal oxide is the primary factor in determining catalytic behaviour [87]. This hypothesis has been supported by Herzing who observed that, for clusters deposited on an iron oxide support, the catalytic activity was solely attributable to small bilayers of $\sim$ 10 atoms [88]. Chen's work found that gold monolayers exhibited lower catalytic activity

than bilayer counterparts despite the former having lower bond coordination. In addition, they found that for islands below a certain size the catalytic activity begins to drop, this has also been observed by Lai, as shown in figure 4.10 [85].

It has also been proposed by Hakkinen *et al.* that the catalytic activity in gold nanoclusters may be caused by their inherent instability [89]. There are a number of structural isomers with similar energies in the small nanocluster size range which are separated by small potential barriers. The instabilities of the structures at finite temperatures will allow clusters to reconstruct during the catalytic process. It was experimentally observed that Au_8 clusters were catalytically active whilst DFT simulations suggest that O_2 adsorption onto an Au_8 cluster is only possible if the cluster is allowed to reconstruct. Hakkinen also reported that partial charge transfer from a MgO support at oxygen vacancy sites was needed to pin and activate the small gold clusters. Further support for the importance of the cluster-support interaction has been produced by Yan *et al.* who deposited gold clusters of ~ 4 nm diameter onto MgO supports that had been annealed at different temperatures [90]. The concentration of oxygen vacancies is known to increase with annealing temperature and the vacating oxygen atoms tend to be ionised, leaving behind trapped electrons in the vacancies. These sites are named farbe-centers and are able to pin gold clusters and provide charge transfer. Yan's work found that the rate of carbon monoxide oxidation increased following a strong correlation with the concentration of these farbe-centers, as shown in figure 4.12.

The possibility of the catalyst support playing a directly active role in catalysis has been suggested [91], but experimental studies have shown that catalysis still occurs when the support is entirely coated and inaccessible to the reactants [87].

Density functional theory investigations into the effects of strain on molecular oxygen adsorption at Au(111) surfaces have been performed by Xu and Mavrikakis [92]. They found that a 10% stretching of bonds on a (111) surface allows weak adsorption where it would otherwise not occur. They also found that a 10% stretching on (211) surface exhibited a similar effect and caused a reduction in the O_2 dissociation barrier from 1.12 eV to 0.63 eV. In a review of the origin of catalysis in gold nanoclusters, Lopez, Mavrikakis, *et al.* opined that this set an upper bound on the contribution of strain and concluded that it is less significant than the changes in binding energy due to low-coordination as seen in figure 4.11b). However, a recent study of the strain in small decahedral nanoclusters suggests that strains greater than 10% could be present at the cluster surface [70]. In this work

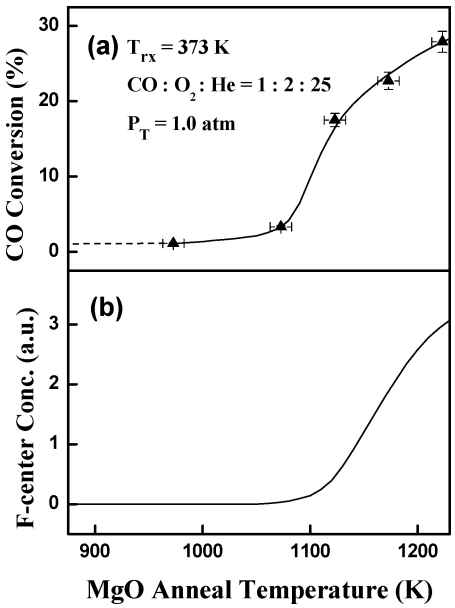

Figure 4.12: Catalytic activity in $\sim$ 4 nm gold clusters shows a strong correlation with the number of farbe-centres on the MgO supports, this is attributed to pinning and charge transfer effects. Taken from [90].

DFT simulations were also used to determine an exponential relationship between carbon monoxide absorption and strain, suggesting that strain may offer a significant contribution to the catalytic activity of gold nanoclusters.

There is a consensus in the literature that the concentration of low-coordination atoms is a dominant factor in the inert-to-catalytic transition of small nanoclusters. Whilst the support does not appear to be involved in the catalytic process directly, the interaction between the cluster and support appears to be important in activating the catalytic properties of the cluster via charge transfer. Strains in the nanocluster would appear to contribute to catalytic activity but to a lesser extent than the aforementioned effects. A more comprehensive review of current research into the catalytic activity of gold has been produced by Cuenya [93]. Many of the factors suggested as possible causes for the catalytic behaviour of gold nanostructures are interrelated. The interaction between the cluster and the support affects not only the epitaxial relationship but consequently the strain and morphology of the cluster, and, as a result, also the electronic band structure. Elucidating the fundamental cause(s) of the catalytic activity is thus difficult and high precision experiments are required to isolated all the factors which could contribute to it. The solution of this problem offers substantial motivation for developing a means to characterise small nanoclusters in three dimensions with atomic resolution.

The sub-Ångstrom resolution of high-angle annular dark field scanning transmission microscopy makes it an obvious choice to study nanoclusters. The image formation process is, however, dependent on a number of structural parameters, including strains and thermal motion. Small multiply-twinned nanoclusters have large, inhomogeneous strains and enhanced thermal motion, particularly at surfaces. In order to correctly interpret HAADF-STEM images it is necessary to understand the effects of these parameters. In this work, molecular dynamics simulations are used to generate these structural properties and multislice calculations are used to determine their effect on HAADF-STEM images.

Chapter 5

Molecular dynamics simulations

5.1 Introduction

Molecular dynamics (MD) simulations model the dynamic behaviour and interactions of atoms. The workings of these simulations will not be described in detail here, however, a brief overview will be given and details pertinent to this work will be discussed.

In the simple MD simulations used in this work, a model system is constructed as a series of atomic coordinates. Each atom is assigned a potential function used to calculate its interactions with other atoms. In heterogeneous systems, several potentials can be assigned to account for the interactions of different species, in this work the model consists only of gold atoms, so only one potential term is required. At the beginning of the simulation the internal energy of the system is defined, often via the temperature as this is a readily measurable entity. The atoms in the system are pseudo-randomly assigned initial velocities in accordance with the Maxwell-Boltzmann distribution. With the initial atomic positions and velocities defined, the simulation begins. The atoms are allowed to travel with their initial velocities from their initial positions for a short time period of the order of 1 femtosecond. After this period the interactions of the atoms are calculated and the forces acting on each atom are determined. Newtons equations of motion are used to determine new velocities for each atom and they are allowed to propagate again for another short time step. This process of propagation over a short time step before recalculation of velocities is then repeated for the duration of the simulation.

Molecular dynamics simulations are often split into an equilibration period and a production period. In both these periods the simulation runs as has been described. The

equilibration period occurs at the beginning of the simulation in which the system hasn't reached a steady state. For example, the randomly assigned velocities at the start of the simulation do not realistically represent those that would be found in an inhomogeneous structure, such as a nanocluster, in which coordination and local strain conditions will affect the mobilities of the atoms. The atom at the core of the cluster would not be expected to have the same dynamical behaviour as those on the surface. Over time, the repeated recalculating of forces and velocities will result in a redistribution of the internal energy of the system into a more physically realistic state. This can be observed by monitoring the internal energy and temperature of the system. When the energy is more realistically distributed (or, more accurately, distributed in good accordance with the potential used, which may or may not be realistic) the system reaches a steady state. The overall temperature of the system can also be adjusted in the equilibration period by a scaling of the atomic velocities until the desired temperature is reached.

Once the system has reached equilibrium the simulation goes into the production period. It is in the production period that information is extracted from the system. In the case of this work, realistic atomic coordinates are extracted at intervals throughout the simulation as will be discussed further in chapters 6 and 7.

5.2 Ensembles

Molecular dynamics can be run in one of several named ensembles, in which different system parameters are held constant. The most commonly used is the microcanonical (NVE) ensemble, in which the number of atoms, volume, and energy of the system are held constant. The alternatives include the canonical (NVT) ensemble, in which the number of atoms, volume, and temperature are held constant, and the isothermal-isobaric (NPT) ensemble in which the number of atoms, pressure, and temperature are held constant. These are often preferred as the temperature and pressure are more readily controlled experimentally.

For the simulations used in this work it is desirable to hold the temperature at a known constant temperature in order to reproduce experimental conditions. This suggests the use of the NVT ensemble, however, this ensemble maintains a constant temperature by introducing a thermostat which compromises the simulation for this work. Thermostat algorithms function by scaling the velocities of the atoms in the simulation in some man-

ner. This velocity scaling occurs between propagation time steps, so the velocities will be altered in a non-physical discontinuous matter, rendering measurements of thermal vibrations non-physical. For this reason, the NVE ensemble is used for all simulations in this work. Fortunately, the classical system acts in accordance with the virial theorem given by equation 5.1.

$$2\left\langle E_{kin}\right\rangle = -\sum_{j=1}^{N}\left\langle \mathbf{F_j}\cdot\mathbf{r_j}\right\rangle \tag{5.1}$$

The virial theorem relates the total time averaged kinetic energy, $\langle E_{kin}\rangle$, of a stable system of N particles to the potential energy, $\langle \mathbf{F_j}\cdot\mathbf{r_j}\rangle$, which binds them. As the form of the interaction potential is known, the portion of the total system energy which is kinetic, as a time-average, can be calculated. The energy in a constant energy simulation can then be selected to give a desired kinetic energy. The system also adheres to the equipartition theorem given in equation 5.2.

$$T = \frac{2\left\langle E_{kin}\right\rangle}{3k_B} \tag{5.2}$$

Which relates the time averaged kinetic energy of an individual atom to the system temperature, T, via the Boltzmann constant, k_B, where each atom has three degrees of freedom. It is thus possible to approximately define the temperature by choosing the appropriate system energy, though the temperature is chosen only as a time average and will fluctuate throughout the simulation as kinetic energy is translated into potential energy and vice versa.

5.3 Atomic potentials

The accuracy of a molecular dynamics simulation is largely dominated by the quality of the atomic potentials used to generate interaction forces during the simulation. The accuracy of simulations of bulk metal systems is well established, however, nanostructures

introduce new effects with their high proportion of low coordination surface atoms. Unfortunately, it is difficult to assess the quality of these simulations through comparisons with experimental data due to limitations in the precision with which the properties of nanostructures can be measured. Consequently, the accuracy of molecular dynamics simulations are often considered with respect to more rigorous first principles density functional theory simulations which include electron effects more extensively. These more extensive simulations are computationally expensive, particularly for large systems of heavy atoms, such as the gold nanoclusters used here. Molecular dynamics simulations make use of simpler empirical potentials which are less computationally expensive and can be performed on desktop computers in reasonable times.

The bonding in metals is dominated by the Coulomb attraction between delocalized conduction electrons and metal ions. The cohesive properties are thus strongly related to the electron density of states [94]. The density of states can be described by its moments, the first moment is the mean energy of the band, the second is the variance or width of the band and the higher moments describe the skewness and flatness of the distribution. For systems of only one atom type, the mean energy of the band can be set to zero, negating the first moment. The second moment is the dominant term in determining physical properties and can be calculated analytically within the tight-binding model whilst higher moments must be determined numerically. Consequently, in modelling the interatomic potentials of transition metals it is common to only consider the second moment of the density of states of the partially filled d-band, this is the second moment approximation. If only the nearest neighbours are considered, the second moment of the density of states can be determined by calculating the sum of hopping integrals between d-band states which emerge as matrix elements in the Hamiltonian of the two-atom system [95]. The integrals are a function only of the distance between the two atoms and lead to the many-body binding energy in equation 5.3.

$$V_B^i = \left[\sum_{j \neq i}^{N} \zeta^2 exp\left(-2q_{ij} \frac{r_{ij} - r_0}{r_0} \right) \right]^{1/2} \tag{5.3}$$

The repulsive potential separating two nuclei can be described by a sum of pairwise Born-Mayer energies given by equation 5.4.

A	0.2061 eV
ζ	1.790 eV
p	10.229
q	4.036
r_0	2.884

Table 5.1: Table of Gupta potential parameters for gold taken from Cleri [94].

$$V_R^i = \frac{1}{2}\sum_{j\neq i}^{N} Aexp\left(-p\frac{r_{ij}-r_0}{r_0}\right) \tag{5.4}$$

The Gupta potential is a combination of these attractive and repulsive terms, equation 5.5.

$$V = \sum_{i=0}^{N}(V_R - V_B) \tag{5.5}$$

This potential is used in all the simulations in this work. It is a semi-empirical potential, the parameters of which are determined from experimental measurements of lattice constants, cohesive energies and elastic constants in bulk specimen. The parameters used in this work are those given by Cleri and Rosato [94] and shown in table 5.1.

The molecular dynamics simulations used in this work are not novel and so do not warrant extended discussion. Other empirical potentials exist which should be satisfactory for the needs of this work but which offer no particular advantages so their relative merits will not be discussed here. The accuracy with which molecular dynamics simulations reproduce thermal motion is discussed in chapter 6.

To ease computation, simulations often only calculate the potential due to atoms within a certain radius of each other. In this work the clusters are small so the potential cutoff is defined to be larger than the diameter of the cluster so that all atoms are included in interaction calculations.

5.4 Dl_poly

The molecular dynamics simulations in this work were performed using the Dl_poly package developed by Smith, Forester and Todorov [96]. Before performing simulations to acquire Mean Square Displacement (MSD) values, the accuracy of the software was investigated through comparing energy minimisation simulations with the results produced by a program independently developed by Prof. Riccardo Ferrando. The first check involved calculating the initial energy of a gold trimer using a Gupta potential (see section 5.3). The Dl_poly package determined the energy to be 4.6784 eV, whilst Ferrando's program and analytical calculations gave a value of 4.7328 eV. The standard Dl_poly simulation uses interpolations of tabulated potentials rather than analytical calculations to reduce computation times but the package is also able to perform full analytical calculations for small structures. Using the analytic calculations the relaxed trimer energy agreed to 8 significant figures (the maximum precision of the package) with the program of Ferrando. The source code of Dl_poly was iteratively edited to increase the resolution of the potential interpolation tables. The initial energy of the trimer was calculated at each iteration and was found to converge on a value agreeing with the analytical value to 6 significant figures. With the more precise interpolation table, conjugate gradient energy minimisations were performed for three pure gold structures: a 309 atom ino-decahedron and two FCC crystalline clusters of 586 and 1289 atoms. In each case the minimised energy was in good agreement with that calculated using Ferrando's program.

All the molecular dynamics simulations in this work are performed using the velocity verlet algorithm and a timestep of 3 fs. Unless otherwise stated, the simulations each consist of an equilibrium period of 0.3 ps before a production run of 0.3 ps during which data is gathered. The root mean squared energy fluctuation during the production run of a simulation typical of those used in chapters 6 and 7 featuring a 284-atom gold nanocluster at ~ 300 K was 4.9×10^{-4} eV, indicating that the timestep is sufficiently small to produce a stable simulation. In the case of periodic simulations, the Gupta potential is used with a cutoff distance of 10 Å, this approximately corresponds to the 6th nearest neighbour distance and prevents self interactions. Simulations of clusters are performed without periodic boundaries and a potential cutoff of 30 Å is used as this range includes all the atoms in the nanocluster systems used in chapters 6 and 7.

The Dl_poly program imposes that there be no net linear or angular momentum of the

cluster centre of mass by default, which is fortuitous as it simplifies calculating the mean square displacement values of individual atoms.

5.5 Production of nanocluster structure models

A program has been written to generate the nanocluster structures featured in this work and can be found in appendix 3. The program is written in Fortran 90 format and can be used to produce icosahral, decahedral, ino-decahedral and Marks decahedral structures. The program requires that the user inputs the desired number of atomic shells and the bond length. For truncated clusters, the degree of truncation must also be defined. The program produces an output file which consists of three columns containing the x, y and z coordinates of each atom. The program starts by constructing an array containing the x, y and z coordinates for each atom in an FCC tetrahedral of the required size, for truncated clusters, the tetrahedra is deformed by removing atoms at this stage. The coordinate array is then copied a number of times, with each copy being multiplied by a rotation matrix to produce the additional tetrahedral segments of the twinned cluster. Finally, a nested do loop is used to scan through the coordinates and remove any doubly occupied sites resulting from the overlap of adjacent tetrahedral segments. The finished cluster is aligned with its $\langle 110 \rangle$ axis along the z-axis with the central atom at the origin. The clusters produced using this method are perfectly regular with straight atomic columns and a uniformly distributed missing wedge.

Chapter 6

Atom Counting

6.1 Introduction

As discussed in chapter 4, determining the three dimensional structure of nanoclusters with atomic precision is vital to clarifying the causes of their exotic behaviour. In this chapter, a new technique combining molecular dynamics and multislice calculations to produce more rigorous HAADF-STEM simulations is described. This is used to assess the effects of sample inhomogeneities such as strain and enhanced thermal motion on image intensities. Since the true structure of the model specimen is known, the accuracy of quantitative analysis methods can be assessed. In particular, the statistical parameter estimation method developed by Van Aert *et al.* [37], and described in section 2.3.8, will be tested in the case of catalytically interesting small multiply twinned metal clusters.

This chapter begins with a discussion of the details of the new methodology. The molecular dynamics simulations are described, detailing the input parameters which are used. An appraisal of the accuracy of the simulations is provided followed by a discussion of the thermal motion and strains which they predict. The convergence testing used to determine appropriate parameters in the multislice simulations are then described and the accuracy of the simulations is confirmed through comparisons with other well-established multislice simulations, as calibrated experimental images suitable for this purpose are not available. The quantitative analysis method will be outlined, followed by the results of the rigorous image simulations and a discussion of their implications for structural characterization.

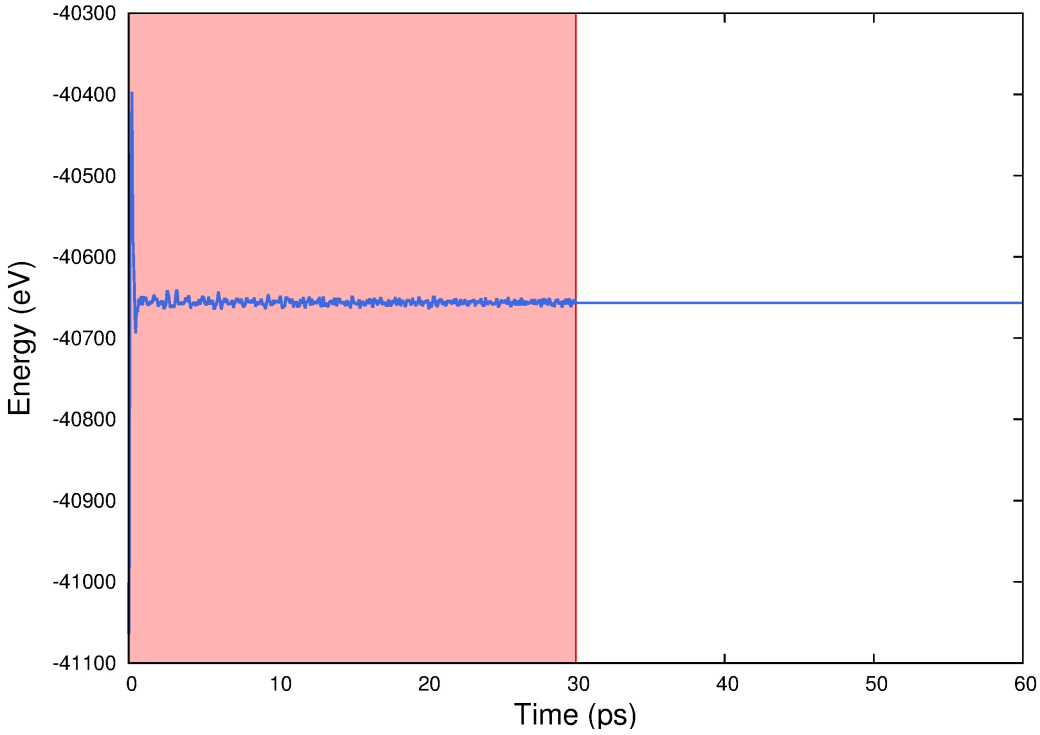

Figure 6.1: Internal energy throughout the MD simulation of a periodic gold structure. The equilibration period is highlighted in red.

6.2 Molecular Dynamics simulations

The accuracy with which the molecular dynamics program used here calculated energies has been confirmed by comparison with results generated by Prof. Riccardo Ferrando, as described in section 5. The accuracy with which thermal vibrations are simulated is ascertained by comparisons with experimental bulk values and by comparison with the calorific curves simulated for similar particles in the literature.

For comparison with bulk values, a simulation of a periodic FCC structure was performed at 295 K. The Gupta potential parameters calculated by Cleri *et al.* were used. These are shown in table 5.1 [94].

The simulation was performed in the microcanonical (NVE) ensemble with timesteps of 3 fs. The simulation begins with a 0.3 ps equilibration period before commencing a 0.3 ps production run. The mean square displacements (MSDs) are calculated by determining the mean position of every atom in the sample over the course of the production run, and then taking the time average of the displacement from this position. The evolution of the internal energy and temperature of the system are shown in figures 6.1 and 6.2 respectively.

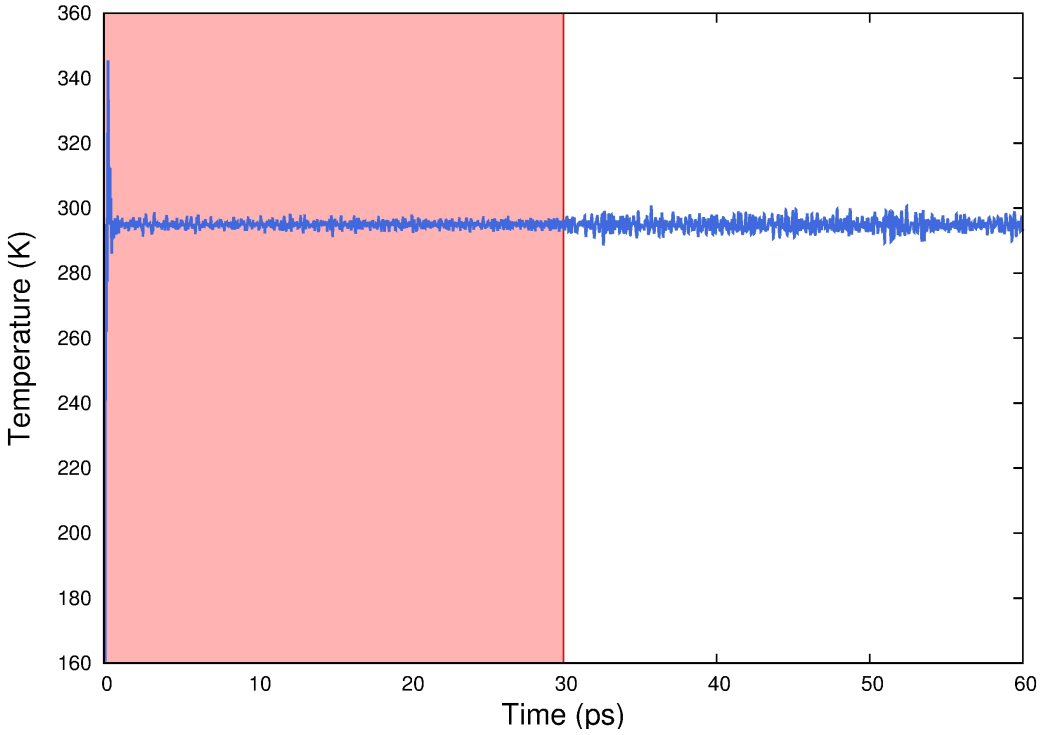

Figure 6.2: Temperature fluctuations throughout the MD simulation, average temperature during the production run was 294 ± 2 K

It can be seen that the system is well equilibrated with only inherent statistical fluctuations in the temperature. The temperature was not fixed during the production run as this would lead to non-physical velocity scaling as described in section 5.2. It was thus approximately controlled by restricting the internal energy of the system. This gave a slight deviation from the target temperature. The average temperature over the course of the production run was 294 ± 2 K.

Averaging over the 10,976 atoms gives a MSD of 0.0093 ± 0.0006 Å^2. By comparison, Peng *et al.* derived a Debye-Waller factor of 0.62 ± 0.02 at 295 K from neutron scattering experiments, this corresponds to a MSD of value of 0.0079 ± 0.0002 Å^2. This establishes that the simulation can reproduce bulk values with reasonable accuracy.

To assess the accuracy with which the thermal properties of a nanocluster can be simulated, a caloric curve has been produced for comparison with similar simulations. A 284 atom cluster is used as this is in the catalytically active size region. The cluster has a Marks decahedral structure as this morphology has been found to be the optimum energy configuration for this size using the Gupta potential [64]. A ball model of the structure is shown in figure 6.3.

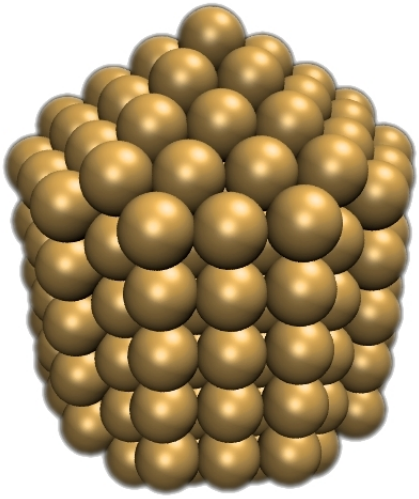

Figure 6.3: a) Structure of a 284 atom Marks-decahedral cluster used extensively in this work.

To produce a calorific curve, a geometrically regular structure was first relaxed using the conjugate gradient method before being used in a series of simulations at increasing temperatures from 250 K to 750 K. An equilibration period was included at each temperature step and the final structure from one simulation was used as the initial structure for the following simulation. The resulting calorific curve is shown in figure 6.4.

The approximate melting point from the simulations is plotted alongside results compiled from the literature in figure 6.5. The melting point here is in good agreement with those found in the literature at this size, suggesting that the simulation models the thermal effects at the nanoscale with reasonable accuracy.

Whilst the cluster undergoes global melting at $\sim$ 650 K, surface melting occurs at $\sim$ 600 K. This can be seen by inspecting the mean square displacements of the surface and core atoms as seen in figure 6.6.

Once an atom has gained enough energy to allow net mobility rather than just oscillations about a mean site, less can be inferred from the MSD values, since the atom no longer has a fixed mean position from which to measure displacements. This surface premelting observed here is widely reported in the literature [80, 74, 73, 75, 78], as is the greater stability of the (111) surfaces compared with the (100) surfaces [97, 78].

With the accuracy of the simulations established, the variation of the MSD at different positions in the cluster can be examined. In the remaining parts of this chapter all

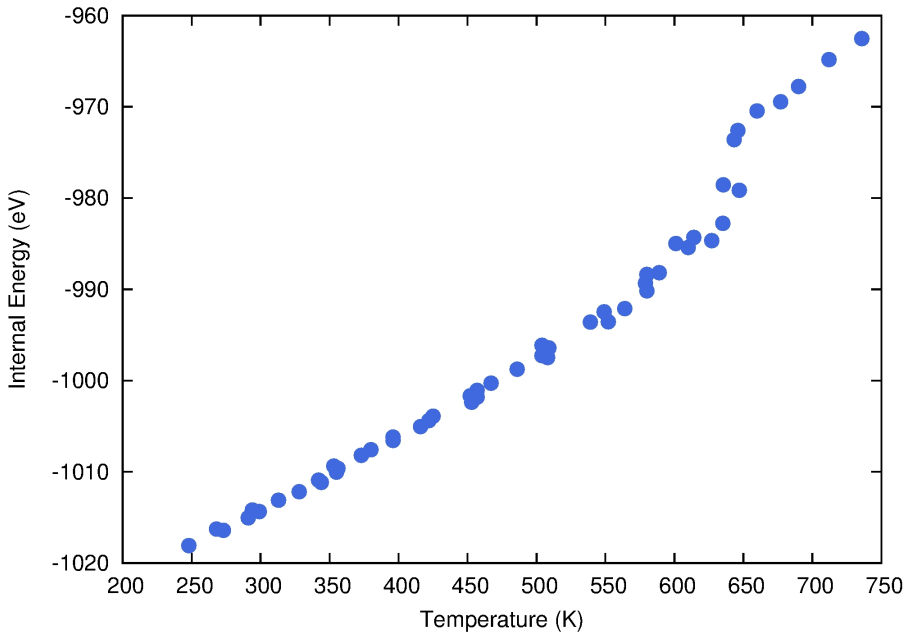

Figure 6.4: Calorific curve for the cluster in figure 6.3. A phase transition can clearly be seen at $\sim$ 650 K.

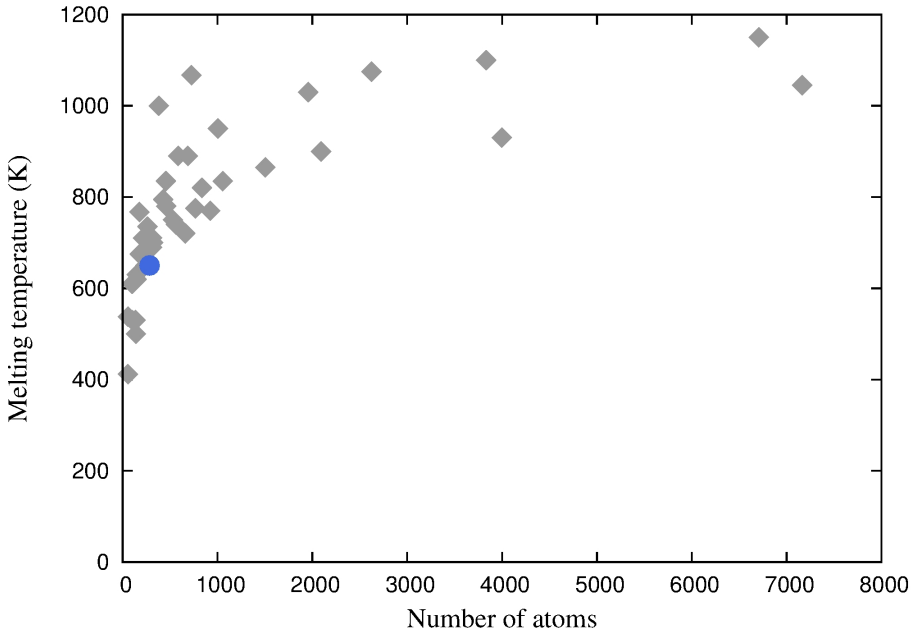

Figure 6.5: Simulated melting points. The result from the simulation in this work is plotted as a blue circle, the grey diamonds are results compiled from the literature [73, 74, 75, 76, 77, 77, 79]. The results from the literature are split into their sources in figure 4.7.

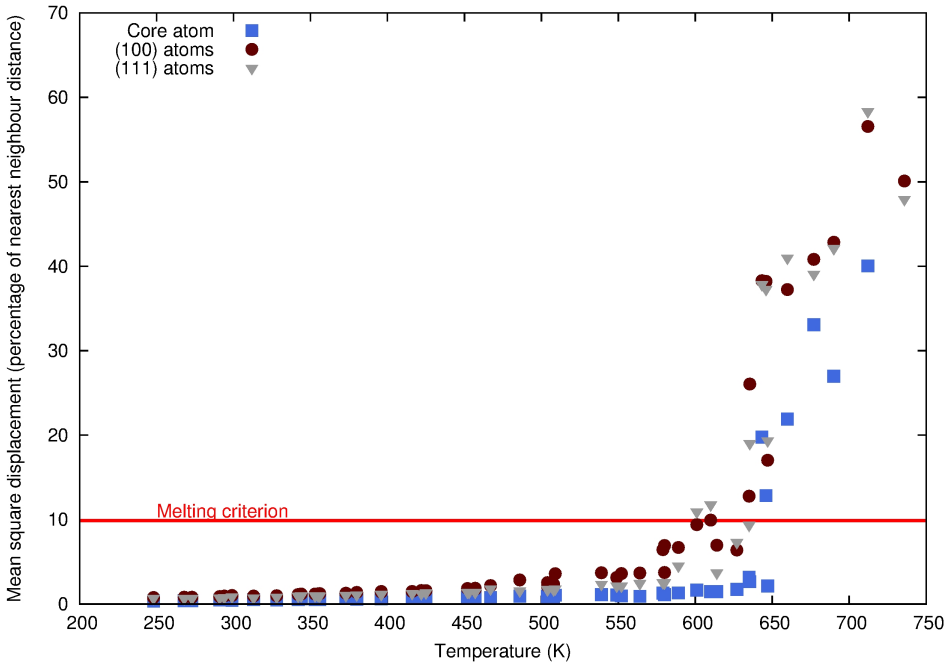

Figure 6.6: Variation of mean square displacements with temperature for a 284 atom gold cluster. The surface undergoes premelting at $\sim$ 600 K. The Melting criterion is defined by the MSD reaching 10% of the nearest neighbour distance.

discussion of molecular dynamics will refer to the simulation performed at 300 K, unless otherwise specified. This temperature is chosen as offering a reasonable value for most microscopy environments. Cryogenic and elevated temperatures are considered in chapter 7. The MSDs of atoms at 300 K are illustrated in figure 6.7, which shows a schematic of the cluster with a sphere at each atom position with a radius given by the MSD of the atom at that location, multiplied by a factor of 40 to enhance visibility.

The surface atoms show enhanced thermal motion, particularly at the vertex positions which have an average MSD of 0.025 ± 0.002 Å^2. The MSDs of the edge atoms are the next largest at 0.022 ± 0.002 Å^2 followed by the (100) surfaces at 0.021 ± 0.003 Å^2 and the (111) surfaces at 0.018 ± 0.002 Å^2. The atoms at the top and bottom tips of the cluster as shown in figure 6.7 exhibit similar MSDs to those in the rest of the (111) surface at 0.019 ± 0.001 Å^2 despite their reduced coordination. This is attributed to surface relaxation in which the tip atoms sink into the surrounding 5 atoms. At greater temperatures the tip atoms sink underneath the surrounding 5 atoms forming a rosette feature that has been reported elsewhere in gold and platinum clusters [98, 99, 100]. The MSDs of the interior atoms show relatively little variation, the core atom has a MSD of 0.008 ± 0.001 Å^2 whilst the first sub-layer under the surface has an average MSD of 0.012 ± 0.001 Å^2.

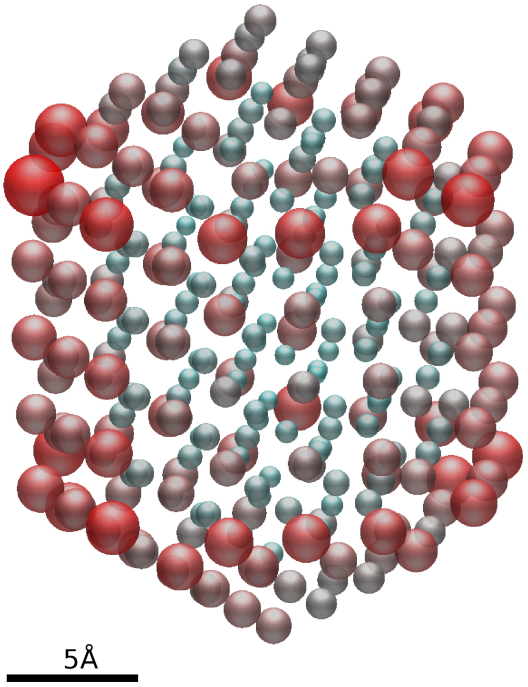

Figure 6.7: Schematic illustrating the variation in the MSD at differing locations within the cluster at 300 K. The sphere radii are given my the MSD component multiplied by a factor of 40 for visibility.

The inhomogeneous structure of the cluster results in somewhat anisotropic thermal motion, however this is negligible at the temperatures typically used in electron microscopy experiments. The new methodology in which the multislice image simulations use frozen phonon configurations extracted from molecular dynamics simulations will include any anisotropy by default, which could be of significance for other model systems.

Relaxing the cluster introduces static disorder to the structure due to the balancing of internal and surface strains. This results in a pseudo spherical curving of the atomic columns as illustrated in figure 6.8, which shows the unique columns of the cluster at 300 K. The columns are labelled by their atom count followed by 'B' or 'I' indicating whether they lie on the border or in the interior of a tetragonal segment, respectively. The columns are plotted with the (110) crystallographic direction on the ordinate axis whilst the abscissa is the radial distance from the (110) axis, so that the full extent of the curvatures of the different types of column can be illustrated on the same plot.

The columns along the borders exhibit greater curvatures than those in the interior and the shorter columns show greater curvatures. To quantify the static displacement, the time-averaged position of each atom in the cluster has been calculated from the 300 K

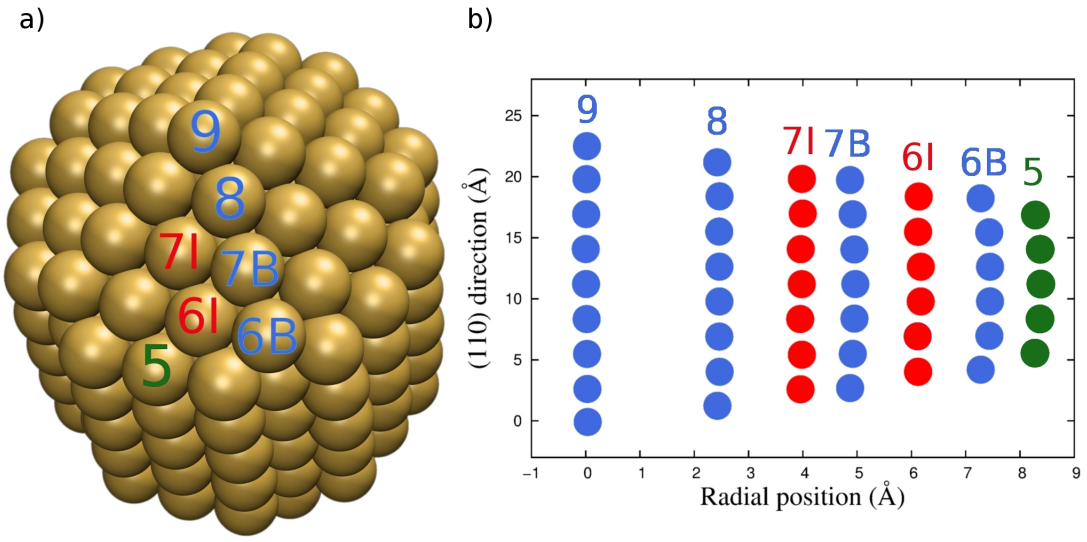

Figure 6.8: Strain causes curvature of atomic columns with displacements transverse to the (110) direction.

simulation. The displacements of each atom from the mean transverse position of its column have been calculated to give a measure of the magnitude of the static displacements. These values are given for each of the seven unique column types in table 6.1.

Column type	Mean static displacement (Å^2)
5	0.037 ± 0.005
6B	0.054 ± 0.006
6I	0.017 ± 0.001
7B	0.024 ± 0.003
7I	0.009 ± 0.001
8	0.012 ± 0.002
9	0.0014 ± 0.0002

Table 6.1: Strain causes static displacements from perfect alignment along the beam axis.

The static displacements are of the same order as the thermal vibrations of the surface atoms at 300 K, with the exception of the central column which is almost perfectly aligned parallel to the beam axis. To determine what effect the static and thermal displacements will have on HAADF-STEM images, a number of multislice simulations have been conducted and are described in the following section.

6.3 Multislice HAADF-STEM simulations

In this section the details of the multislice simulations used throughout this work are discussed. The convergence tests needed to determine sampling rates are provided to explain the input parameters for the simulations. The methodology and results of several simulations using differing structural models and frozen phonon algorithms are provided which allow the effects of thermal atomic displacements and static strain-induced displacements to be isolated. The implications of the results for atom counting procedures are then discussed, including a test application of the statistical parameter estimation procedure described in section 2.3.8.

In order to assess the results of multislice simulations, it is necessary to apply a quantitative image analysis technique. This section begins with a description of the analysis method used in this work so that the subsequent discussions of convergence tests can be understood.

6.3.1 Quantitative analysis

In order to quantitatively compare simulated images, an analysis program has been written. The program reads both tiff and DM3 image files, as well as double precision data files formatted specifically for this work. The purpose of the program is to identify and quantify intensity spots in zone axis images such as that in figure 6.9. The images are quantified in terms of their absolute intensities. Both peak intensities and integrated intensities are calculated, as they offer complimentary insights into the image formation process. A cross-correlation process is used to identify peaks in the image, the user provides an approximate spot diameter for the intensity peaks, which is used to produce a Gaussian function array for normalised cross-correlation with the image. This yields an array of equal size to the HAADF image with a value at each pixel quantifying the quality of the correlation at that point. The pixels producing cross-correlation values above a given threshold are taken to be the approximate positions of peaks in the image. Image integrations are then performed over the area of a small disk, which is rastered around the approximate peak position. The centre of the disk with the largest integrated intensity is then taken to be the centre of the peak. A centre of mass calculation may offer a more accurate result but is more computationally expensive and the accuracy is of little benefit here. With the centres of the peaks identified, the image is split into Voronoi cells about

each peak and the intensity of each peak is given by the integrated intensity within the cell. Figure 6.9 shows an example simulated image of a Marks decahedron along the (110) direction with peaks identified and Voronoi cells shown by blue lines. The numbers show the integrated intensity (blue) and atom counts (green).

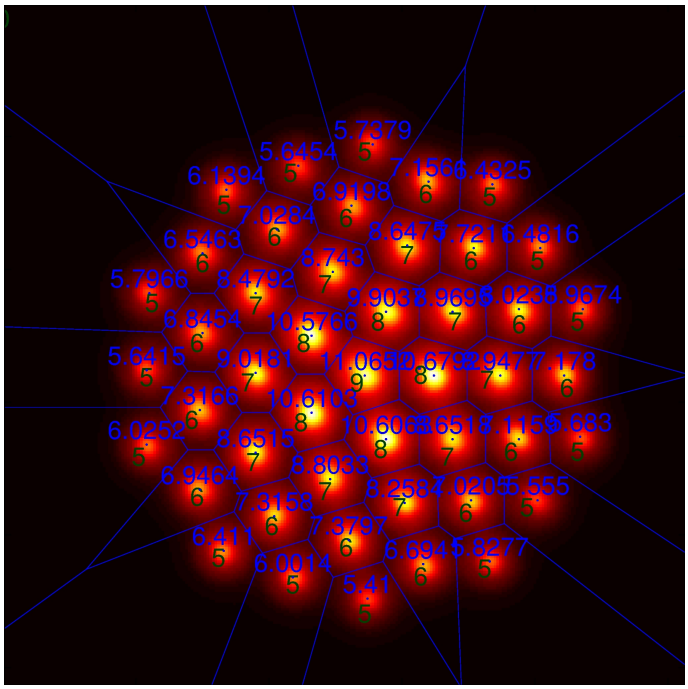

Figure 6.9: HAADF-STEM image of a Marks decahedron along the (110) zone axis. The blue lines show Voronoi integration cells used to quantify the intensity of each peak. The blue text gives the integrated intensity as a fraction of the incident beams and the green number indicates the number of atoms in the column responsible for each intensity peak.

In cases where there is a large overlap in column intensities it may be preferable to use the intensity of the rastered disk, which should be less susceptible to contributions from neighbouring columns, however, in this work the columns are well defined with little overlap. The Voronoi integration technique has the advantage that it gives a good approximation to measuring the high-angle scattering cross-section and, in comparison with single peak or small disk intensities, it is more robust to effects which act to smear the peak intensity over a larger area, but do not reduce scattering [101]. For simulated images for which the specimen structure is known, the model input file can be read by the image analysis program in order to relate the image intensity spots to the corresponding atomic columns allowing the atom count or MSDs to be attributed to each image peak.

The quantitative analysis program for this work is a script which must be executed in Matlab. The script is included in appendix 4, and included in the electronic copy as the matlab script Peakfinder.m.

6.3.2 Convergence tests

To ensure that the multislice simulations are functioning correctly, a number of convergence tests have been performed to check that sufficiently high sampling rates are used. There are several parameters which must be checked, the number of sampling points in the final image must be sufficiently large that atomic columns are well resolved, the scattering potential slices must be sampled at a suitable rate in both real and reciprocal space, and the array describing the electron beam wavefunction must also be well sampled in real and reciprocal space. In addition, the frozen phonon algorithm must be used with enough configurations so that image intensities converge within reasonable bounds. Since the aim of this work is to investigate column integrated intensities, these are used as the metric by which convergences are determined.

The real space size and the sampling rate of the transmission function arrays define the maximum image frequencies and real space resolution in accordance with equations 6.1, 6.2 and 6.3.

$$\Delta x = \frac{a}{N_x} \tag{6.1}$$

$$\Delta k_x = \frac{1}{a} \tag{6.2}$$

$$|k_{xmax}| = \frac{1}{2\Delta x} \tag{6.3}$$

Since the realspace size of the transmission function, a, is fixed to accommodate the sample, the sampling size, N_x must be made large enough that k_{xmax} is greater than the angular range of scattered electrons in the wavefunction, otherwise electrons scattered beyond k_{xmax} will be lost. To determine the minimum sampling rate needed, a series of simulations for a small a=10 Å supercell have been performed. The integrated wavefunction intensities and image-integrated HAADF signal have been recorded for each image and are plotted in figures 6.10 and 6.11.

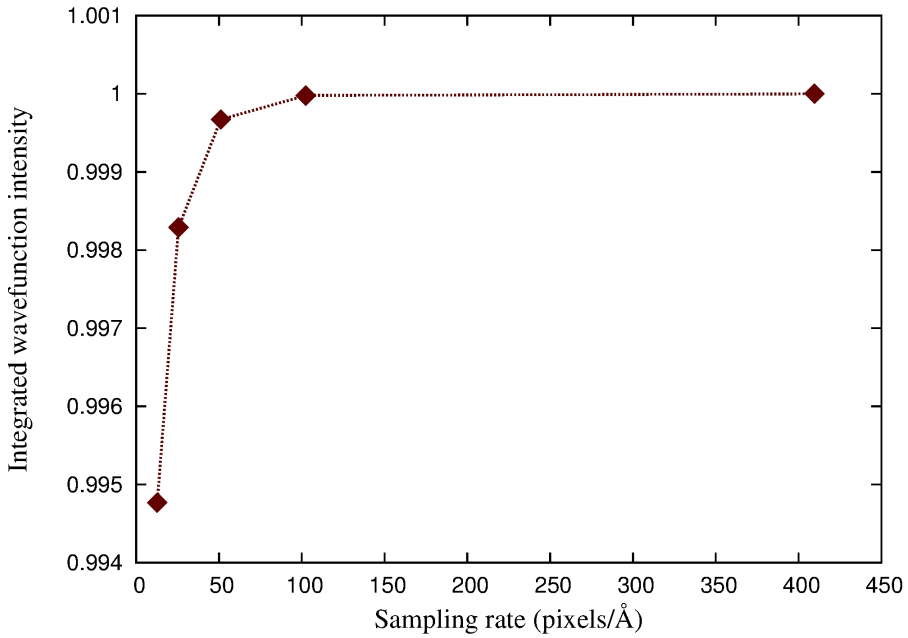

Figure 6.10: Integrated intensities of the exit wavefunction at varying transmission function sampling rates.

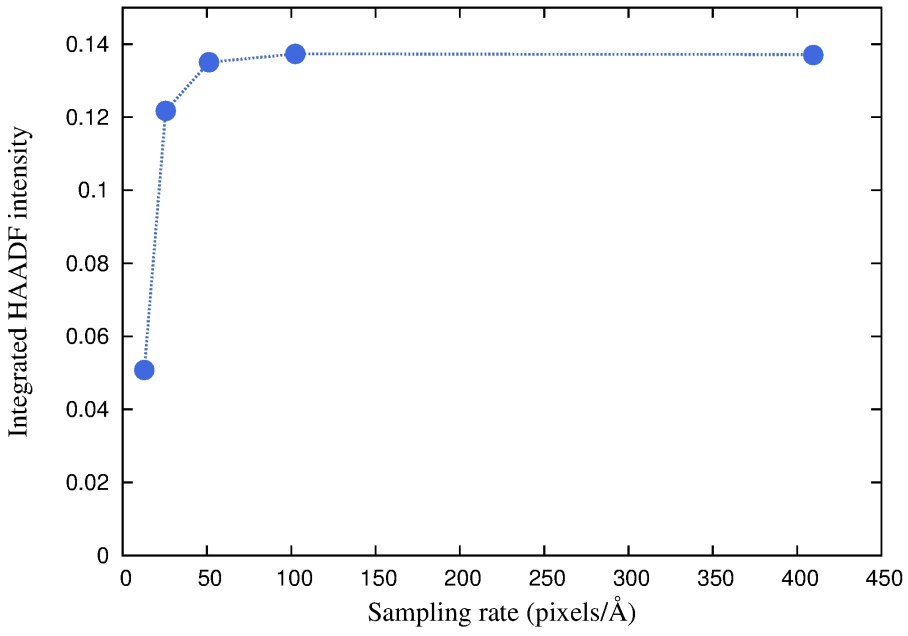

Figure 6.11: a) HAADF integrated intensities at varying transmission function sampling rates.

It can be seen that for sampling sizes below ~ 100 pixels per Å, the transmission function is undersampled and significant portions of the wavefunction are lost to higher angles than the sampling allows. For sample sizes above 100 pixels per Å, the entire wavefunction remains within the array and the intensity at the HAADF detector has converged. The efficiency of DFFTs is better for array sizes which are an integer powers of 2, so for the production simulations in this work, a transmission function sampling size of 4096^2 is applied to a real space supercell of 77×77 Å, to give a sampling rate of 53 pixels per Å. This should be just below the converged regime for the sampling test described above, however, this does not necessarily mean that this sampling rate is not high enough for this work, as the maximum scattering angle occurring in a simulation will vary depending on the specimen. To check that this sample rate is high enough for a typical cluster simulation, a one-off simulation using the larger sampling size of 6144^2 has been conducted. Figures 6.12 a) and b) show the resulting images using array sizes of 6144^2 and 4096^2, respectively. Figure 6.12 c) shows the difference between the images and figure 6.12 d) shows the integrated column intensities from both images which exhibit an excellent agreement.

As the two simulations show a good agreement, the transmission function sampling size of 4096^2 will be used throughout this chapter. For all subsequent production simulations the intensity of the wavefunctions at the exit surface has been monitored to ensure that only negligible intensity falls outside of the maximum frequency range, the loss in all cases is 0.0004% or less.

As the focused wavefunction is narrow, it need not be described by as broad a real space array as the transmission function. Figure 6.13 show the variation of incident intensity with the wavefunction array size. It can be seen that the intensity of the wavefunction has converged well by ~ 2048 pixels and changes by only 0.1% in doubling the array size to 4096 pixels. Consequently, the probe will be sampled by a 2048^2 array for all simulations in this chapter.

Having ensured that the dimensions of the wavefunction and transmission function arrays produce accurate calculations, the number of incident probe positions must be chosen. This must be sufficiently high to resolve individual column intensities with enough detail to characterise them. The nanocluster in this work has a diameter of approximately 15 Å so, allowing for a 5 Å border around the cluster, a 25 Å real space image size is used. Figure 6.14 shows three images generated under identical conditions, with the exception of the number of incident beam positions. Throughout this work the images are sampled

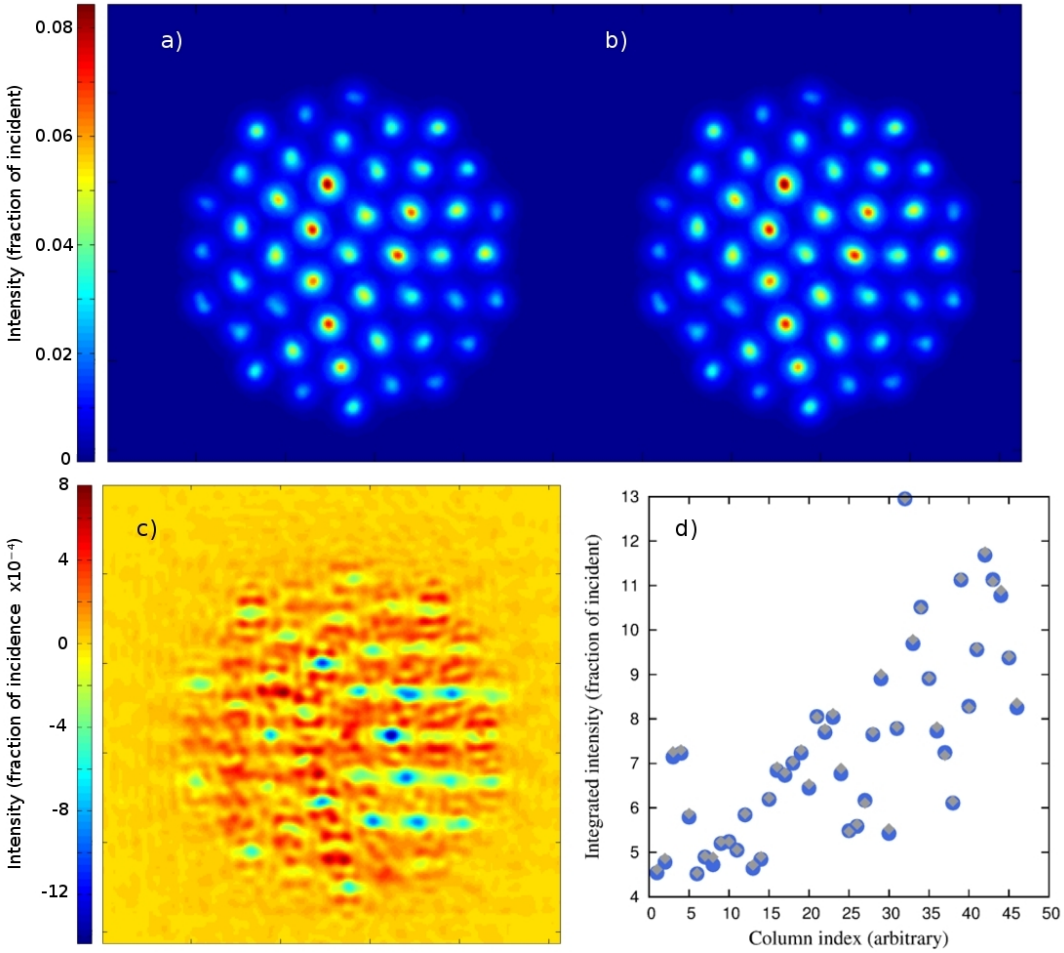

Figure 6.12: Typical image simulations for this chapter using transmission function array sizes of a) 6144^2 and b) 4096^2. c) shows the difference subtracting b) from a), note that the intensities are $\times 10^{-4}$. d) Comparison of individual integrated column intensities for the 6144^2 (grey) and 4096^2 (blue) array sizes.

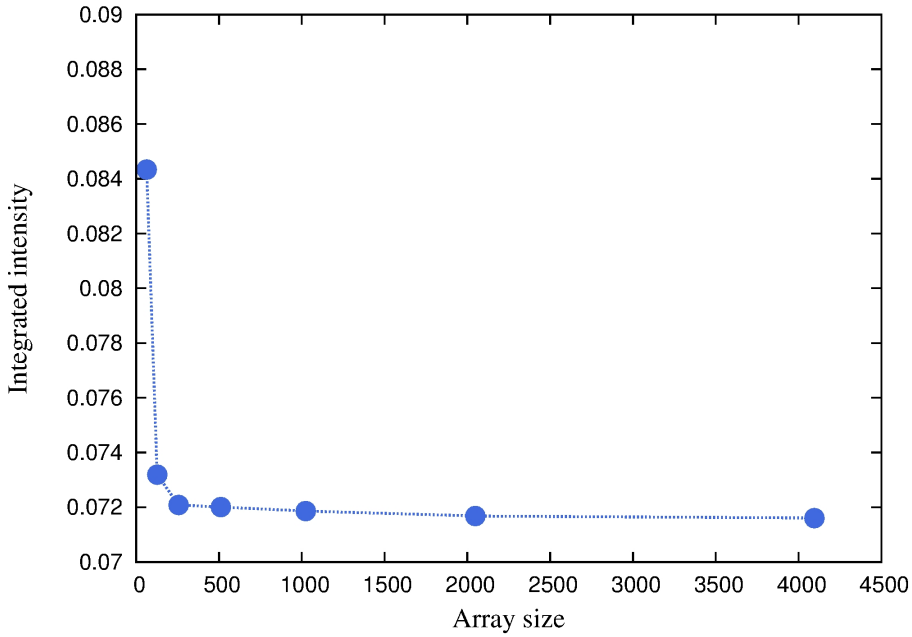

Figure 6.13: Image integrated intensities for vary probe array sizes, the real space pixel size is the same as that of the transmission function array.

with 256^2 incident beam positions produces an image with 10.24 pixels per Å, so there are approximately 30 pixels separating nearest neighbours.

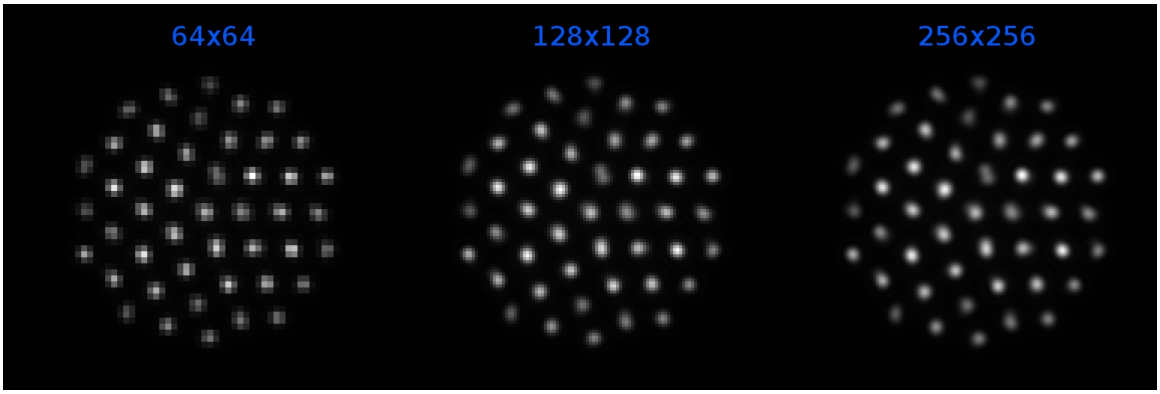

Figure 6.14: Nanocluster images generated under identical conditions but with differing scanning resolutions. Throughout this work 256×256 pixel images are used for quantitative analyses.

As described in section 3.2.3, the frozen phonon algorithm is used to reproduce the effects of thermal motion on image formation. This entails taking the average of a series of images with perturbed atomic positions. It is thus necessary to ensure that the number of images is great enough that the intensities have converged. A series of images for frozen phonon configurations taken from a molecular dynamic simulation of a 284 atom nanocluster at 300 K have been generated to determine how many configurations are needed

for the intensities to converge. Figure 6.15 shows the convergence of the configuration averaged intensities for 10 arbitrarily selected columns.

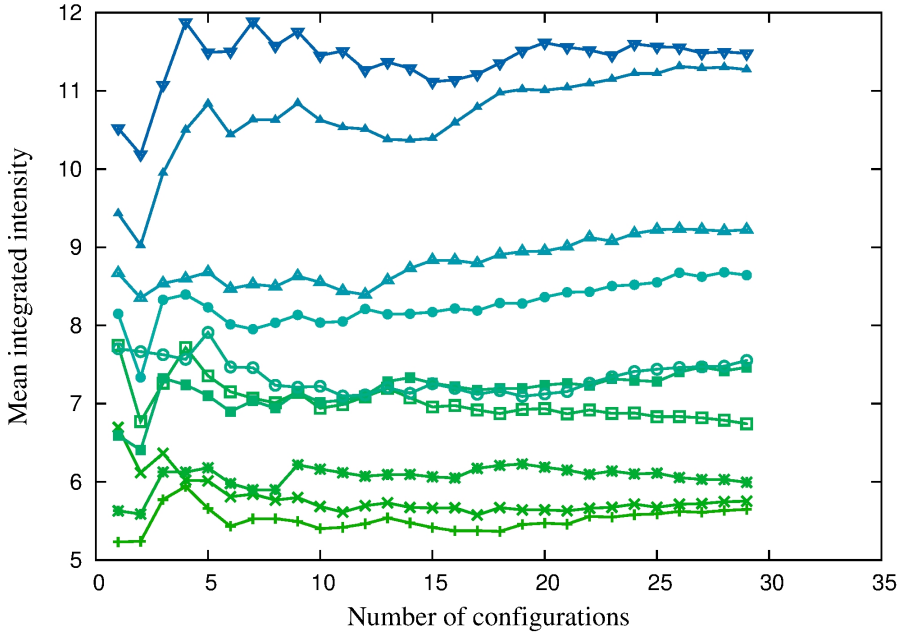

Figure 6.15: Convergence of a frozen phonon series, each line shows the mean intensity of a peak in the HAADF image. The thermal fluctuations of the sample preclude the intensities converging to single values.

Unlike the array convergences described previously, the intensities here do not converge to single values, rather, they oscillate about a mean point due to the pseudo-random method of conventional frozen phonon algorithms or, in this case, the inherent variation of the molecular dynamics configurations. The percentage difference between consecutive mean intensities for the same 10 peaks are plotted in figure 6.16, the difference converges to less than 2% for all columns after 20 configurations.

The mean percentage difference between consecutive mean intensities is $0.2 \pm 0.7\%$ after 20 configurations and $0.3 \pm 0.6\%$ after 30 configurations which suggests there is negligible benefit to using more than 20 configurations.

The multislice algorithm is best suited to structures which are periodic along the beam axis, due to the quantization of z-coordinates to the nearest slice. In this work the sample has a nominal lattice constant of 2.88 Å, as the sample has an FCC crystallography, the separation of atoms in the (110) direction is ~ 1.44 Å. The structures in this work are not strictly periodic. Gold has been reported to exhibit contracted bond lengths at surfaces and this is reproduced in the molecular dynamics simulations used here [102]. The enforced

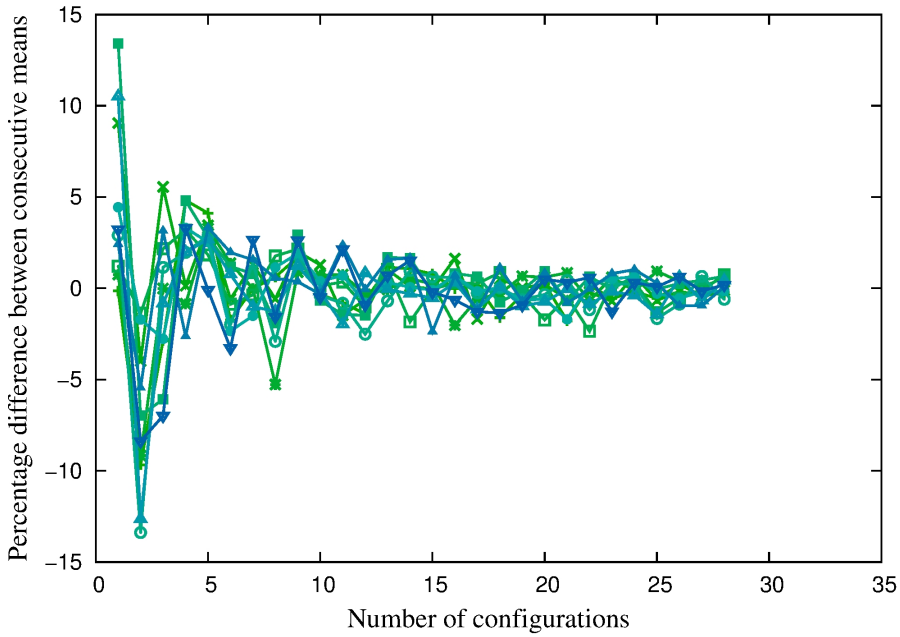

Figure 6.16: The difference between consecutive mean values in the frozen phonon configuration converges to less than 2% for all peaks after 20 configurations.

periodicity of standard multislice simulations thus introduces errors. The mean spacing in a relaxed structure at 300K has been found to be 1.424 ± 0.002 Å. The simulations in this work all use 1.44 Å slices which should retain the FCC structure and produce accurate images. The errors that are incurred will be minimised by the use of a broad detector range.

Since the image intensities are the focus of this work, they are recorded as a fraction of the incident beam intensity in double precision, to minimize quantization errors. This precision is good enough for the investigations in this work and considerably higher than that used in experimental electron microscopy.

6.4 The effects of inhomogeneities on multislice simulations of nanoclusters

To determine the effects of the inhomogeneous strain and thermal motion on HAADF-STEM images, multislice calculations have been performed. To isolate the contributions of the static and thermal disordering, three simulations have been performed. In the first, a regular Marks decahedral cluster with no structural optimisation is used with homogeneous bulk MSD values taken from the literature [72]. This model thus exhibits none of

the nanoscale effects other than the twinned morphology. Simulations using these approximations have been reported in the literature [41]. In the second simulation, the cluster structure is relaxed, but still uses homogeneous bulk MSD values, so that the effects of the static displacements can be determined. Models such as this have also been used in published reports [35]. In the third simulation, cluster configurations are extracted from a molecular dynamics simulation at 300 K and used directly in the frozen phonon algorithm of the multislice simulation. This simulation is more rigorous than any reported in the literature as it includes static displacements and inhomogeneous, anisotropic thermal motion. Throughout this section, these simulations will be referred to as simulations I, II and III, respectively, to maintain brevity.

All of the simulations are performed under typical HAADF-STEM settings, though a perfect lens system was assumed, with no aberrations. The input parameters are listed in table 6.2.

Accelerating voltage	200 keV
Defocus	-50 to +50Å
Convergence angle	20 mrad
Detector range	90 230 mrad
Slice thickness	1.4 Å
Frozen phonon configurations	20
Transmission function size	4096^2 pixels / 77 Å
Probe function size	2048^2 pixels / 35.5 Å

Table 6.2: Multislice simulation parameters

Lens instabilities are accounted for by taking images at -50, -25, 0 , 25 and 50 Å defocii, and combining them with a Gaussian weighting.

The results for simulation I, II and III are shown in figures 6.17a), b) and c), respectively.

Simulation II produced the image with the greatest integrated intensity at 86.1×10^{-4}, as a fraction of the total incident intensity. Simulation I produced a similar, but lower integrated intensity, of 81.51×10^{-4}, whilst simulation III yielded a significantly reduced intensity of 60.8×10^{-4}. This suggests that the introduction of enhanced thermal motion in simulation III results in a reduction in integrated intensities. This is consistent with observations from multislice simulations of gold [7] but contradicts theoretical predictions [47]. This contradiction is addressed further in chapter 7.

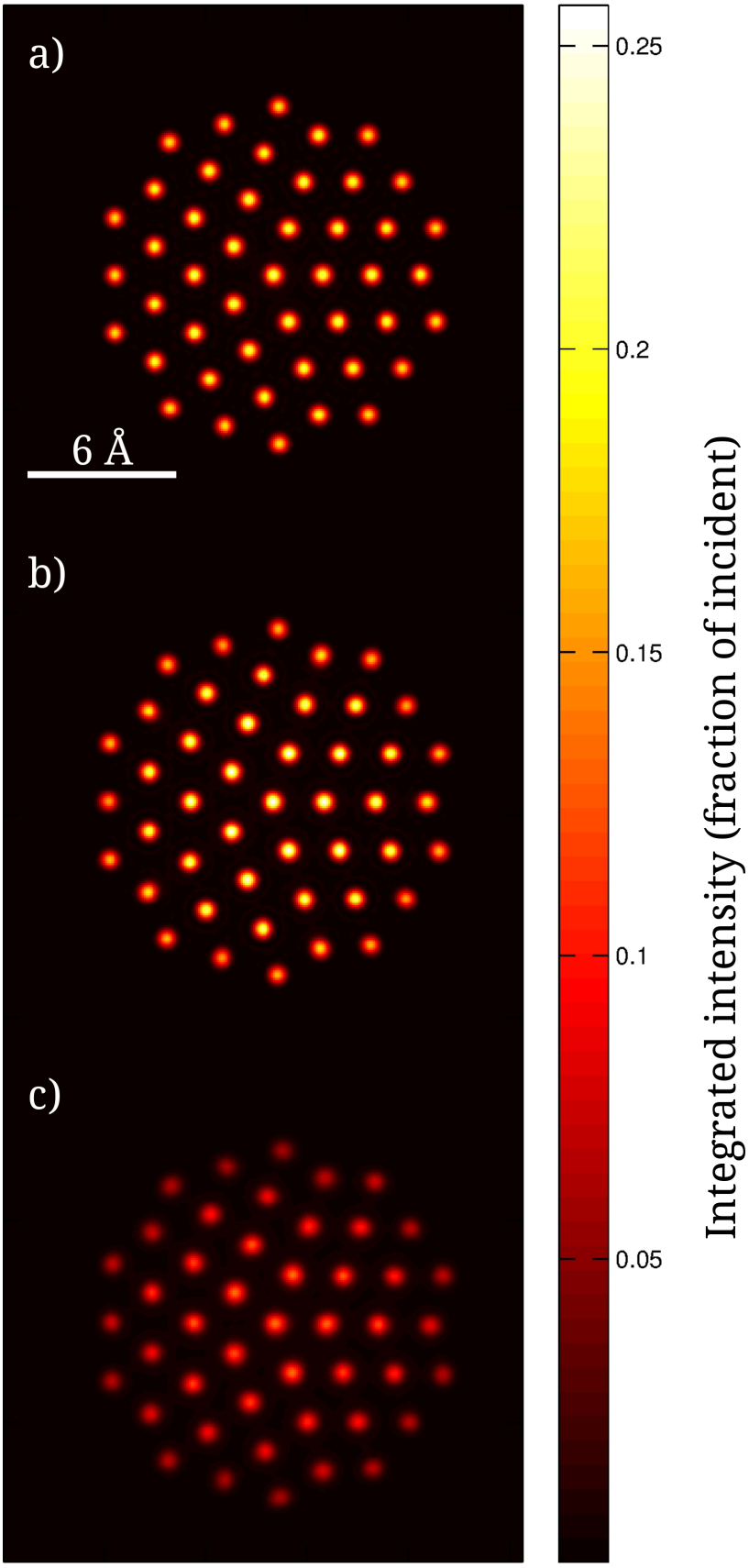

Figure 6.17: HAADF-STEM image of a Marks decahedron along the (110) zone axis. a) Simulation I, regular structure and bulk MSDs. b) Simulation II, Relaxed structure and bulk MSDs. c) Simulation III, Molecular dynamics configurations with relaxed structure and realistic thermal motion.

The quantitative analysis method described in section 6.3.1 has been applied to the three images. Since the relationship between atom count and peak intensity is the primary interest in this investigation, the integrated intensities are plotted against the atom count of the corresponding columns. Figure 6.18 shows the results for simulation I, the regular structure with bulk MSDs.

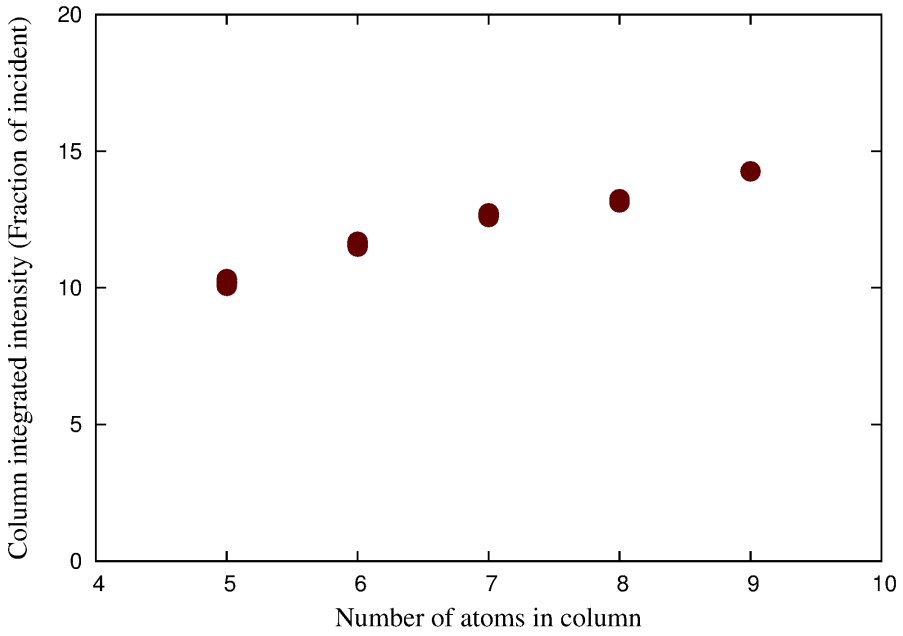

Figure 6.18: Relationship between atom count and peak intensity from simulation I using a regular structure with bulk MSDs.

The results show spreads of intensity for each atom count, rather than a single value, due to the quasi-random nature of the frozen phonon model. The integrated intensities show a monotonic increase with atom count, which is the fundamental requirement for atom counting procedures. The relationship is almost linear and there is a distinct step in intensities between consecutive atom counts. Figure 6.19 shows the integrated intensities of the columns, normalized by the number of atoms in the column, to give the effective HAADF intensity per atom.

These results show a diminishing return in intensity as the number of atoms in a column increases. This is attributed to the diminishing increase in the breadth of the scattering cross section as further atoms are added to the columns. A reduction in the intensity of the beam close to the optical axis due to prior scattering events may also reduce the absolute high-angle scattering intensity from atoms further down the column. The diminishing increase in intensity is consistent with observations of intensity saturating at sample thicknesss above a certain threshold [103].

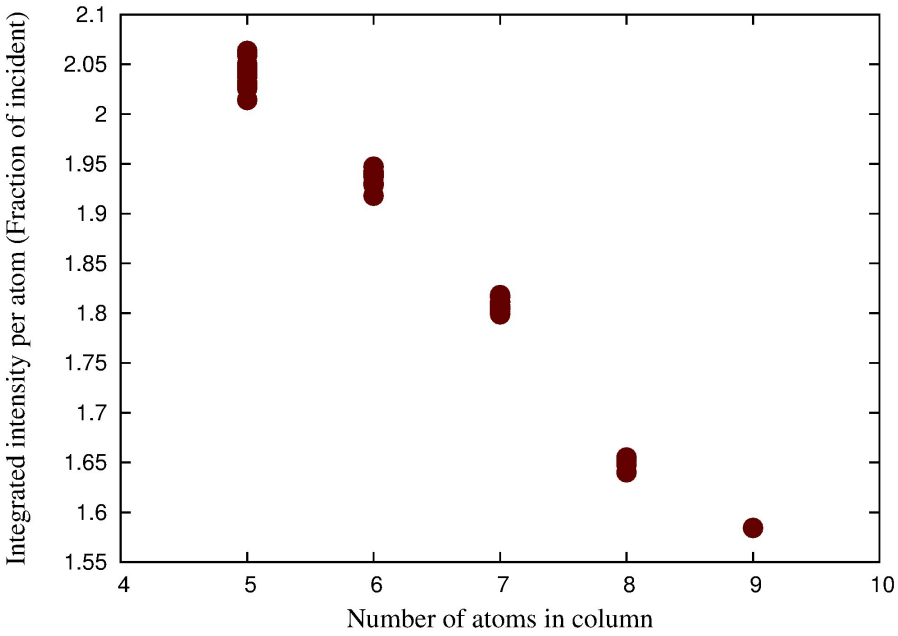

Figure 6.19: Integrated intensity per atom from simulation I.

In simulation II, the structure was relaxed using a conjugate gradient method with Gupta potentials, and thermal vibrations were modelled by bulk MSDs. The results of the quantitative analysis are shown in figure 6.20.

Simulation II also produces an almost linear, monotonic relationship between integrated peak intensity and atom count. This simulation exhibits some divergence in the intensities due to columns containing the same number of atoms. This occurs in both the 6-atom and 7-atom columns which both show two distinct intensity groupings. This is correlated with the position of the columns in the cluster; the lower intensity groups are populated by intensity peaks due to columns on the border of the five-fold symmetry whilst the peaks of greater intensity are caused by the columns in the interior of the FCC segments. To determine the cause of this local environment effect, the columns from the cluster were separated into a grid with a 2Å separation between neighbouring columns. This is sufficient to prevent overlapping intensities in the image as well as reducing the possibility of transverse interference and cross-talk of the beam between neighbouring columns. If any of these effects are persistent they should be identifiable, as they will be quantitatively different than in the image of the cluster structure. The shape of the columns, that is, the relative position of the atoms within the columns, was maintained. The results of this simulation exhibit the same intensities as the cluster, within the uncertainty due to the frozen phonon algorithm. The columns that were on the borders of the

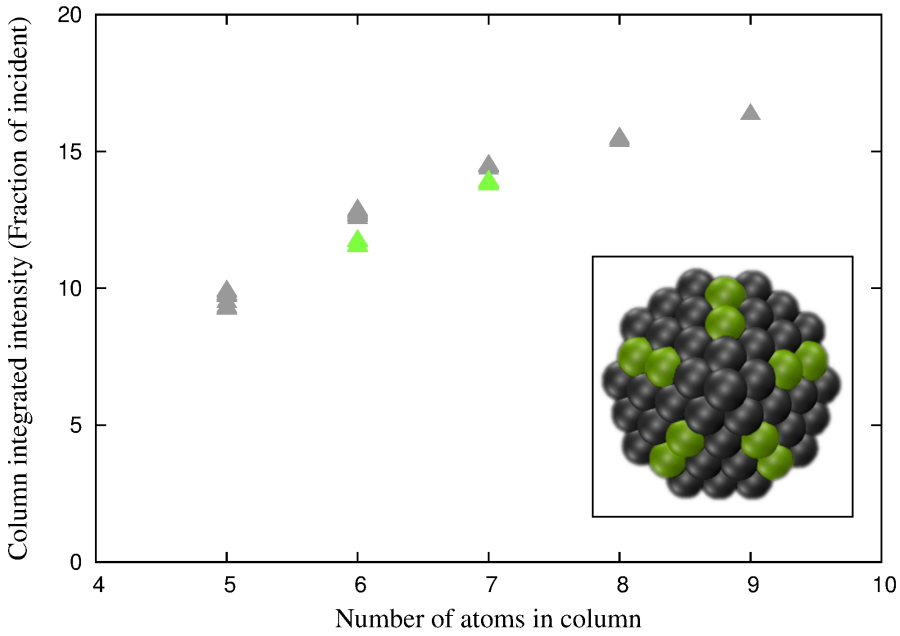

Figure 6.20: Relationship between atom count and peak intensity from simulation II using a relaxed structure with bulk MSDs. Inset image shows the location of twinned border columns which exhibit reduced intensities relative to interior columns of the same atom count.

five-fold symmetry still exhibit lower intensities than those from the interior. The intensity divergence can thus be attributed to the curvatures of the columns. From figure 6.8 it can be seen that the 6B and 7B columns have greater static displacements than their interior counterparts, 6I and 7I. The reduction in intensity is expected to be caused by a reduction in the beam focussing effect associated with channelling, as discussed in section 2.17. This will be discussed further in chapter 7. This hypothesis is supported by inspecting the intensities normalised by atom count in figure 6.21.

These results show less ordered behaviour than in simulation I, with little atom-count dependence.

The quantified results of simulation III, which used configurations from a molecular dynamics simulation at ~ 300 K, are shown in figure 6.22.

The intensities from this simulation are less well converged than in simulations I and II due to the larger thermal vibrations. There is generally a monotonic relationship between intensity and atom count in this simulation, however, the groupings are less distinct with some overlapping between the 5 and 6 atom column intensities. The central 9 atom column also exhibits a deviation from monotonic behaviour as its intensity is lower than those of

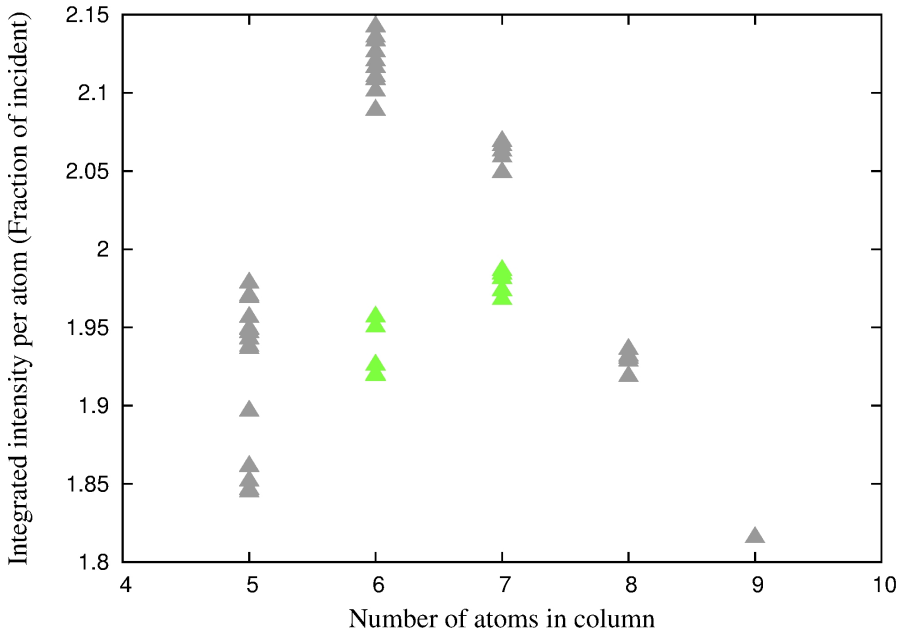

Figure 6.21: Integrated intensity per atom from simulation II.

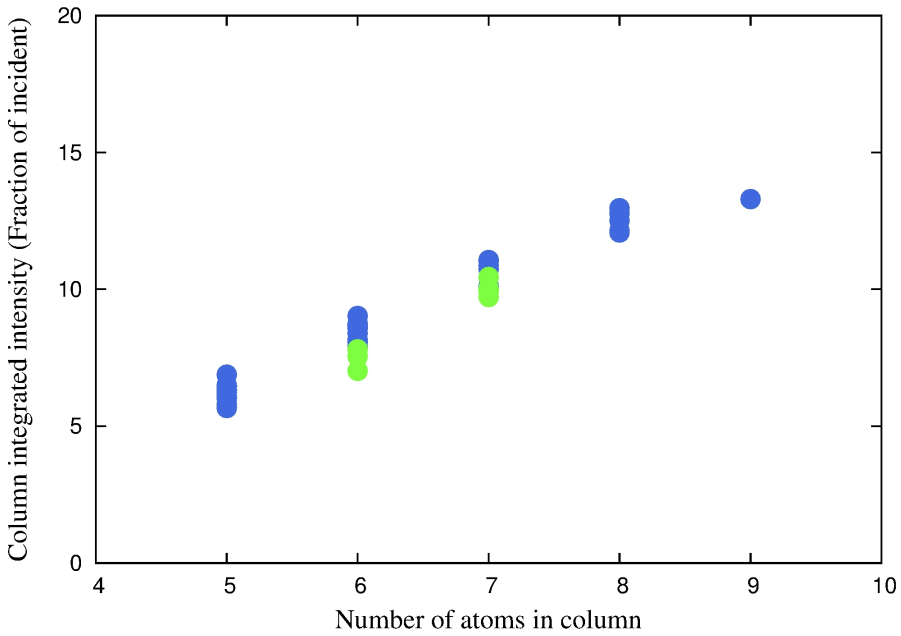

Figure 6.22: Relationship between atom count and peak intensity from simulation III using configurations taken from a molecular dynamics simulation. The green points are due to border columns as in figure 6.20.

several of the 8 atom columns. Diverging intensities in the peaks, due to interior and border columns, are again observed, though this is less clear than in simulation II because of the increased spread of values due to the thermal vibrations. The normalized intensities are shown in figure 6.23.

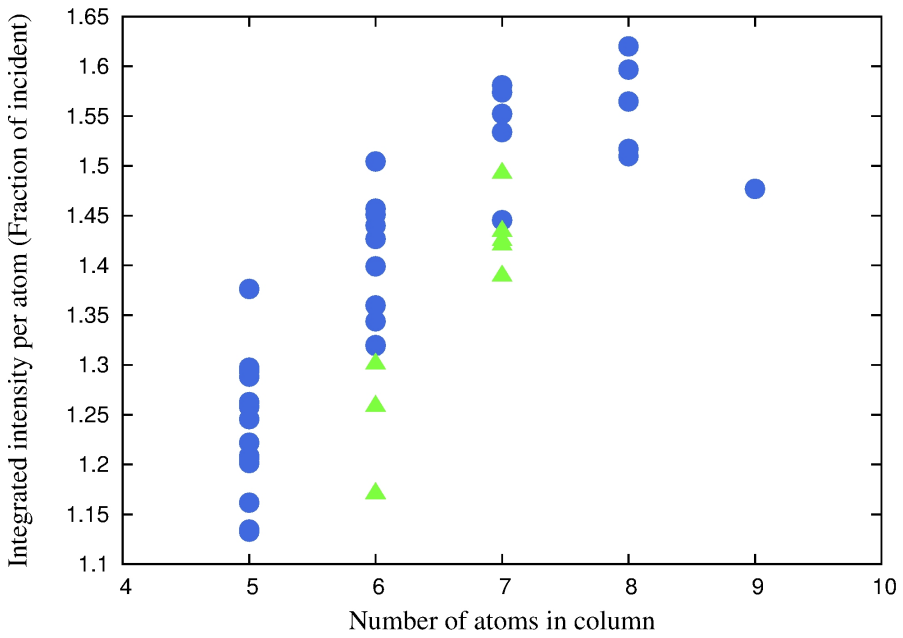

Figure 6.23: Integrated intensity per atom from simulation III.

In this case, with the exception of the central column, the intensity per atom is enhanced for atoms in longer columns. The exceptional behaviour of the 9 atom column could be due to its unique position in the cluster, with 5 nearest neighbours, or it could be due to its good alignment with the optical axis and small MSDs. The enhanced scattering per atom for atoms in longer columns is attributed to the focussing effect of electron channelling, this is investigated further in chapter 7.

Figure 6.24 shows the results of simulations I, II and III superimposed on the same axis.

The increased intensities of simulation II relative to simulation I are attributed to an increase in the scattering cross-section due to the relaxation of the columns. On examining the structures of the central 9 atom column of the two structures, it is found that the first four atoms of the columns are almost identically aligned with the optical axis, however, the lower 5 atoms in the relaxed structure are curved away from the axis and so present an increased cross-section. This interpretation would seem to be contradicted by the reduction in intensities in simulation III relative to simulation II. The time-averaged structure

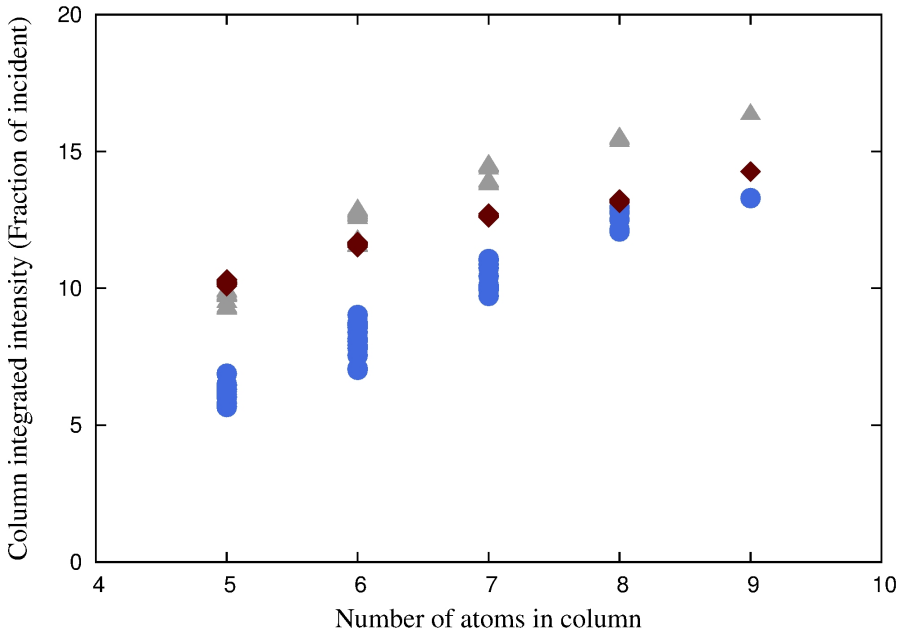

Figure 6.24: Column integrated intensities from simulations I (Red diamonds), II (grey triangles and III (blue circles).

from the molecular dynamics simulation used in simulation III is similar to structure II in terms of the magnitude of static displacements, so the difference in intensity is attributed to the greater thermal motion in simulation III. However, in this case the displacements from perfect alignment cause a reduction in peak intensities, rather than an increase. Considering the scattering per atom in these simulations offers a hypothesis to explain these seemingly contradictory observations. In simulation I, the regular structure with perfectly aligned columns and small bulk MSDs, the longer columns exhibit reduced scattering per atom. This is consistent with the interpretation that the probability of scattering is linked to the size of the scattering cross-section, thus if the cross-sections of two atoms overlap, the total cross section is lower than if they were separate. In simulation III, the molecular dynamics simulations with strain-induced static displacements and larger thermal vibrations, the longer columns exhibit increased scattering per atom. This is consistent with the channelling interpretation in which the electron beam is focussed onto the column resulting in greater scattering for atoms in a column than individually. These observations suggests that there are two different regimes, one for cases of low structural disorder and one for larger disorder. At low structural disorder, the intensity increases with additional disorder, due to an increased cross-section.At high structural disorder, intensities decrease with additional disorder, due to an attenuation of channelling. The results of simulation II are consistent with this, as the more statically disordered short columns fall in the inten-

sity enhancement regime, whilst the less disordered central columns fall in the intensity reducing regime. This hypothesis is discussed further in the following chapter.

6.5 Sample tilting

It is well established that tilting a sample away from zone axis alignment effects the extent to which channelling occurs, and, consequently, reduces the HAADF-STEM contrast. Small metallic nanoclusters are inherently unstable, particularly under beam exposure and so obtaining perfect zone axis images is difficult. In this section, the sensitivity of HAADF-STEM images to mistilt is investigated by performing a series of simulations at varying mistilts. In each case, identical frozen phonon configurations from a molecular dynamics simulation are used so that any differences can be attributed solely to mistilt. The multislice simulation methodology used in this section is the same as that in the rest of the chapter. It is important to note that the cluster rotations are performed by multiplying the atomic coordinates by a rotation matrix. Many multislice simulations offer a means to replicate sample rotation by applying incremental offsets of the transmission slices transverse to the optical, axis but the accuracy of this method is poor, particularly for larger rotations. The results of 8 simulations at differing mistilts are shown in figure 6.25. The tilted images exhibit asymmetrical intensity distributions, with off-centre columns becoming brighter at certain mistilt angles than under perfect zone-axis alignment. A prominent example is indicated with a white arrow in the 0.1 ° mistilted image. This is attributed to the rotationally symmetric curvatures of the columns as tilting the can cluster improve the alignment of the first few atoms of columns to one side of the cluster, as illustrated in figure 6.26.

Similar effects, with unexpectedly bright off-centre columns, are often seen in experimental images, such as that in figure 6.27, which shows an experimental HAADF-STEM image of a gold nanocluster similar to those modelled here.

In this example, a small mistilt can clearly be observed due to the asymmetrical intensity spots. This observation could be of significant value to the interpretation of experimental HAADF-STEM images of such small pseudo-spherical nanoclusters, as it offers a means by which even small mistilts can be observed and accounted for by inclusion in comparative image simulations. To assess the importance of mistilt in atom counting procedures, the quantitative analysis process as been applied to each of the images in figure 6.26. The variation of the integrated intensity of the entire cluster with mistilt angle is shown in figure 6.28. This shows a dramatic reduction in intensity from the perfectly aligned image to a mistilt of just 0.1°. At greater mistilts the rate of reduction in intensity

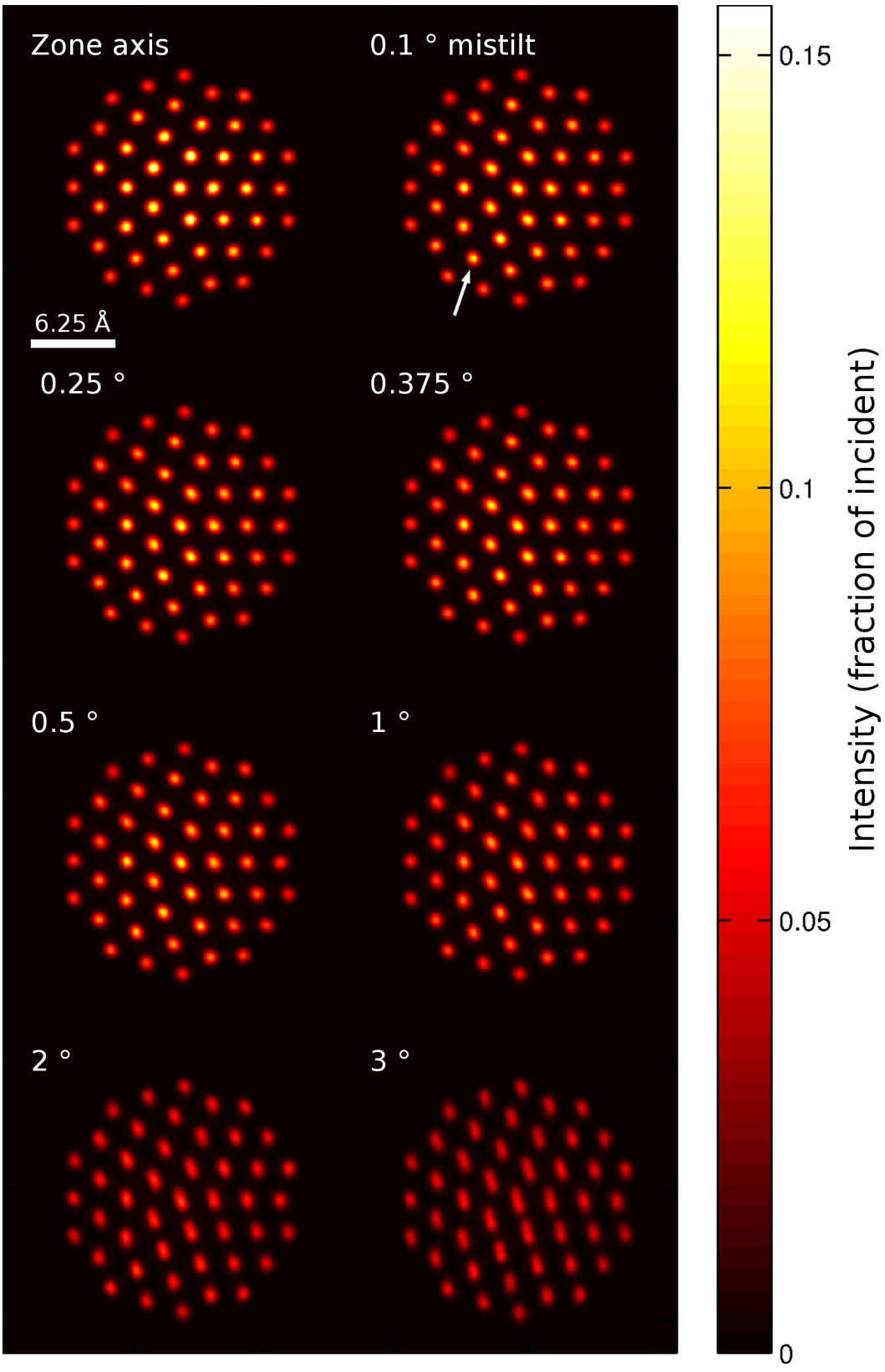

Figure 6.25: Tilt series for a 284 atom gold nanocluster.

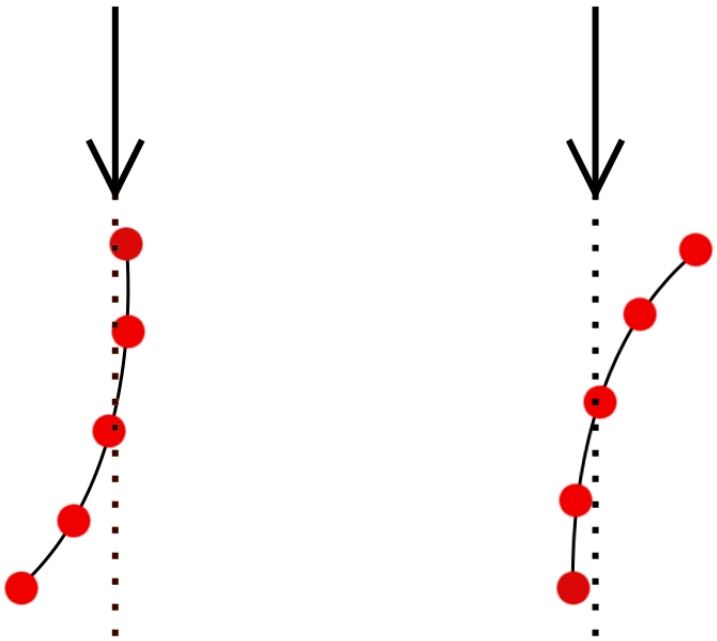

Figure 6.26: Tilting the cluster improves the alignment of some columns bringing them into channelling conditions and increasing scattering intensity.

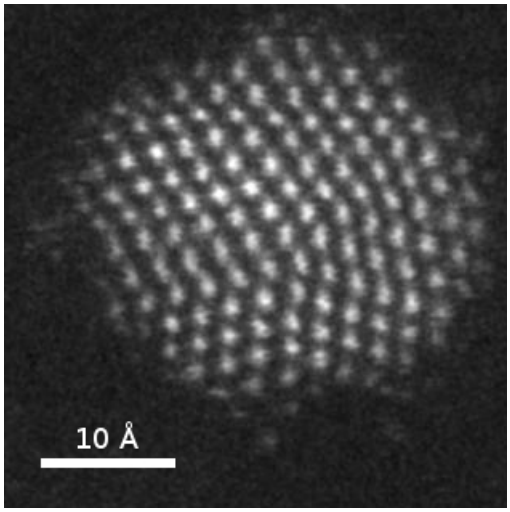

Figure 6.27: Experimental image of a small gold nanocluster with irregular Marks decahedral morphology exhibiting an off-centre high intensity region. Courtesy of Dr D. He, University of Birmingham.

with mistilt drops, it is not observed here, but the sensitivity to tilt would be expected to entirely disappear far from a zone-axis orientation where no channelling occurs. In the literature, a multislice simulation of a single palladium column found this to occur at a mistilt of approximately 5 ° [101].

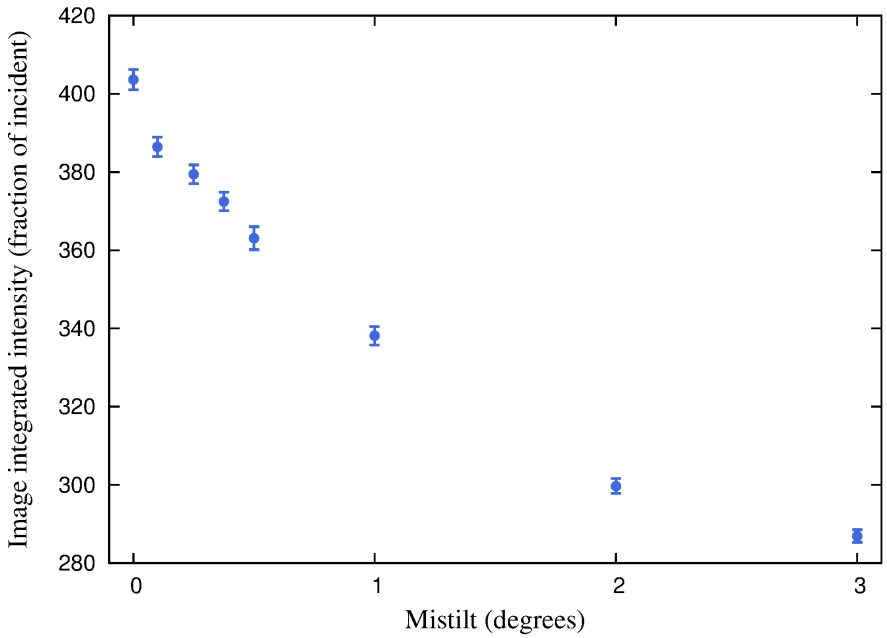

Figure 6.28: Image integrated intensities from the tilt series.

The reduction in intensities observed here is not evenly distributed amongst the peaks, the peaks corresponding to longer columns show greater reductions in intensities, as seen in figure 6.29.

Mistilting the cluster out of zone-axis alignment causes a reduction in intensities because the channelling effect is diminished. This has a relatively greater effect on longer columns as they are more susceptible to channelling because their greater length allows the beam to be focussed onto the column to a greater extent. It is useful to note that a relatively greater reduction in the intensities of longer columns is indicative of a reduction in electron channelling, as this informs the analysis in the following chapter.

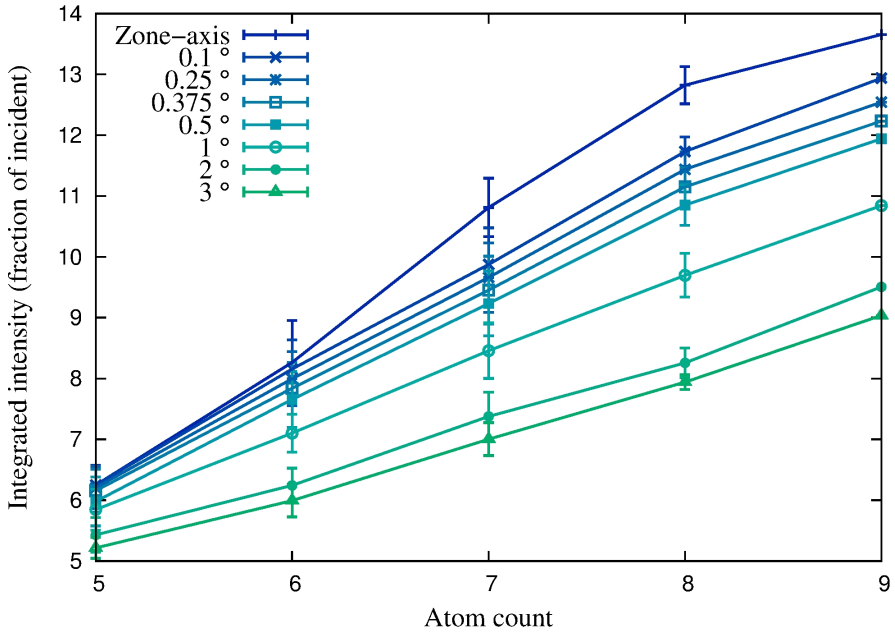

Figure 6.29: Peak integrated intensities from the tilt series.

6.6 Implications for atom counting

As described in section 2.3.8, there are two main approaches to extracting atom counts from HAADF-STEM images: those relying on comparisons with simulations, and those based on statistical parameter estimation techniques. For comparisons between experimental and simulated images, it is desirable to calibrate the images such that they express the intensities as a fraction of the incident beam. This facilitates comparisons on an absolute scale and so reduces the likelihood of features being overlooked due to the use of scaling parameters. Such absolute comparisons require rigorous simulations including all the known parameters. The results of the three simulations are shown together in figure 6.24. In comparing simulations I, II, and III the most obvious difference is the large reduction in intensities in simulation III due to the greater thermal motion observed in the molecular dynamics simulations. The structural relaxation also plays an important role, as can be seen from the reduced intensities of the peaks due to border columns compared with those from the better aligned interior columns. It is clear from these results that to pursue accurate structural characterizations through quantitative comparisons of simulated and experimental images, it is vital to account for the inhomogeneities observed at the nanoscale. The rigorous simulation method developed here offers a means by which to this can be accomplished, however, experimental comparisons are still needed to determine the accuracy of the method. The strong sensitivity of image intensities to mistilt presents a significant problem in achieving quantitative comparisons between simulations and experiments. Whilst reproducing a known mistilt in simulations is not problematic, determining that a mistilt is present in the experimental image is difficult, unless the sample is sufficiently tilted that the peaks are visibly asymmetrical. In the case of small multiply twinned nanoclusters, the varying tilt sensitivity of atomic columns with pseudo-spherical curvatures offers a means by which mistilt can be determined. The simulations here produced notable intensity variations visible at 0.1° mistilt.

Analysis methods based on statistical parameter estimation rely on a monotonic relationship between atom count and peak intensity without more detailed physical interpretation. To assess the applicability of such techniques to single-shot images of small nanoclusters, an image is needed for which the structure is known. Producing structures to specified geometries is not within the capabilities of current fabrication methods, nor are there any alternative characterisation methods of sufficient resolution. To address this issue, the analysis techniques described by Van Aert *et al.* have been applied to the results

of simulation III. Simulation III is more rigorous than any previously reported multislice simulation as it includes all known sample parameters thoroughly. As such, it is well suited to testing the analysis procedure.

A dataset has been constructed which consists of the integrated intensities of the 46 peaks seen in figure 6.17 c). An iterative process of fitting increasing numbers of Gaussian components to the dataset has been undertaken using the integrated classification likelihood (ICL) criterion to determine the optimum number of components. In this case, the ICL gives an optimum of 6 components, indicated by the minima in figure 6.30.

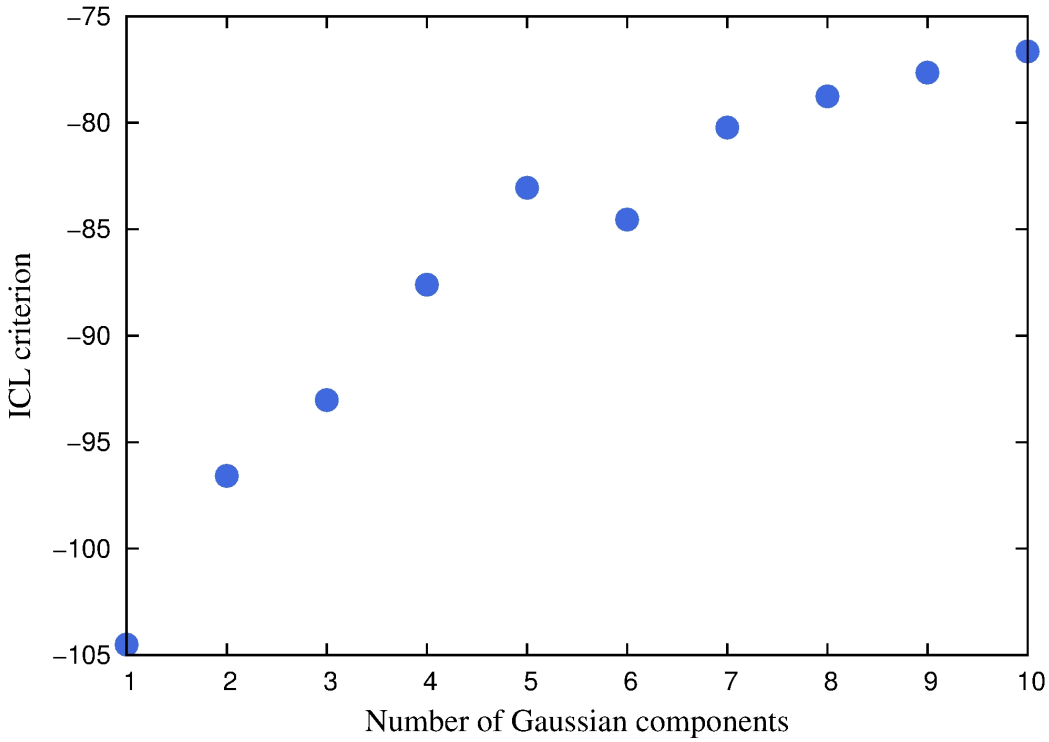

Figure 6.30: Integrated classification likelihood criterion shows a minima at the optimum number of Gaussian components to fit the intensity dataset.

The Gaussian fitting for this number of components can be seen superimposed on a histogram of the dataset in figure 6.31.

The mean intensity values of the fitted Gaussian peaks are approximately equally separated in intensities, as shown in figure 6.32. This suggests a good monotonic relationship between peak intensity and atom count.

This suggests that there are 6 different column lengths in the structure. This analysis method does not directly yield an absolute atom count, rather, it produces a series of components at successively greater intensities. In the case that one of the components has

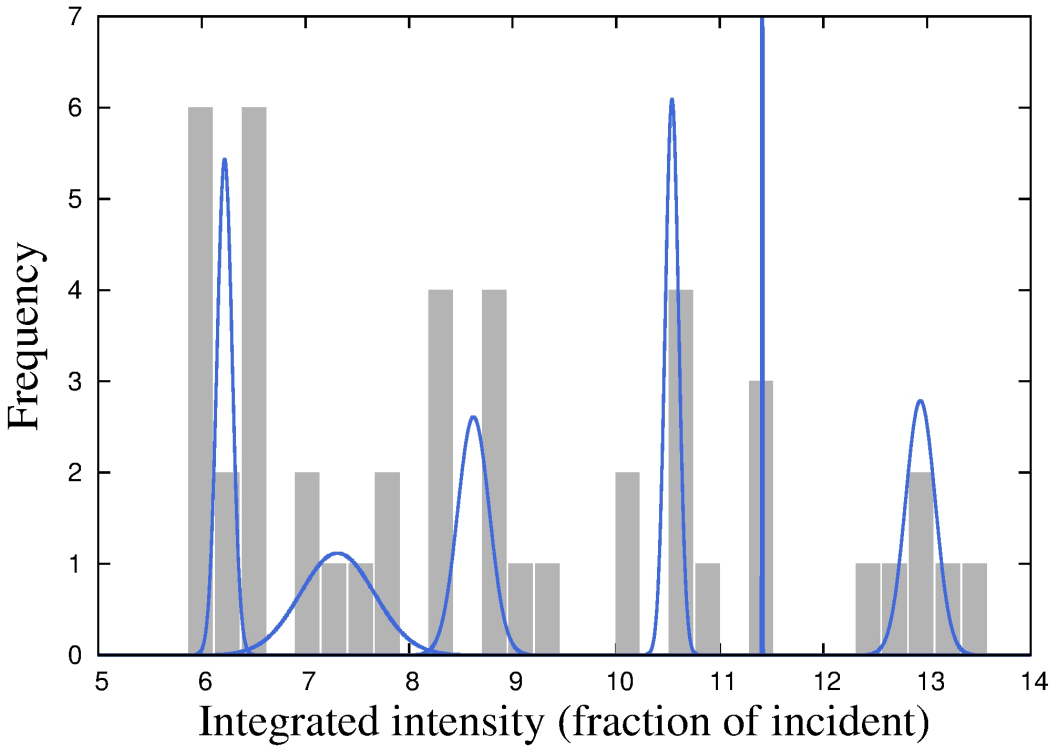

Figure 6.31: Results of the statistical parameter estimation. Six Gaussian components are identified with approximately equidistant mean intensities as shown in the inset graph.

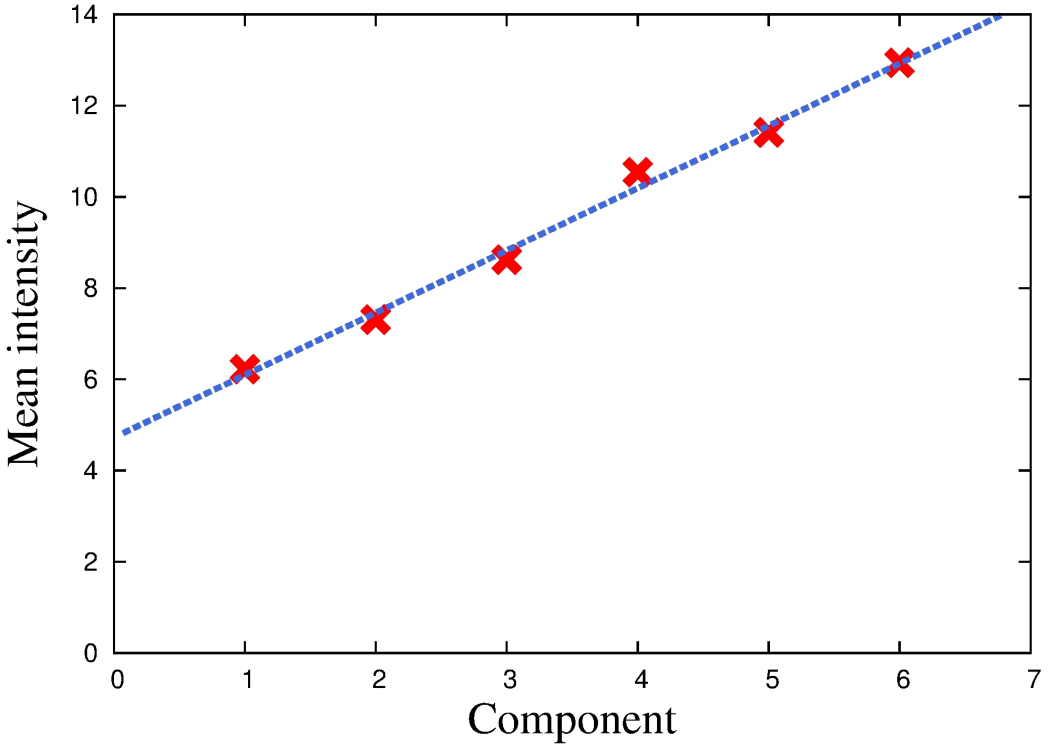

Figure 6.32: Mean intensities of the fitted Gaussian peaks in figure 6.31.

an absolute intensity similar to the relative intensity between adjacent components, it can be reasonably assumed that this is the single-atom component, and successive components can be assigned atom counts relative to it. As the simulation used here was performed for an idealised sample with no surface ad-atoms, there is no single-atom component to extrapolate from so only the relative atom counts can be considered here.

Correlating the known sample structure with the intensity dataset reveals that the statistical parameter estimation technique has successfully split the dataset into groups due to columns of different atom counts, with the exception of the peak due to the 9 atom column which was not discernible from the 8 atom columns. Additionally, the fitting process distinguished between columns of the same atom count but different curvatures (this strain-induced divergence was described in section 6.4). Consequently, the Gaussian components correspond to the 5, 6B, 6I, 7B, 7I and combined 8 and 9 columns as labelled in figure 6.8. In the literature, atom counts have been assigned in a trivial manner with an additional atom count for each Gaussian component. In this case, the 6I columns would be assigned an additional atom relative to the 6B columns and, similarly, the 7I columns would be assigned an additional atom relative to the 7B columns. The cascading over count which results from this trivial assignment would lead to an overcount of 36 atoms for the 284 atom cluster. In applying this method to experimental images, the error could be identified using the intensity of isolated atoms to give an indication of the magnitude of the intensity steps between atom counts, however, as discussed previously, the intensity per atom is expected to very depending on the properties of the structure.

Mistilting of the sample introduces further difficulties for this statistical parameter estimation procedure. As noted in the previous section, the mistilting of curved columns improves the alignment of some, whilst reducing the alignment of others, this leads to a broadening of the intensity distributions of columns containing the same number of atoms. This produces unexpectedly bright spots for the well-aligned columns which will result in a miscount in many circumstances where atom counts are trivially assigned. Whilst a few columns become better aligned with mistilt, overall the alignment is reduced, causing a reduction in electron channelling and a reduction in intensities. This affects longer columns more than the shorter columns, and so the sensitivity of peak intensities to atom count is reduced.

6.7 Ino-decahedral cluster

In section 2.3.8, a report by Li *et al* was discussed in which a comparison is drawn between an experimental image of a decahedral cluster and a conventional simulation of a 309 atom ino-decahedral structure [35]. In that report it is concluded that the match between image and simulation is sufficiently good to identify the experimental cluster as a 309 atom ino-decahedron, however, some discrepancy at the cluster surface was noted and attributed to the greater thermal vibrations of the under-coordinated atoms. This discrepancy provided some of the impetus to investigate the effects of realistic thermal vibrations. A simulation of a 309 atom ino-decahedral gold cluster has been performed using molecular dynamics simulations in order to assess the validity of the attribution to thermal effects. For this simulation, the same microscope parameters listed in table 6.2 were used. To replicate the beam broadening effects of the microscope, caused by the finite source size and instabilities, the simulated image has been convoluted with a Gaussian with a full width half maxima of 0.8 Å. The resulting image is shown in figure 6.33a), alongside line profile plots in figure 6.33b). During the molecular dynamics simulations an atom shifted from an edge column onto one of the (100) surfaces, the column is indicated by a white arrow.

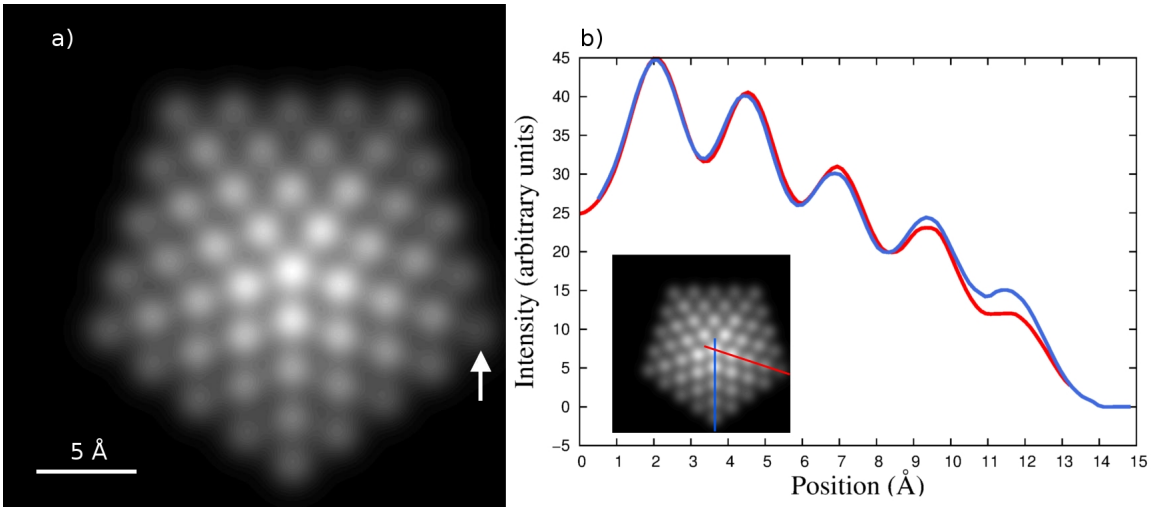

Figure 6.33: a) Simulated image of a 309 atom ino-decahedral gold nanocluster generated using frozen phonon configurations from a molecular dynamics simulation, the arrow indicates an edge column from which an atom was displaced during the course of the simulation. b) The line profile with a missing atom from the surface column shows better agreement with Li's experimental image [35]

In comparing the line profiles from this simulation with those in the original report (reproduced in figure 2.13), it can be seen that the surface columns exhibit lower intensities than those in the conventional simulations. Whilst the profile including the regular,

fully occupied surface column exhibits a lower surface intensity than in the conventional images, the peak is still well resolved and has a greater relative intensity than the outer column of the experimental image. The profile with the under occupied surface column, however, offers a better agreement, exhibiting a shoulder rather than a defined peak. The displacement of an atom from the column also provides an explanation for the small additional shoulder at the end of the experimental profile. Consequently, the author concludes that the discrepancy between the simulated and experimental images in Li's report can be partially attributed to enhanced surface vibrations, but that it is also likely that there was a missing or displaced atom during image acquisition. The edge columns of the inodecahedral are especially under-coordinated, and structures in which they are not present (Marks decahedra), or are under-occupied, are more energetically favourable, so such a defect is likely to occur [64].

6.8 Chapter conclusion

In this chapter a new method combining molecular dynamics and multislice simulations has been introduced. This method solves the problem of the lack of atomic-resolution experimental data on the thermal motion and structural relaxations of small metal nanoclusters. Both of these factors have been found to affect multislice image formation and so this new method produces images that are both quantitatively and qualitatively different from those produced using conventional methods. These results have confirmed the need to account for these factors in such a rigorous manner. The affects of mistilt on nominally zone-axis images of small gold nanoclusters has been investigated using this simulation method. The resulting images exhibit unexpectedly bright off-centre intensity spots, as are often seen in experimental images. These bright spots are attributed to an improvement in the alignment of curved columns, causing an in increase in electron channelling, which results in increased high-angle scattering as the beam is tightly focussed onto atoms in the column. These bright spots may prove valuable in discerning mistilt in such images. The highly-localized strain in multiply-twinned gold nanoclusters results in different geometrical arrangements of columns containing the same number of atoms, depending on their position in the cluster. The greater curvatures of columns lying along twin boundaries than those in the interior of FCC segments was found to cause a reduction in integrated peak intensities in zone-axis HAADF-STEM images. This caused a bimodal distribution of intensities for columns of the same length.

The sensitivity of image intensities to mistilt from the zone-axis is expected to present problems in analysing experimental images with reference to simulated images, especially in cases where the cluster rotates during image acquisition. The statistical parameter estimation technique proved to be an effective tool in identifying the intensity peaks due to equivalent columns in HAADF-STEM images of small nanoclusters. Indeed, the method employed here was sufficiently accurate to discriminate between intensity spots produced by columns containing the same number of atoms but with different curvatures. This highlights the need for informed physical interpretation when analysing images of inhomogeneous structures, this can be provided by rigorous image simulations. With improved microscope stabilities the strain sensitivity observed here could prove useful in measuring strains from single HAADF-STEM images.

A simulation of an ino-decahedral gold cluster has been performed to assess the accuracy of Li's attribution of reduced surface contrast to enhanced thermal vibrations [35]. It has been found that, whilst greater thermal vibrations did reduce the intensity of surface columns, the effect was not significant enough to account for Li's experimental observations. The reduced intensity observed is expected to be caused by a missing atom during image acquisition.

Chapter 7

Temperature effects

7.1 Introduction

In the previous chapter, a new method was introduced to account for sample inhomogeneities in multislice simulations using molecular dynamics simulations. The images produced using this new method exhibited reduced intensities compared with those produced using more conventional simulations. The reduction in intensity was attributed to the increased thermal motion observed in the molecular dynamics simulation.

The prevailing wisdom in electron microscopy has historically been that the principle effects of thermal vibrations on HAADF-STEM images are the destruction of coherence and an increase in the breadth of the scattering cross-section, resulting in increased high-angle scattering [7, 47]. However, in addition to the results reported in the previous chapter, there have been several reports in the literature of HAADF-STEM intensities reducing with increased temperatures [19, 7, 104].

To address this contradiction, this chapter begins with a systematic study of the effects of thermal vibrations on HAADF-STEM image formation. Following this, the effects of cryogenic and elevated temperatures on images of a prototypical gold nanocluster are predicted through a series of simulations at varying temperatures. Throughout this chapter the discussions are framed in terms of thermal vibrations, however, the effect of the vibrations is to introduce structural disorder into the crystalline samples. In this sense, the investigations are also relevant to structural disordering caused by strains and defects.

Accelerating voltage	200 keV
Defocus	0 Å
Convergence angle	20 mrad
Detector range	90 230 mrad
Slice thickness	1.4 Å
Frozen phonon configurations	20
Transmission function size	4096^2 pixels / 77 $\times$ 77 Å
Probe function size	2048^2 pixels / 35.5 $\times$ 35.5 Å

Table 7.1: Multislice simulation parameters for systematic studies of thermal motion effects.

7.2 Systematic studies

In order to address the paradoxical results reported in the literature a number of simulations have been performed. These simulations make use of physically unrealistic structures which allow greater insights into the image formation process than more complicated real-world structures by removing crystallographic imperfections and thereby isolating the Mean Square Displacement (MSD) parameter as the only variable.

A test model has been constructed which consists of 1197 gold atoms in a bulk-like FCC crystalline geometry. The columns are 9 atoms deep along the optical axis and have been assigned a variety of differing MSDs. Conventional frozen phonon multislice simulations have been performed using the microscope parameters in table 7.1.

In this case, the conventional frozen phonon simulation is preferred so that thermal effects can be isolated from other factors affecting image intensities. A range of MSD values between 0 and 0.25 Å^2 were used, for reference, the MSD of bulk crystalline gold is 0.0079 ± 0.0002 Å^2 at 295 K and the nearest neighbour distance is 2.88 Å [72]. The image resulting from this simulation is shown in figure 7.1.

The quantitative analysis procedure described in section 6.3.1 has been applied and the resulting peak intensities have been linked to the MSD values of their corresponding columns. The peak intensities are plotted against their respective MSD values in figure 7.2.

The image has been quantified in terms of both peak intensities and integrated Voronoi cell intensities. Measuring both the peak intensities and Voronoi cell intensities is insightful because it is expected that channelling should cause an increase in peak intensities with

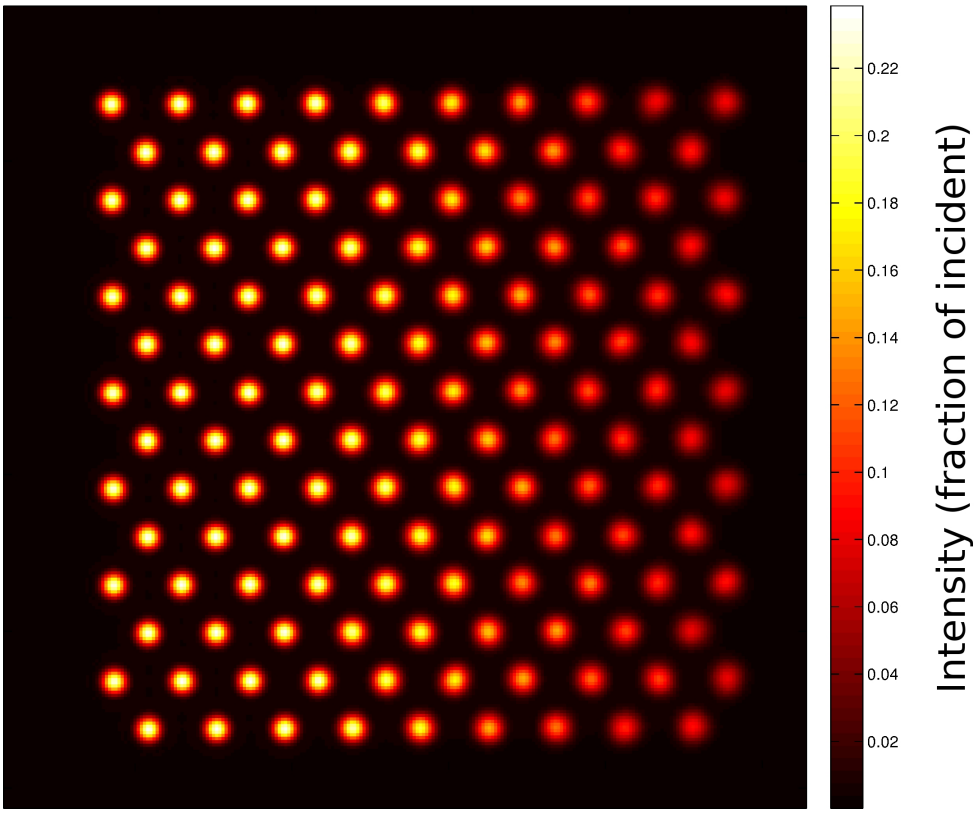

Figure 7.1: Multislice simulation of an array of 9 atom deep Au columns with increasing MSD values from left to right.

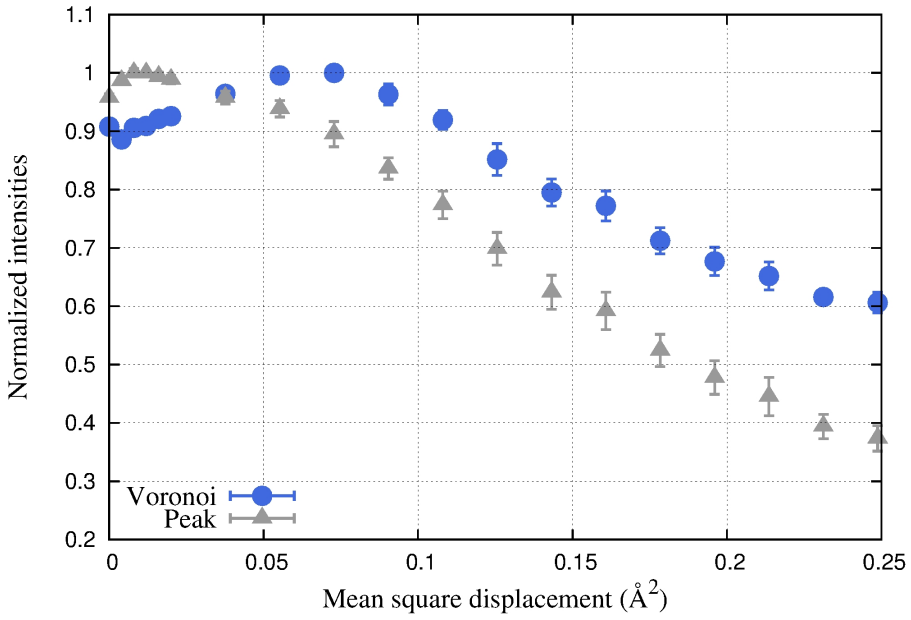

Figure 7.2: Effect of thermal vibrations on 9 atom deep Au columns using both peak and Voronoi quantification. The intensities are normalized to the greatest values.

a lesser effect on intensities further away from the peak, whilst an increase in the area of the scattering cross-section should produce an increase in scattering away from the peak and a reduction in peak intensities. The results in figure 7.2 are normalized to allow visualization on the same axes. In both the peak and Voronoi cases, two distinct regimes are observed in the results. At low MSDs the intensity increases with MSD whilst at higher MSDs the intensity is reduced. In the case of the peak intensities the turning point occurs at approximately 0.01 $Å^2$ whilst the Voronoi integration method gives a turning point at approximately 0.07 $Å^2$.

In the previous section it was proposed that two regimes of thermal scattering existed, the low-disorder regime in which the effective area of the scattering cross-section dominates the intensity sensitivity, and the high-disorder regime in which diminishing channelling effects dominate. This systematic study supports this hypothesis. It can be seen from figure 7.2 that for small MSDs the increase in scattering cross-section with MSD is the dominant effect. The later turning point of the Voronoi intensities indicates that, whilst peak intensities are being reduced by diminished channelling, the increased scattering away from the peak centre results in an increase in overall scattering. At higher MSDs, however, the increased cross-section no longer compensates for the reduction in electron channelling down the column and the overall scattering intensity begins to drop.

To test that this observation is not unique to the scattering factor of gold, the simulation has been repeated with an identical structure but using the scattering factors of copper atoms. The scattering factors of gold and copper used in the simulations are shown in figures 7.3 and 7.4 [7].

The gold atoms have a far larger potential than the copper atoms at small radial distances, this is expected to increase the likelihood of electron channelling in gold columns [104]. The copper atoms have greater scattering factors above approximately 2 Å however, the potential is small this far from the centre of the atom and is expected to have a minimal effect on the electron beam.

In the multislice simulation of the copper sample, the geometry of the gold system is retained, using the crystalline gold lattice parameters in order to isolate the effect of changing the scattering factor. The image resulting from the simulation has been quantified in terms of peak intensities and the results are shown in figure 7.5.

This simulation also yields two distinct regimes though in this case there is a less no-

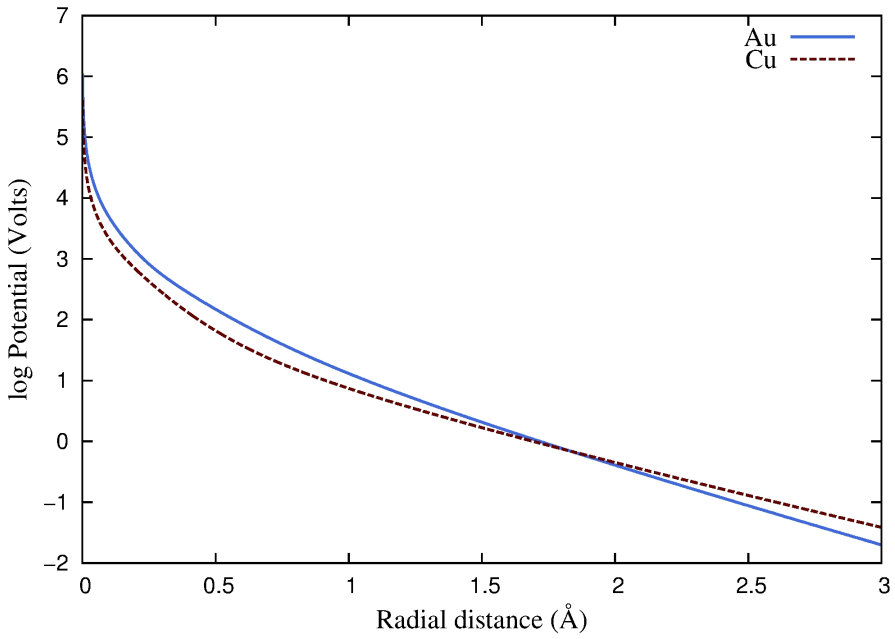

Figure 7.3: Scattering factors of Au and Cu over 3 Å radius.

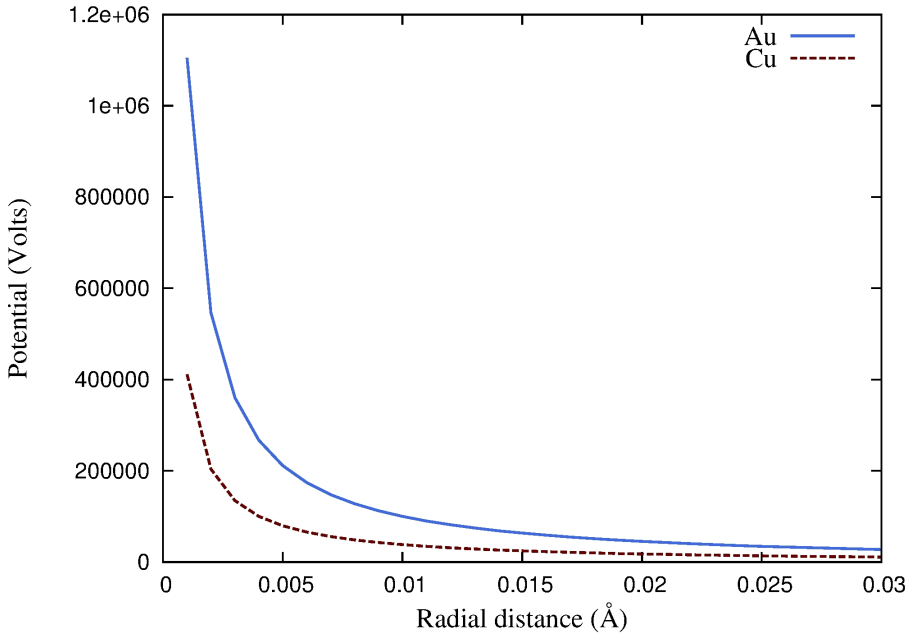

Figure 7.4: Detailed view of the scattering factors of Au and Cu over 0.03 Å radius.

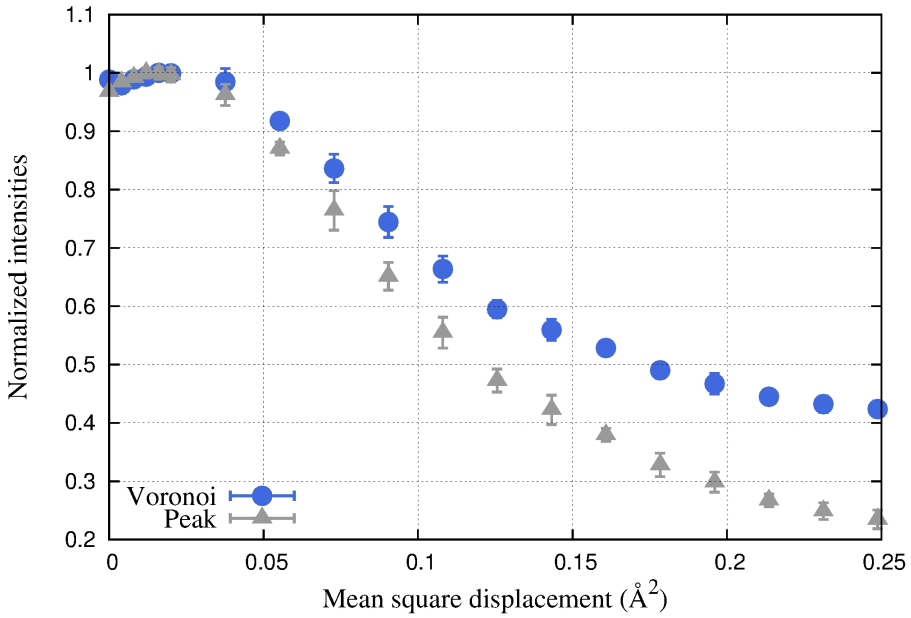

Figure 7.5: Effect of thermal vibrations on 9 atom deep Cu columns using both peak and Voronoi quantification. The intensities are normalized to the greatest values.

table separation between the turning points of the peak and Voronoi intensities. This could be due to more rapidly diminishing electron channelling with increasing MSD, which would be expected due to the narrower scattering factor of copper.

To observe the electron channelling phenomenon, three multislice simulations have been performed, during which the wavefunction has been recorded just prior to multiplication with the transmission slices (those containing the scattering factors). These simulations were performed using the same parameters listed in table 7.1. Of these simulations, the first is a null result in which the beam freely propagates along the optical axis, in the second simulation the beam is focused directly onto a perfectly aligned column of 30 gold atoms separated by 1.44 Å along the beam axis. In the third simulation the gold atoms are replaced with copper atoms with the same 1.44 Å spacing. To visualize the propagation of the beams the intensities of the wavefunction at each slice have been calculated and a linear interpolation has been performed between each of the slices. The resultant 3D beam has been projected into 2D. The linear interpolation does not correctly model the propagation of the beam between the slices, however, it does allow the major features to be visualized. The interpolation also required pixel values to be quantized to 8-bits resulting in a coarsening of the intensity levels, which are calculated in double precision. The three results are shown in figures 7.6, 7.7 and 7.8. Only the central 3 Å of each beam is displayed as this region exhibits the key features whilst maintaining a readily interpretable

contrast range. The positions of the atoms are superimposed as green circles on the projected beam images and the intensity profiles of the beam at 0, 0.5 and 1 Å from the beam centre are plotted beside each propagation image. It should be noted that the beams in the figures have been stretched transverse to the direction of beam propagation so that the features may be seen clearly.

The freely propagating beam, figure 7.6, exhibits very little variation during the 40 Å propagation, though a small reduction in the axial intensity and corresponding increases in the off-axis intensities indicate the diffraction of the beam after its focal point at 0 Å. The beam focussed onto the gold column, shown in figure 7.7 shows significant scattering to high angles and oscillatory intensities along the beam axis. These oscillations are the electron channelling effect, in which the attractive potential of the atoms draw the beam into the column. The focussing effect can be seen with bright peaks at the axis that are of greater intensity than the initially focussed beam. For the gold scattering factors, the channelling oscillates with intensity peaks separated by 9.3 ± 0.3 Å. The beam propagating along the copper column, figure 7.8, exhibits lesser high-angle scattering in comparison to the beam propagating along the gold column, due to the smaller scattering factor of the copper atoms. The beam is focussed less frequently by the copper atoms, with peaks separated by approximately 24 Å, and has a larger depth of focus. The peak intensity of the focussed beam is greater along the copper beam than in the gold case, this is unexpected. This could be due to greater high-angle scattering by gold atoms prior to the focal point resulting in greater beam broadening, and so less intensity at the focal point. There are few gold atoms before the first focal point which casts some doubt on this explanation. The greater scattering potential of the copper atoms beyond 2 Å from their centre, as seen in figure 7.3, could also be responsible. To observe how thermal motion effects the electron channelling, beam propagation has been recorded for another three columns containing 30 gold atoms. For each of these simulations a frozen phonon configuration has been generated. The three structures have MSDs of 0.027 Å^2, 0.066 Å^2 and 0.122 Å^2, the beam propagation down these columns is shown in figure 7.9.

With a MSD of 0.027 Å^2, electron channelling is still significant in figure 7.9a), though focussing occurs less frequently and with less intensity than in the perfectly aligned column. This MSD falls within the regime in which increasing disorder increases high-angle scattering, as evidenced by an increase in intensity of 11.8% with respect to the perfectly ordered column. A further increase in MSD to 0.066 Å^2 also gives a further increase

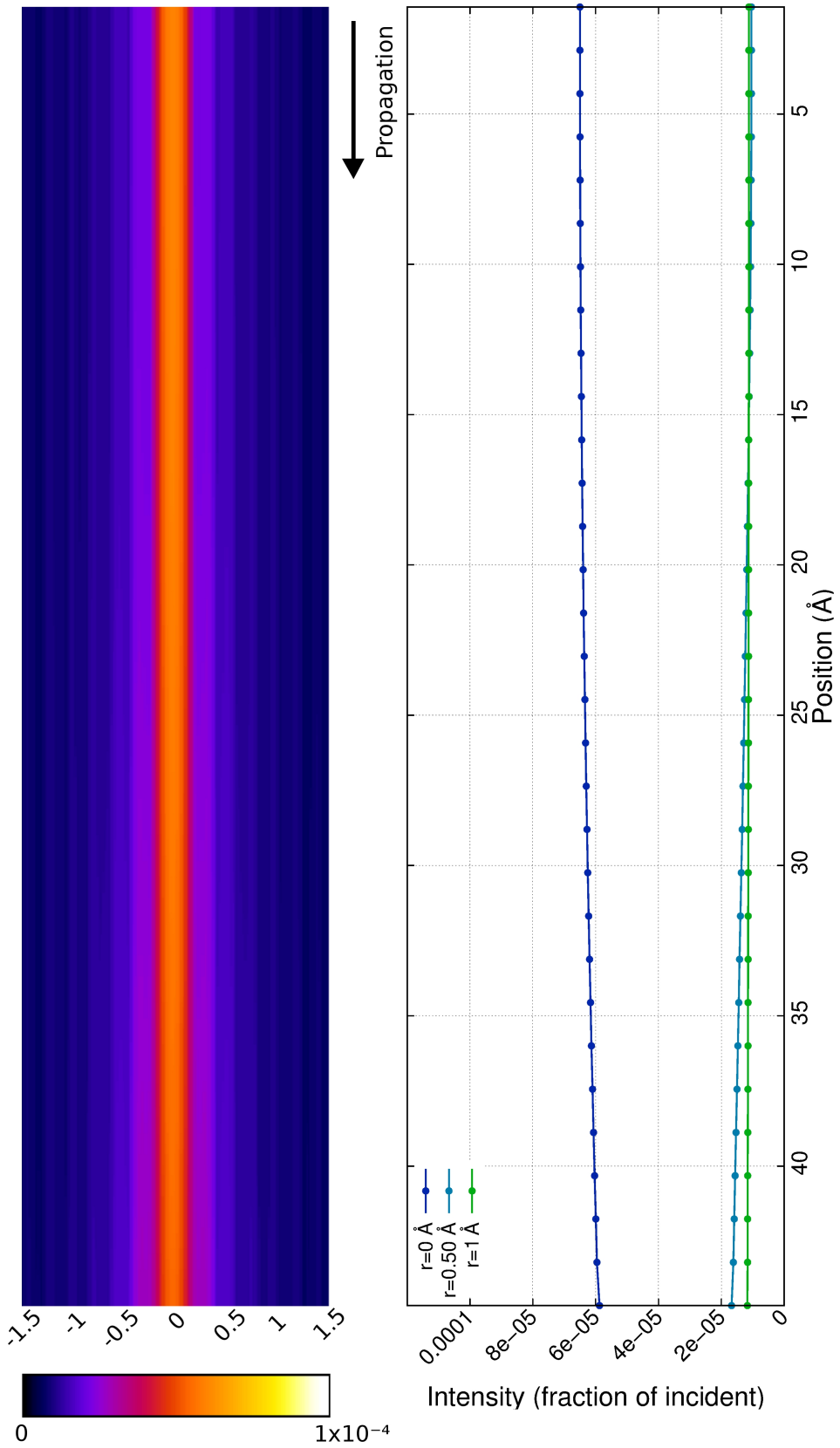

Figure 7.6: Beam propagation during a multislice simulation. With no sample interaction the beam exhibits a small broadening after focussing at the top of the figure.

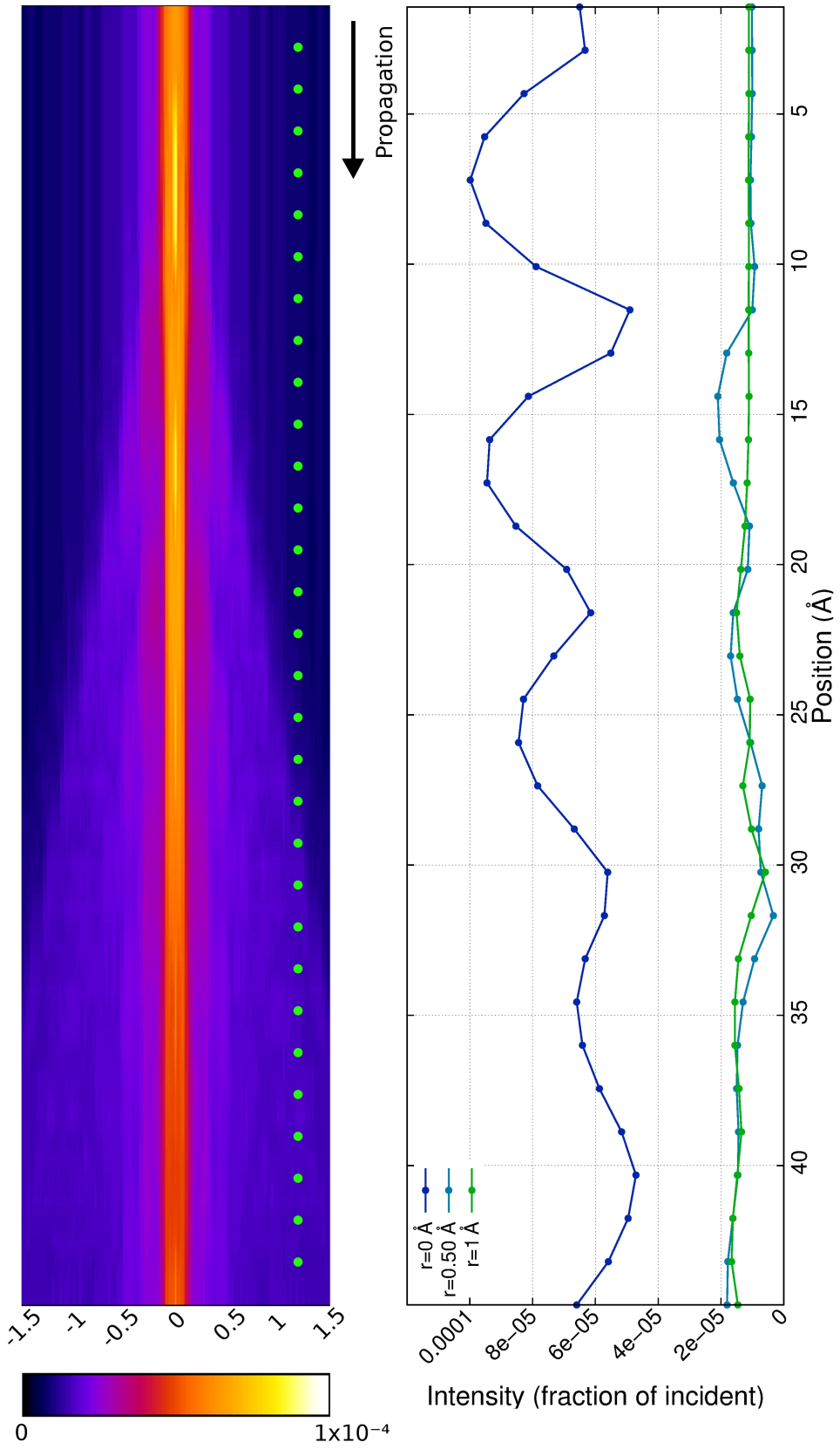

Figure 7.7: Beam propagation down a column of 30 Au atoms, the height of the atom are indicated by the green circles, offset from their positions on the beam axis to give an unobstructed view of the beam.

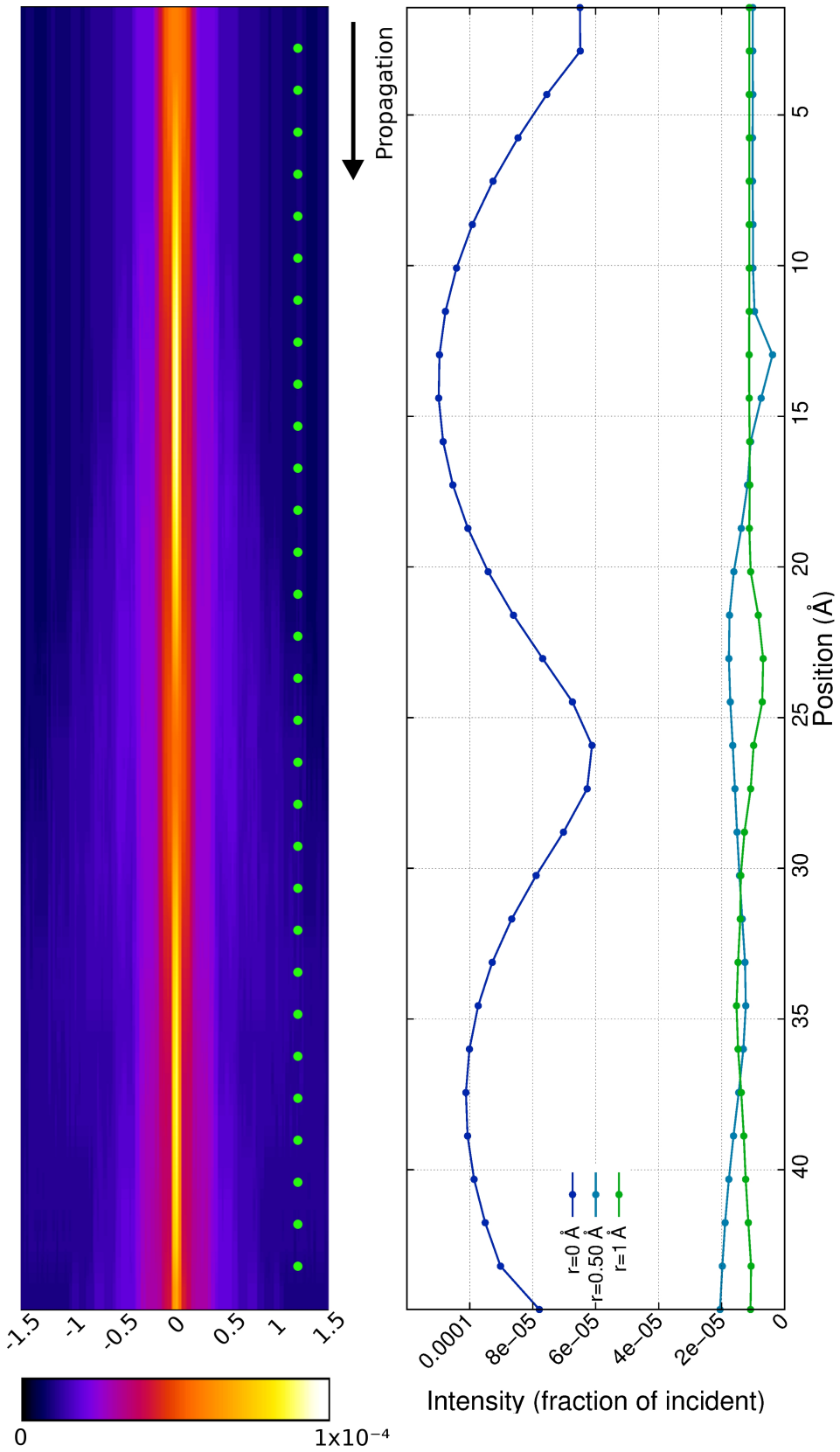

Figure 7.8: Beam propagation down a column of 30 Cu atoms in the same arrangement as the gold atoms in figure 7.7.

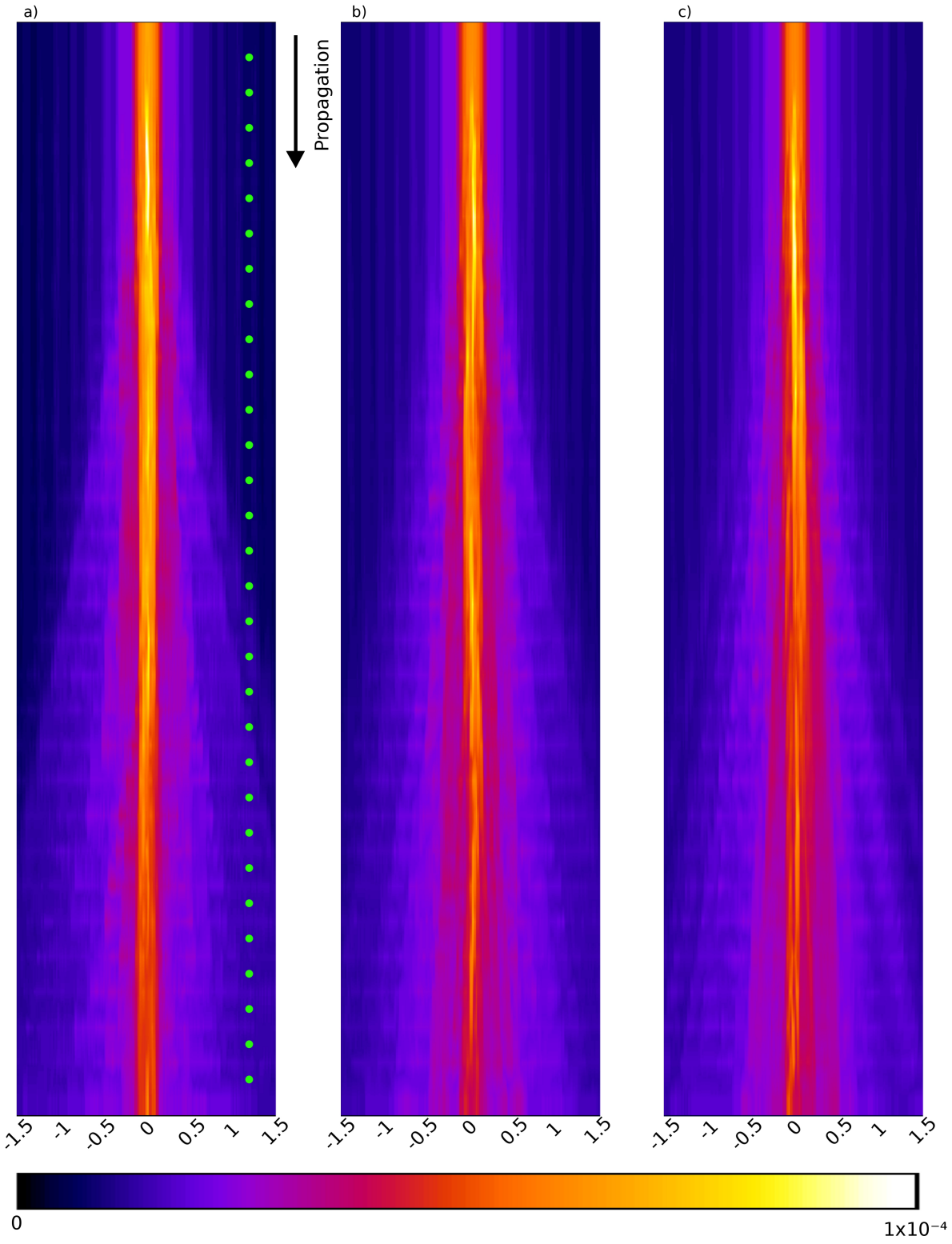

Figure 7.9: Beam propagation down a column of 30 Au atoms with MSDs of a), 0.027 Å^2, b), 0.066 Å^2, and c), 0.122 Å^2.

of 1.7% relative to the column with an MSD of 0.027 Å^2. The beam propagating down the column with a MSD of 0.122 Å^2, shown in figure 7.9, appears to have undergone more scattering than the other beams, however, much of the scattered beam remains relatively close to the beam axis. This means that it doesn't reach the high-angle detector and no longer passes sufficiently close to subsequent atoms to undergo an additional high-angle scattering interaction. Consequently, this MSD lies within the regime in which increased disorder reduces high-angle intensity, and indeed, the intensity at the 90-230 mrad HAADF detector has fallen by 6% relative to the perfectly aligned column and 20% relative to the column with a MSD of 0.066 Å^2.

The systematic studies described in this section confirm the existence of two different regimes with regards to the effects of the thermal motion of atoms on HAADF-STEM image intensities. To attribute the intensity of a feature to thermal motion, even qualitatively, it is necessary to know which regime is applicable to that particular case. It has been shown here that this depends on a non-trivial combination of the types of atoms involved, their thermal motion, and, by extension, their static geometrical arrangement. Determining the regime applicable to a specific image feature is thus most reliably achieved through the use of rigorous simulations. To give context to the relationship between peak intensity and MSDs, and to provide an approximate guide to the regimes in which certain systems fall, figure 7.2, which shows the MSD dependence of image intensities for 9-atom gold columns, has been reproduced with the addition of labels indicating the relevance of various MSD values. This is shown in figure 7.10.

This figure suggests that almost all gold samples are likely to fall within the low-disorder regime at room temperature, however, features with large static displacements due to strain and defects may fall within the high-disorder regime. Nonetheless, most bulk-like crystalline gold samples are expected to exhibit increasing intensities with small additional disordering.

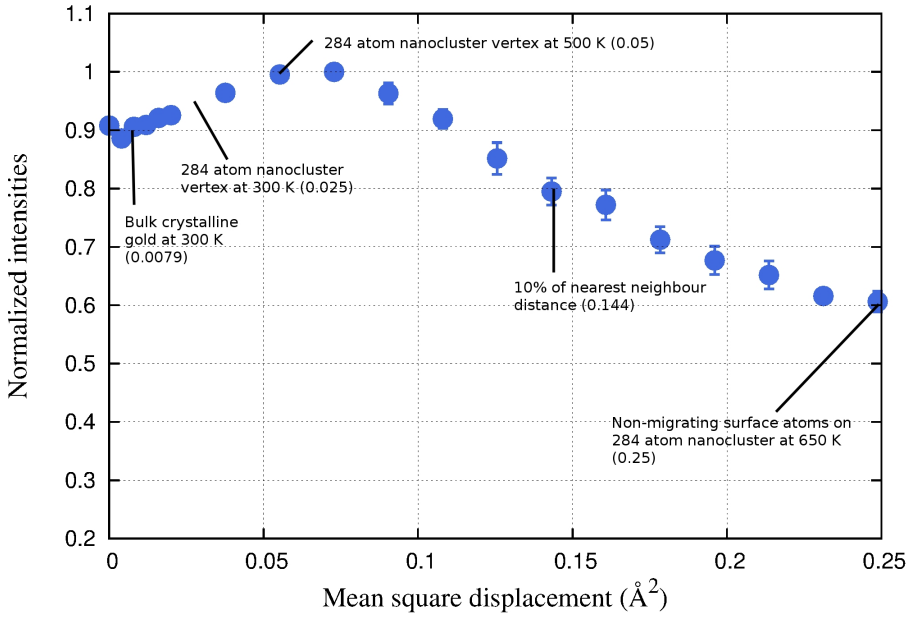

Figure 7.10: The variation of integrated peak intensities with MSD for 9 atom deep gold columns.

7.3 Nanocluster temperature series

In the previous section, the existence of two regimes, in which thermal motion either increases or reduces HAADF-STEM intensities, was demonstrated. This is expected to complicate the interpretation of image intensities, particularly in the case of inhomogeneous samples in which static displacements will also have an effect. To predict the manner in which thermal vibrations will effect HAADF-STEM images of the prototypical gold nanocluster used throughout this work, a number of multislice image simulations have been performed over a range of temperatures.

The simulations have been performed using frozen phonon configurations extracted from molecular dynamics simulations. The prototypical system is the same 284 atom Marks decahedral gold cluster described in chapter 6. Temperatures have been chosen ranging from approximately 77 K, which may be achievable experimentally with liquid nitrogen cooling, up to 900 K, beyond the solid to liquid phase transition. The ten temperatures selected are indicated by blue circles on the calorific curve in figure 7.11.

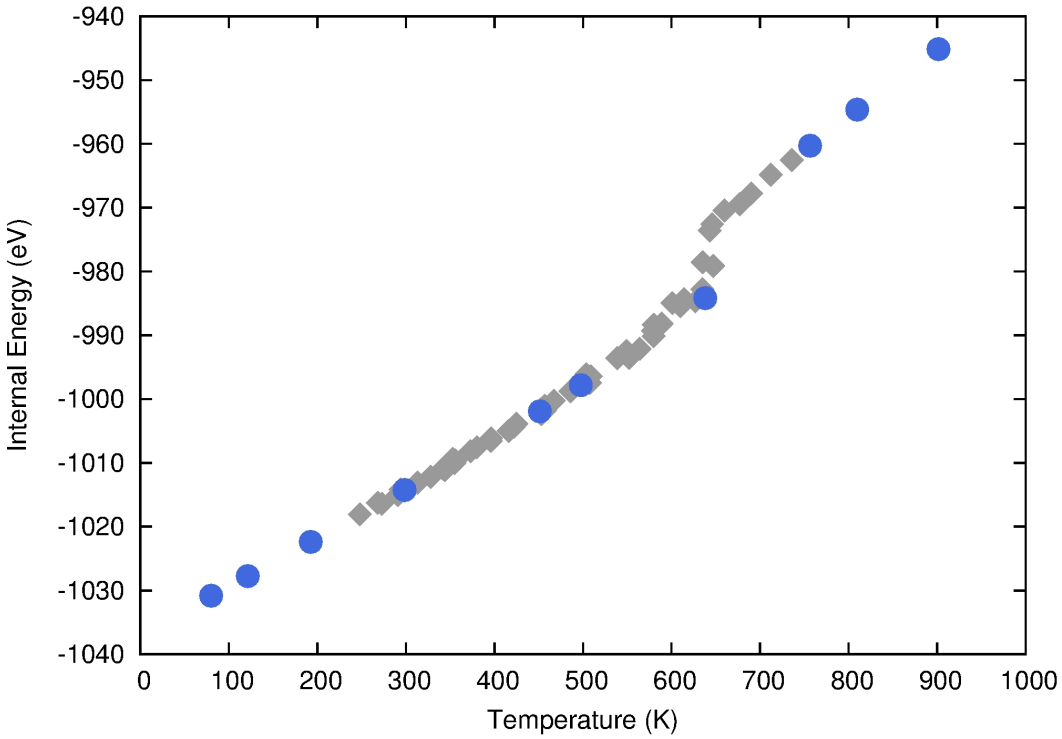

Figure 7.11: Calorific curve produced using molecular dynamics simulations. Image simulations have been performed using frozen phonon configurations from the molecular dynamics simulations indicated by the blue circles in the calorific curve. Both cryogenic and elevated temperatures are included.

Target temperature	Mean temperature
77 K	80 ± 2 K
125 K	121 ± 4 K
200 K	192 ± 7 K
300 K	299 ± 11 K
450 K	451 ± 17 K
500 K	498 ± 18 K
625 K	638 ± 27 K
750 K	757 ± 30 K
825 K	810 ± 31 K
900 K	902 ± 36 K

Table 7.2: Mean temperatures over the production period of molecular dynamics simulations.

Throughout this section the simulations will be referred to by their target temperatures, the mean temperatures over the course of the production runs for each of the simulations are given in table 7.2. At higher temperatures, particularly after melting, the temperatures become less stable due to the greater mobility of the atoms.

In the simulations at temperatures up to, and including 450 K, the cluster retains its multiply twinned crystalline structure with no reconstructions. The only structural effects of increasing the temperature within this range are a small amount of thermal expansion and an increase in the magnitude of thermal vibrations. During the 500 K simulation some surface reconstructions occur, an atom from the edge of one of the reentrant facets moves onto the adjacent (100) surface, resulting in the migration of two further atoms to occupy the vacancies, as shown in figure 7.12. This is the only reconstruction in this simulation.

At 625 K the surface reconstructions become more widespread but the cluster morphology is generally maintained. By 750 K, the melting phase transition has occurred, above this temperature the cluster is an amorphous liquid droplet. A typical structure is shown in figure 7.13.

The multislice simulations at each temperature were performed using the parameters in table 7.3. These are identical to those used in chapter 6, however, in this case a single defocus value is used, rather than the range of values previously used to account for current instabilities in the imaging system. Varying the defocus was found to have a negligible effect on Voronoi integrated intensities as it mainly caused a broadening of intensity peaks, rather than a change in scattering intensity. Including this effect increases simulation time by a factor of five whilst not substantially altering the quantitative data analysed here.

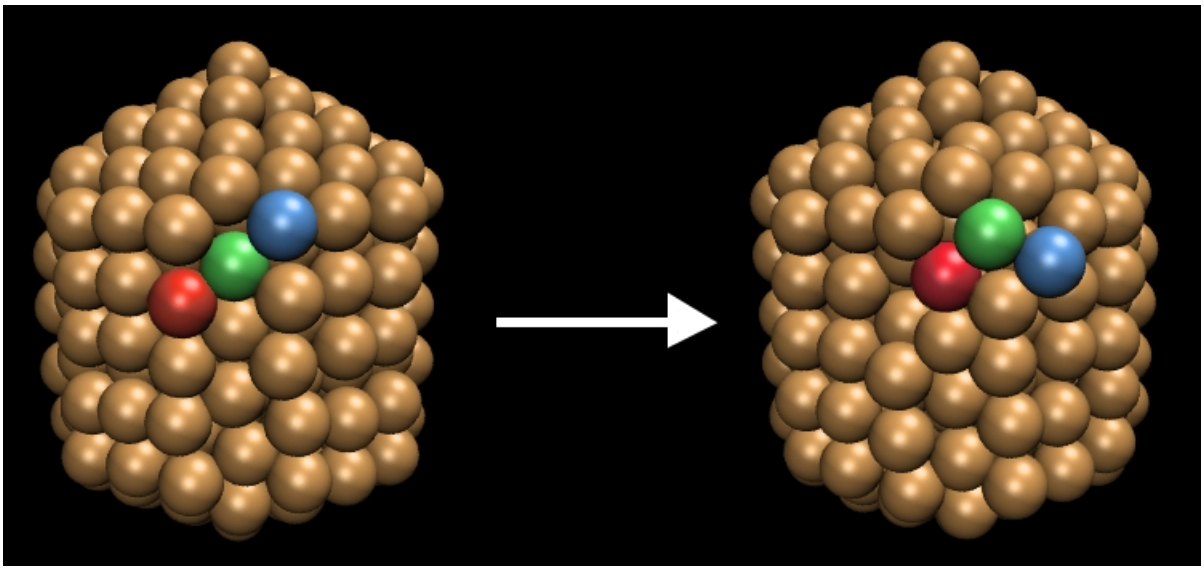

Figure 7.12: Surface reconstruction occuring at 500 K.

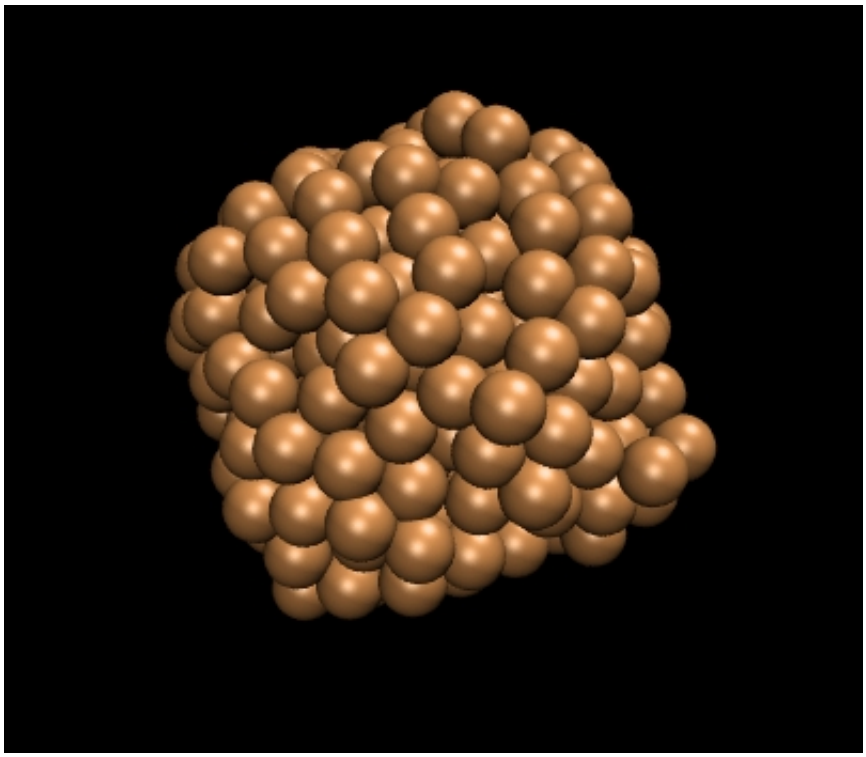

Figure 7.13: Typical cluster configuration during the molecular dynamics simulation at 750 K.

Accelerating voltage	200 keV
Defocus	-52 Å
Convergence angle	20 mrad
Detector range	90 230 mrad
Slice thickness	1.4 Å
Frozen phonon configurations	20
Transmission function size	4096^2 pixels / 77 Å
Probe function size	2048^2 pixels / 35.5 Å

Table 7.3: Multislice simulation parameters

The images resulting from the multislice simulations are shown in figure 7.14.

Qualitatively, a reduction in image intensities and a broadening of the intensity peaks can be seen as the temperature is increased from 77 K to 450 K. The surface reconstructions can be seen at 500 K and 625 K as they lead to a reduction in average intensities in the affected columns. At 750 K and above, there are no resolved column peaks, though some small peaks are present due to short-distance ordering in the droplets. The images of the melted clusters exhibit similar intensities and are smeared over a significantly larger area than the images of the crystalline structures. The image-integrated intensities from each of the simulations are plotted in figure 7.15.

The integrated intensities decay with temperature from their peak at 77 K. There is no region of increasing intensity with thermal motion. This suggests that the combination of static displacements and thermal motion, even at low temperatures, is sufficient to place the clusters in the high-disorder regime described in the previous section. For the melted clusters at 750 K and above, the image intensities have converged to a minima which can be attributed to a complete destruction of channelling effects.

To more closely analyse the images of the crystalline structures, the Voronoi cell peak quantification method has been applied. For the simulations at 500 K and 650 K, the intensities of peaks due to columns which underwent reconstructions have been removed from the dataset, as the number of atoms in these columns fluctuated over the course of the frozen phonon algorithm. The results of the quantitative analysis are shown in figure 7.16.

The droplet intensity indicated by the red line has been calculated by taking the mean of the integrated intensities over an area at the centre of the droplets in the simulations at 750 K, 825 K, and 900 K. These intensities were integrated over a square of the same

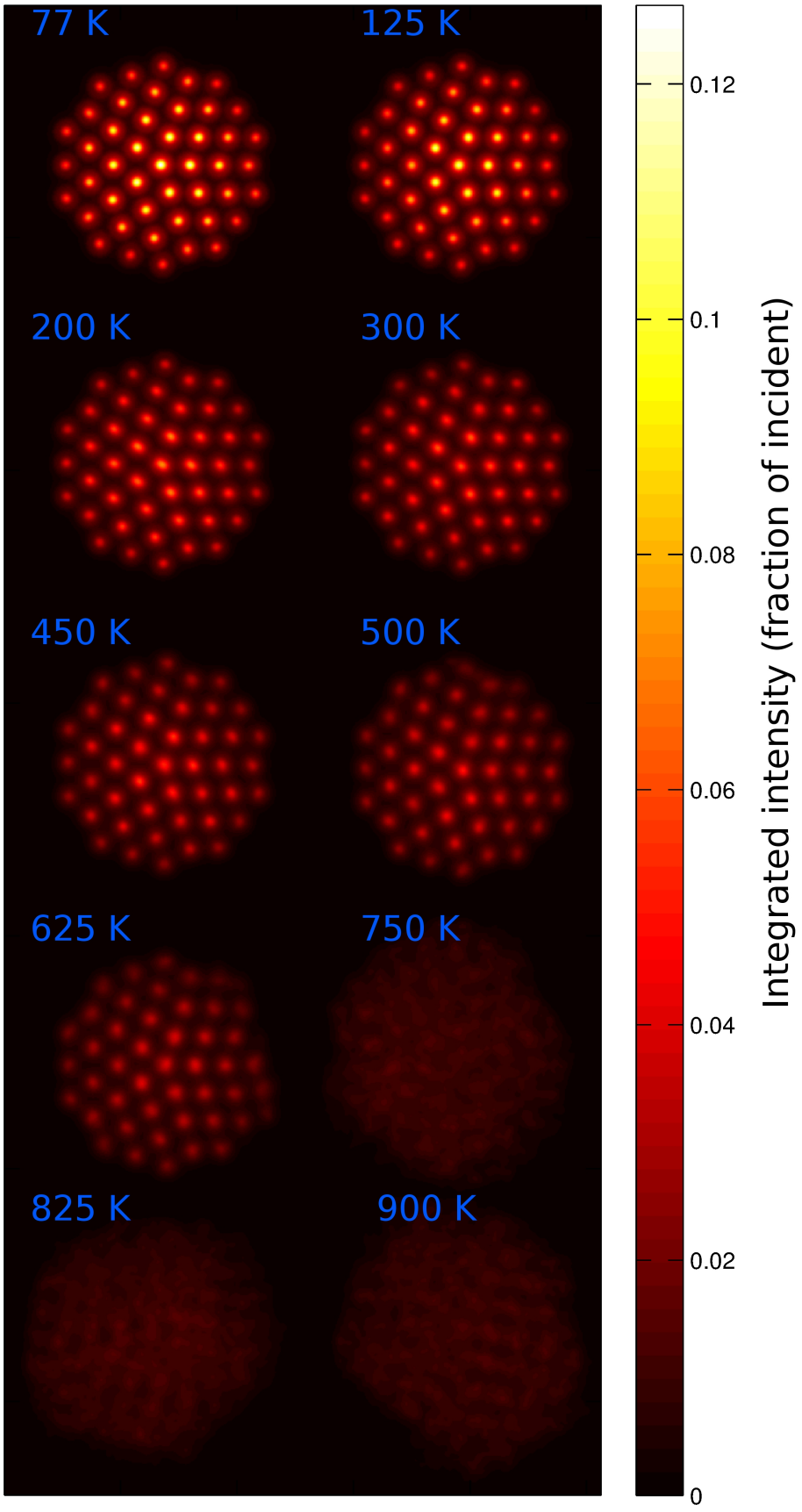

Figure 7.14: Image simulations of a 284 atom gold cluster at various temperatures, the melting point is approximately 650 K.

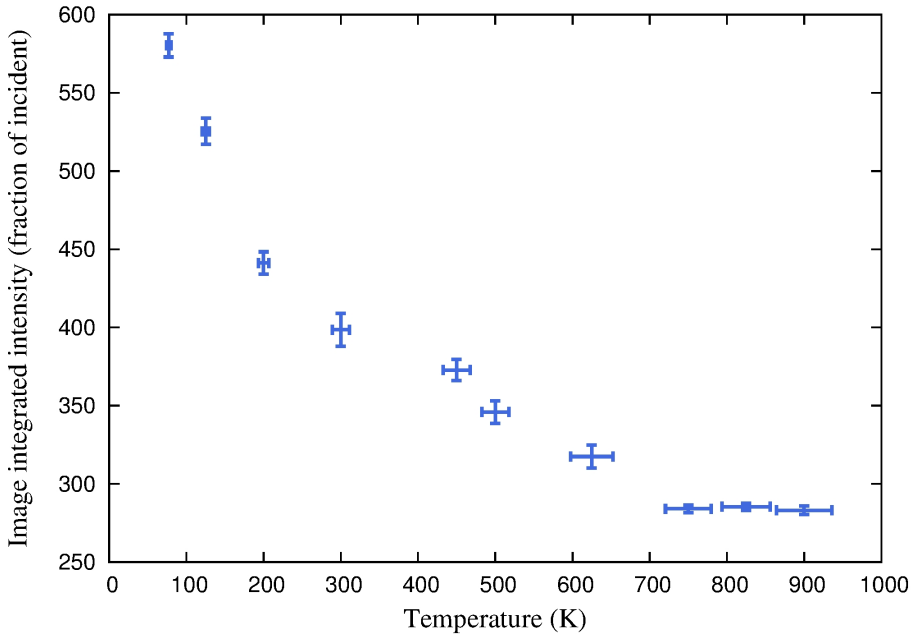

Figure 7.15: The image integrated intensities of the images in figure 7.14 decay exponentially with temperature reaching a minimum when the cluster melts.

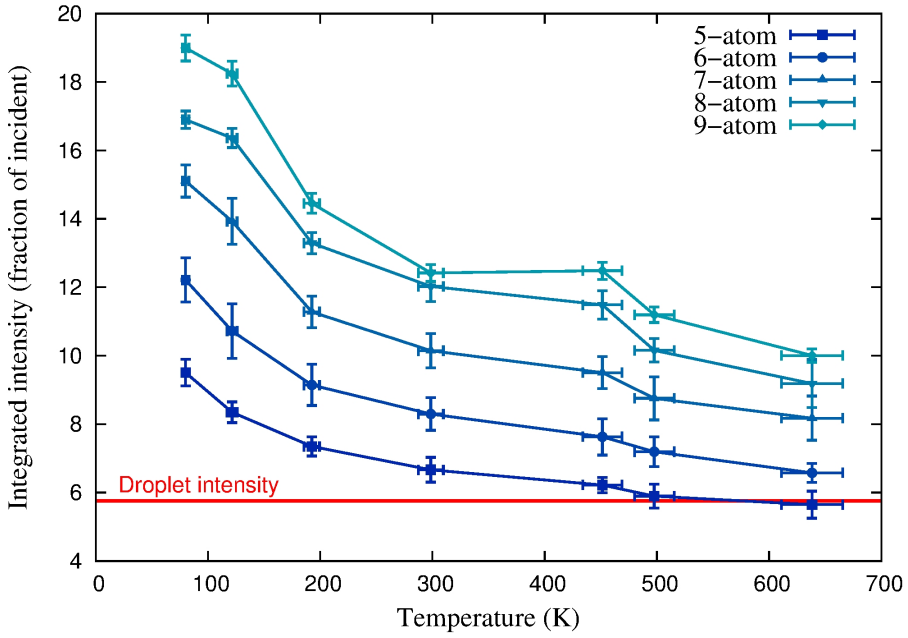

Figure 7.16: Mean intensities of columns of different lengths over a range of temperatures whilst the cluster remains crystalline.

area as the mean of the areas of the Voronoi cell surrounding the central 9 atom peak in the crystalline analyses. In almost all cases, the peak intensities diminish with increasing temperature. An exception occurs for the nine atom column which produces a greater integrated intensity peak at 450 K than at 300 K, though the increase is slight and a reduction is within the limits of the error in the frozen phonon convergence. The longer columns have greater absolute intensity reductions, which is consistent with the reduction being caused by attenuated channelling. The reduction of the intensities of the five atom columns to below that of the droplet intensity is a good indication that surface premelting is imminent. A melted surface would be expected to produce a slightly lower intensity than the droplet intensity indicated here, due to the difference in the thickness of the sample between the surface and centre. This result suggests that it may be possible to quantify the kinematics of surface atoms from the intensity of surface columns relative to the intensity of a liquid cluster. To achieve this experimentally would be challenging, as it requires calibrated detectors to measure absolute intensities and an image system which is robust to changes in sample temperature. However, it could offer insights into the mobility of surfaces on catalytically active clusters.

In the literature there have been several suggestions that the temperature sensitivity of HAADF-STEM imaging could be exploited to make thermal measurements. The simulations reported here suggest that this is possible in principle, but that the analysis is non-trivial and should be performed with reference to simulations. Furthermore, slight mistilts to the sample which are not readily observed via the shape of intensity peaks can dramatically reduce image intensities, and so would need to be precisely controlled in order to make accurate absolute thermal measurements. The best sample to observe the temperature effect would be one in which channelling effects are readily diminished. Lighter atoms with narrower scattering cross-sections are better in this sense, however, concessions would have to be made to achieve sufficiently good signal strength. In addition, the sample must be highly regular to minimise channelling attenuation by static displacements.

7.4 Chapter conclusion

In this chapter, the effects of thermal motion on HAADF-STEM image intensities have been investigated in a systematic manner. Two regimes of differing behaviour have been

identified for crystalline structures. In the low disorder regime, small additional disordering due to thermal motion or small static displacements results in an increase in HAADF-STEM image intensities by increasing the area of the scattering cross-section. In the high-disorder regime, additional disordering results in a reduction HAADF-STEM image intensities by reducing the extent to which the electron beam channels down atomic columns and is focused onto subsequent atoms. Evidence for these mechanisms has been found in the form of a correlation between the high-angle signal and focussed peaks of electron channelling during beam propagation, which diminishes with increased disorder. This explanation for the effects of thermal motion on image intensity reconciles the apparently contradictory observations reported in the literature. A combination of molecular dynamics and multislice simulations have been used to produce a series of HAADF-STEM images of a 284 atom gold nanocluster at a variety of temperatures ranging from 80 K to 902 K. The results show decreasing intensities with thermal motion, even between the low temperatures of 80 and 121 K. The increasing thermal motion of the atoms at these temperatures would be expected to cause an increase in intensities, however, the distorted crystalline structure of the small nanocluster introduces sufficient disorder to place the system in the high-disorder regime.

Chapter 8

Conclusion

The primary aim of this work was to determine a means by which to account for the inhomogeneous thermal motion expected for gold nanoclusters in multislice HAADF-STEM simulations. This was specifically prompted by qualitative differences between the intensities of experimental and simulated images at the surface of clusters [35], and more broadly by a tendency of reports in the literature to attribute unexpected image intensities to discrepant thermal motion, without further investigation [36, 105]. The magnitude of thermal vibrations is expected to be greater for low coordination atoms, and x-ray diffraction experiments of bulk agglomerates of nanoclusters provide experimental evidence for this. Unfortunately, there is currently no method capable of experimentally measuring the mean square displacements of individual atoms. To resolve this problem, a new method has been developed in which structure coordinates are extracted at intervals from molecular dynamics simulations for use as frozen phonon configurations in multislice HAADF-STEM simulations. The results of these simulations proved to be qualitatively, as well as quantitatively, different from those produced using more conventional approaches, confirming the value of the new method. In this work, the method was applied to prototypical gold nanocluster subjects, however, it is equally applicable to any structure for which molecular dynamics simulations can be accurately performed.

The rigorous HAADF-STEM simulation method produced a more realistic image than any yet reported, and, as the structure used in the simulation was known, provided a means by which to test image analysis procedures reported in the literature. The statistical parameter estimation technique introduced by Van Aert *et al.* [37], was applied to the image. This proved highly effective in identifying intensity peaks produced by unique column types, however, the trivial atom assignment method which has been used in the literature

introduced a miscount. This was caused by the assignment of additional atom counts to both peaks in bimodal intensity distributions. It is thus concluded that, whilst the statistical parameter technique is useful in producing initial model structures, rigorous image simulations should be employed to check and optimize the models due to the non-trivial image formation process.

In applying the new methodology, it was found that the inclusion of enhanced thermal vibrations relative to conventional simulations, caused a reduction in image intensities. This contradicts several reports in the literature [47, 7], though other reports were found to concur with this observation [106, 101, 19]. To further investigate this issue a number of systematic investigations were performed. These yielded two different regimes determining the effects of thermal motion on image intensities. For low-disorder systems, an increase in thermal motion was found to cause an increase in image intensities, which is attributed to an increasing projected scattering cross-section. For high-disorder systems, increasing thermal motion was found to cause a reduction in intensities, which is attributed to a diminishing of electron channelling and the associated beam focussing that increases the intensity of the beam on the atomic column. Recording the propagation of electron beams down different columns that fall into the differing regimes provides support for this mechanism as an explanation for the observed behaviour. These observations reconcile the apparently contradictory reports in the literature, but add complication to the interpretation of HAADF-STEM images, particularly as the level of disorder at which the two regimes occur has been found to vary with atom type. To determine which regime applied to the prototypical gold nanocluster at various temperatures, a series of image simulations were performed using frozen phonon configurations from molecular dynamics simulations at a range of temperatures between approximately 77 K and 900 K. In all cases below the melting point at approximately 650 K, an increase in temperature caused a reduction in image intensities. This indicates that the strain-induced static disorder of the small cluster is sufficient to place it in the high-disorder regime. The intensities reached a minimum after the melting point, retaining similar integrated intensities between 750 K and 900 K, due to a complete destruction of electron channelling effects.

To facilitate the completion of simulations in this work in a reasonable time, a multislice program has been written which exploits the highly-parallel architecture of graphical processing units. The pace of development in graphical processing hardware is expected to provide increasing benefits for the use of this program in the future.

In future work, experimental confirmations of the effects observed here are desirable. Crystalline gold samples are likely to be a poor structure in which to observe this effect as they have large scattering factors, and so channelling effects are less readily diminished by thermal vibrations than would be expected for lighter atoms. Simulations using copper scattering factors, whilst physically unrealistic, suggest that channelling effects are too readily diminished so that the regime in which additional thermal motion increases image intensity is rather narrow. An intermediate element with highly-regular crystalline structure with minimal defects should make an ideal candidate.

The methods developed in this work should allow for the structural characterization of gold nanoclusters through quantitative comparisons between experimental and simulated images, with iterative model optimization to achieve a good match. This requires experimental images produced after calibration of the microscope detector, as suggested by LeBeau *et al.* [36].

Appendix A

1 The Debye-Waller factor

The magnitude of thermal motion is often described in terms of the Debye-Waller factor (DWF), a term used in x-ray or neutron scattering to describe the probability of coherent scattering. It is described by equation A.1 in which the angled brackets represent a thermal or time average.

$$\mathrm{DWF} = \left\langle e^{i\mathbf{q}\cdot\mathbf{u}} \right\rangle^2 \tag{A.1}$$

Here, $\mathbf{q}$ is the scattering factor and $\mathbf{u}$ is the displacement of the scattering centre from its equilibrium position. This term incorporates any effects which reduce coherence, including thermal motion but also static displacements from the regular crystalline structure. In the literature the DWF has also been used interchangeably with the B-factor [36]. However, strictly, the B-factor is related to the DWF as in equation A.2 [72].

$$DWF = e^{\left(-B\frac{g^2}{16\pi^2}\right)} \tag{A.2}$$

For the gth structure factor of a perfect crystal. The B-factor is related to the Mean Square Displacement (MSD) by equation A.3

$$B = 8\pi^2 \left\langle \mathrm{MSD}^2 \right\rangle \tag{A.3}$$

The nomenclature is further confused as the commonly used MSD is, in fact, defined as “the mean square of the x component of the thermal displacement of an atom from its equilibrium position”. To avoid any ambiguity this definition of the mean square displacement is used throughout this work.

2 CuSTEM.cu

A multislice scanning transmission electron microscopy simulation program, written in CUDA to exploit the parallelism of graphical processing unit architecture.

```
/*
////////////////////////////////
/////////|CuSTEM.cu|////////////
////////////////////////////////

Multislice high-angle annular dark field scanning transmission
election microscopy simulator based on the autostem program in
Kirklands TEMSIM package (Copyright 1998-2011 Earl J. Kirkland).

Copyright (C) 2013-2014 Richard Aveyard

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 3 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, see <http://www.gnu.org/licenses/>.

------------------------------ NO WARRANTY ------------------------------
THIS PROGRAM IS PROVIDED AS-IS WITH ABSOLUTELY NO WARRANTY
OR GUARANTEE OF ANY KIND, EITHER EXPRESSED OR IMPLIED,
INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
IN NO EVENT SHALL THE AUTHOR BE LIABLE
FOR DAMAGES RESULTING FROM THE USE OR INABILITY TO USE THIS
PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA
BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR
THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH
ANY OTHER PROGRAM).

-------------------------------------------------------------------------

22/04/2013 Richard Aveyard (ra514@york.ac.uk)

Compile with:
g++ tiffsubs.c -c
nvcc -Xcompiler="-fopenmp" -o CuSTEM CuSTEM.cu slicelib.cu tiffsubs.o↩
     -lm -lcufft -lgomp -arch sm_13 -DGPUSHMEM=130 -g
g++ imgavg.C -o imgavg


To make parallel calls to the GPU the cuda compute mode must
```

```
48 be set using the command:
49 sudo nvidia-smi -c EXCLUSIVE_PROCESS
50
51 */
52
53 #include <stdio.h>  /* ANSI C libraries used */
54 #include <stdlib.h>
55 #include <string.h>
56 #include <math.h>
57 #include <time.h>
58 #include <cuda.h> /* Header for GPU handling*/
59 #include <cuda_runtime.h>
60 #include <cufft.h>
61 #include <omp.h>
62 #include <iostream>
63 #include "slicelib.h"    /* misc. routines for multislice */
64 #include "tiffsubs.h"    /* file I/O routines in TIFF format */
65 #define USE_OPENMP       /* define to use openMP */
66 #ifdef USE_OPENMP
67 #define walltim() ( omp_get_wtime() )
68 double walltimer;
69 #endif
70 #define BW (2.0F/3.0F)  /* antialiasing bandwidth limit factor */
71 #define ABERR 1.0e-5     /* max error for a,b */
72 #define NCMAX   512 /* max number of characters per line */
73 #define NPARAM  64  /* number of parameters */
74 #define ADF      0  /* modes of collector */
75 #define CONFOCAL 1
76 #define TRUE     1
77 #define FALSE    0
78 #define NZMAX    103     /* max atomic number Z */
79 #define NRMAX    100     /* number of in look-up-table in vzatomLUT */
80 #define RMIN     0.01    /* r (in Ang) range of LUT for vzatomLUT() */
81 #define RMAX     5.0
82
83 cufftDoubleComplex *trans, *dtrans, *dtranswork ;/*Host CPU arrays*/
84 cufftDoubleComplex *probe, *dprobe, *dtemp;
85
86 cufftHandle  planT;   /*  transmission function FFT plans */
87 cufftHandle  planP;   /*  probe function FFT plans */
88
89 float *propxr, *propxi;
90 float *propyr, *propyi;
91 float *dpropxr, *dpropxi;
92 float *dpropyr, *dpropyi;
93 float *dkxp2, *dkyp2, *dkx2, *dky2, *dk2max;
94 float zmin, zmax;
95 int nx, ny, nxprobe, nyprobe, nslice, *dtranspara, *transpara;
96 float *kx, *ky, *kx2, *ky2, *kxp, *kyp, *kxp2, *kyp2;
97 float  **rmin, **rmax, *xp, *yp;
98 float *xa, *ya, *za, *occ, *wobble;
99 float *xa2, *ya2, *za2, *occ2;
100 int natom, *Znum, *Znum2, l, is;
101 double wavlen, k2maxp, Cs3,Cs5, df,apert1, apert2, pi, keV;
102 float ax, by, cz;
103 float dfa2, dfa2phi, dfa3, dfa3phi;
104 double *almin, *almax, *k2max, *k2min, deltaz;
```

```
long nbeamt;

/*Functions are defined at end of this file*/
double periodic( double pos, double size );

void STEMsignals( double x[], double y[], int npos, double ***detect,
 int ndetect, double ThickSave[], int nThick, double sum[],
char fileotpre[], int svbeam, int nR, int PixP[], int ixp,
int iwobble );

void trlayer( const float x[], const float y[], const float occ[],
    const int Znum[], const int natom,
    const float ax, const float by, const float kev,
    cufftDoubleComplex *trans, const long nx, const long ny,
    double *phirms, long *nbeams, const float k2max);

/*CUDA Functions*/
__global__ void cudalayer(cufftDoubleComplex *dtranswork, float *dkx2
,float *dky2, float *dk2max );

__global__ void cudatrans(cufftDoubleComplex *dtranswork,
cufftDoubleComplex *dprobe, cufftDoubleComplex *dtemp,
int *dtranspara);

__global__ void cudaprop(cufftDoubleComplex *dprobe,
cufftDoubleComplex *dtemp, float *dkxp2, float *dkyp2,
 float *dpropxr, float *dpropxi, float *dpropyr,
 float *dpropyi, float *dk2max);

/* spline interpolation coeff. */
int splineInit=0, *nspline;
double  *splinx, **spliny, **splinb, **splinc, **splind;
/* extra globals for confocal mode */
int doConfocal;
int *collectorMode, *PixP;
float dfa2C, dfa2phiC, dfa3C, dfa3phiC;/*astigmatism parameters*/
double *collectMin, *collectMax;
double Cs3C, Cs5C, dfC, apert1C, apert2C;/*aberrations of collector lens*/

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
int main()
{
    char filein[NCMAX], fileout[NCMAX], fileoutpre[NCMAX], beamout[NCMAX];
    char description[NCMAX], cmode;
    const char version[] = "15-March-2013";
    int ix, iy, i, idetect, nout, nxout, nyout,
        ncellx, ncelly, ncellz, iwobble, nwobble,
        ndetect, nprobes, ip, nThick, it, ns, svbeam, nR;
    int  l1d, lwobble, lxzimage, Pb, BO, iR, ixp, trkpx, trkpy;
    long nbeamp, nbeampo;
```

```
156     long  ltime;
157     unsigned long iseed;
158     float *param, ***pixr, **pixout, temp, pmin, pmax;
159     float wmin, wmax, xmin,xmax, ymin, ymax, temperature;
160     double scale, *x, *y, sum, *sums, w, ***detect, ***detect2,
161        tctx, tcty, xi,xf, yi,yf, dx, dy, totmin, totmax,
162        ctiltx, ctilty, timer, sourcesize, sourceFWHM, *ThickSave,
163        vz, rsq, trkx, trky, k2,k2maxa,k2maxb;
164     FILE *fp;
165 //Openmp settings, uncomment to use!!
166 //int nthreads = 3;
167 //omp_set_num_threads(nthreads);
168 //------------------------------------------------------------------//
169
170 /* start by announcing version etc */
171 printf("CuSTEM version dated %s\n", version );
172 printf("This program is provided AS-IS with ABSOLUTELY NO WARRANTY\n"
173             " under the GNU general public license\n\n" );
174
175     printf( "Calculate STEM images using GPUs\n");
176 #ifdef USE_OPENMP
177     printf( "and multithreaded using openMP\n");
178 #endif
179     printf( "\n" );
180
181 /*----------------  get simulation options ----------------- */
182
183     pi = 4.0 * atan( 1.0 );
184
185     printf("Name of file with input atomic "
186            "potential in x,y,z format:\n");
187     ns = scanf("%500s", filein );
188    strcpy (beamout, fileoutpre);
189    strcat (beamout, "beam");
190
191     printf("Replicate unit cell by NCELLX,NCELLY,NCELLZ :\n");
192     ns = scanf("%d %d %d", &ncellx, &ncelly, &ncellz);
193     if( ncellx < 1 ) ncellx = 1;
194     if( ncelly < 1 ) ncelly = 1;
195     if( ncellz < 1 ) ncellz = 1;
196
197     printf("STEM probe parameters, V0(kv), Cs3(mm), Cs5(mm),"
198            " df(Angstroms), apert1,2(mrad) :\n");
199     ns = scanf("%lg %lg %lg %lg %lg %lg",
200           &keV, &Cs3, &Cs5, &df, &apert1, &apert2);
201 l=0;/*Winding number for electron vortices set to 0 for normal ↩
    operation (ra514)*/
202     printf( "Magnitude and angle of 2-fold astigmatism"
203         " (in Ang. and degrees):\n");
204     ns = scanf( "%f %f", &dfa2, &dfa2phi);
205     dfa2phi = (float) (dfa2phi * pi /180.0F);
206
207     printf( "Magnitude and angle of 3-fold astigmatism"
208         " (in Ang. and degrees):\n");
209     ns = scanf( "%f %f", &dfa3, &dfa3phi);
210     dfa3phi = (float) (dfa3phi * pi /180.0F);
211
```

```
212     wavlen = wavelength( keV );
213     printf("wavelength = %f Angstroms\n", wavlen);
214     if( apert1 > apert2 ) {
215        printf("Bad probe aperture specification.\n");
216        printf("apert1 must be less than apert2.\n");
217        printf("apert1=%f, apert2 = %f\n", apert1, apert2);
218        exit( 0 );
219     }
220
221     printf("Size of specimen transmission function"
222           " Nx,Ny in pixels : \n");
223     ns = scanf("%d %d", &nx, &ny);
224
225     printf("Size of probe wave function"
226           " Nx,Ny in pixels : \n");
227     ns = scanf("%d %d", &nxprobe, &nyprobe);
228
229     printf("Crystal tilt x,y in mrad. :\n");
230     ns = scanf( "%lf %lf", &ctiltx, &ctilty );
231     ctiltx = ctiltx * 0.001;
232     ctilty = ctilty * 0.001;
233
234     l1d = askYN("Do you want to calculate a 1D line scan");
235
236     if( l1d == 1 ) {
237         lxzimage = askYN("Do you want to save all depth information ↩
                as xz image");
238         nThick = 1;
239     } else {
240         do { printf("Number of thickness levels to save, including"
241                 " the end(>=1):\n");
242              ns = scanf( "%d", &nThick );
243         } while (nThick <= 0);
244         ThickSave = (double*) malloc1D( nThick, sizeof(double), "↩
                ThickSave");
245         if( nThick > 1 ) {
246            printf( "type thickness (in Ang.) of %d intermediate ↩
                   layers"
247                 " :\n", (nThick-1) );
248            for( it=0; it<(nThick-1); it++) ns = scanf( "%lf", &↩
                   ThickSave[it] );
249         }
250     }
251
252     printf("File name prefix to get output of STEM multislice result ↩
        "
253         "(no extension):\n");
254     ns = scanf("%500s", fileoutpre);
255
256
257     do {    printf("Number of detector geometries (>=1):\n");
258             ns = scanf( "%d", &ndetect );
259     } while (ndetect <= 0);
260
261     almin  = (double*)  malloc1D( ndetect, sizeof(double), "almin" );
262     almax  = (double*)  malloc1D( ndetect, sizeof(double), "almax" );
263     collectorMode = (int*) malloc1D( ndetect, sizeof(int), "↩
```

```
        collectorMode" );
264
265     doConfocal = FALSE;
266
267     for( idetect=0; idetect<ndetect; idetect++) {
268         printf("Detector %3d, type: min max angles(mrad)"
269                 " or radius(Ang.) \n followed by m or A\n", idetect↩
                    +1);
270         ns = scanf("%lg %lg %c",
271                 &almin[idetect], &almax[idetect], &cmode );
272         if( (cmode == 'm') || (cmode=='M') ) {
273                 collectorMode[idetect] = ADF;
274                 printf( "normal ADF detector\n");
275         } else if( (cmode == 'a') || (cmode=='A') ) {
276                 collectorMode[idetect] = CONFOCAL;
277                 printf( "confocal detector\n");
278                 doConfocal = TRUE;
279         } else {
280                 printf( "unrecognized collector mode = %c\n", cmode);
281                 exit( 0 );
282         }
283     }
284
285     if( doConfocal == TRUE ) {
286         printf("Collector lens parameters, Cs3(mm), Cs5(mm),"
287                  " df(Angstroms), apert1,2(mrad) :\n");
288         ns = scanf("%lg %lg %lg %lg %lg",
289                          &Cs3C, &Cs5C, &dfC, &apert1C, &apert2C);
290         printf( "Magnitude and angle of 2-fold astigmatism"
291                          " (in Ang. and degrees):\n");
292         ns = scanf( "%f %f", &dfa2C, &dfa2phiC);
293         dfa2phiC = (float) (dfa2phi * pi /180.0F);
294         printf( "Magnitude and angle of 3-fold astigmatism"
295                 " (in Ang. and degrees):\n");
296         ns = scanf( "%f %f", &dfa3C, &dfa3phiC);
297         dfa3phiC = (float) (dfa3phiC * pi /180.0F);
298
299         if( apert1C > apert2C ) {
300                 printf("Bad collector aperture specification.\n");
301                 printf("apert1 must be less than apert2.\n");
302                 printf("apert1=%f, apert2 = %f\n", apert1C, apert2C);
303                 exit( 0 );
304         }
305     }
306
307     if( l1d == 1 ) {
308         printf("xi, xf, yi, yf, nout :\n");
309         ns = scanf("%lg %lg %lg %lg %d", &xi, &xf, &yi, &yf, &nout);
310         nprobes = nout;
311     }else {
312         printf("xi,xf,yi,yf, nxout,nyout :\n");
313         ns = scanf("%lg %lg %lg %lg %d %d",
314              &xi, &xf, &yi, &yf, &nxout, &nyout);
315         nprobes = nyout;
316     }
317
318 //..... inputs for recording beam evolution .........
```

```
319    svbeam = askYN("Do you want to save exit waves?\n");
320                 printf("Number of incident beams to track\n");
321               ns = scanf( "%d", &nR );
322         PixP = (int*) malloc1D( nR, sizeof(int), "PixP");
323
324            printf( "type position of %d probes to track:\n", (nR) );
325            for( iR=0; iR<nR; iR++){
326                 ns=scanf("%lf %lf", &trkx, &trky);
327         trkpx=  floor( trkx/ ((xf-xi)/nxout)+0.5f );
328         trkpy=  floor( trky/ ((yf-yi)/nyout)+0.5f );
329    PixP[iR]= trkpx*nyout + trkpy;
330    printf("xf=%lf, xi=%lf, yf=%lf, yi=%lf, nxout=%i, nyout=%i\n", xf,↩
        xi,yf,yi,nxout,nyout);
331    printf("beam%i, trkx= %lf, trky= %lf, trkpx=%i, trkpy=%i, PixP[iR↩
        ]=%i\n",iR,trkx,trky,trkpx,trkpy,PixP[iR]);
332                 }
333 //..... inputs for recording beam evolution.........
334
335     printf("Slice thickness (in Angstroms):\n");
336     ns = scanf("%lf", &deltaz );
337     if( deltaz < 1.0 ) {
338         printf("WARNING: this slice thickness is probably too thin"
339             " for autostem to work properly.\n");
340     }
341
342     lwobble = askYN("Do you want to include thermal vibrations");
343     if( lwobble == 1 ) {
344         printf( "Type the temperature in degrees K:\n");
345         ns = scanf( "%g", &temperature );
346         printf( "Type number of configurations to average over:\n");
347         ns = scanf( "%d", &nwobble );
348         if( nwobble < 1 ) nwobble = 1;
349         ltime = (long) time( NULL );
350         iseed = (unsigned) ltime;
351         if( ltime == -1 ) {
352             printf("Type initial seed for random number generator:\n"↩
                );
353             ns = scanf("%ld", &iseed);
354         } else {
355             printf( "Random number seed initialized to %ld\n", iseed ↩
                );
356         }
357         printf( "Type source size (FWHM in Ang.):\n" );
358         ns = scanf( "%lf", &sourceFWHM );
359     } else {
360         temperature = 0.0F;
361         nwobble = 1;
362         sourceFWHM = 0.0;
363     }
364     /* convert FWHM to standard deviation
365             by dividing by 2*sqrt(2*ln(2)) */
366     sourcesize = sourceFWHM / 2.354820045;
367
368     timer = cputim();    /* get initial CPU time */
369 #ifdef USE_OPENMP
370    walltimer = walltim();  /* wall time for opneMP */
371 #endif
```

```
372
373     param = (float*) malloc1D( NPARAM, sizeof(float), "param" );
374     for( i=0; i<NPARAM; i++) param[i] = 0.0F;
375
376     /*  calculate relativistic factor and electron wavelength */
377     wavlen = (float) wavelength( keV );
378     printf("electron wavelength = %g Angstroms\n", wavlen);
379
380     /*-----read in specimen coordinates and scattering factors-------*/
381
382     natom = ReadXYZcoord( filein, ncellx, ncelly, ncellz,
383         &ax, &by, &cz, &Znum, &xa, &ya, &za, &occ, &wobble,
384         description, NCMAX );
385
386     printf("%d atomic coordinates read in\n", natom );
387     printf("%s", description );
388
389     printf("Lattice constant a,b,c = %12.4f, %12.4f, %12.4f\n", ax,by↩
        ,cz);
390
391 #ifdef USE_OPENMP
392     /*  force LUT init. to avoid redundant init in parallel form */
393     rsq = 0.5;  /* arbitrary position */
394     for( i=0; i<natom; i++) vz =  vzatomLUT( Znum[i], rsq );
395 #endif
396
397    /*calculate thickness levels to save (1D mode)
398     or check range (2D mode)*/
399     if( l1d == 1 ) {
400         if( lxzimage == 1 ) {
401             /* save all thickness levels  */
402             nThick = (int) ( cz/deltaz + 0.5 );
403             ThickSave = (double*) malloc1D( nThick, sizeof(double), "↩
                ThickSave");
404             for( it=0; it<nThick; it++) {
405                 ThickSave[it] = deltaz*(it+1);
406             }
407         } else {
408             nThick = 1;
409             ThickSave = (double*) malloc1D( nThick, sizeof(double), "↩
                ThickSave");
410             ThickSave[0] = cz;
411         }
412         printf( "save up to %d thickness levels\n", nThick );  /* ↩
            diagnostic */
413     } else {
414         ThickSave[nThick-1] = cz;  /*  always save the last level */
415         for( it=0; it<(nThick-1); it++)
416         if( (ThickSave[it] < 0.0) || (ThickSave[it] > cz) ) {
417             printf("Bad thickness level = %g A, allowed range= "
418                 "0.0 to %f A\n", ThickSave[it], cz );
419             exit( 0 );
420         }
421     }  /* end if( l1d == ... */
422
423     if( lwobble == 0 ) {
424         printf( "Sorting atoms by depth...\n");
```

```
425         sortByZ( xa, ya, za, occ, Znum, natom );
426     }
427     /* to add random offsets */
428     xa2 = (float*) malloc1D( natom, sizeof(float), "xa2" );
429     ya2 = (float*) malloc1D( natom, sizeof(float), "ya2" );
430     za2 = (float*) malloc1D( natom, sizeof(float), "za2" );
431     Znum2 = (int*) malloc1D( natom, sizeof(int), "Znum2" );
432     occ2 = (float*) malloc1D( natom, sizeof(float), "occ2" );
433
434     /*  calculate the total specimen volume and echo */
435     xmin = xmax = xa[0];
436     ymin = ymax = ya[0];
437     zmin = zmax = za[0];
438     wmin = wmax = wobble[0];
439
440     for( i=0; i<natom; i++) {
441         if( xa[i] < xmin ) xmin = xa[i];
442         if( xa[i] > xmax ) xmax = xa[i];
443         if( ya[i] < ymin ) ymin = ya[i];
444         if( ya[i] > ymax ) ymax = ya[i];
445         if( za[i] < zmin ) zmin = za[i];
446         if( za[i] > zmax ) zmax = za[i];
447         if( wobble[i] < wmin ) wmin = wobble[i];
448         if( wobble[i] > wmax ) wmax = wobble[i];
449     }
450     printf("Total specimen range is\n %g to %g in x\n"
451            " %g to %g in y\n %g to %g in z\n", xmin, xmax,
452            ymin, ymax, zmin, zmax );
453     if( lwobble == 1 )
454    printf("Range of thermal rms displacements (300K)= %g to %g\n",
455             wmin, wmax );
456     /*  check for valid scan coordinates  */
457
458     if( (xi < 0.0) || (xi > ax) ||
459         (xf < 0.0) || (xf > ax) ||
460         (yi < 0.0) || (yi > by) ||
461         (yf < 0.0) || (yf > by) ) {
462 printf("WARNING: Coordinates out of range will be made periodic.\n");
463             printf("xi,xf,yi,yf= %f, %f, %f, %f\n", xi, xf, yi, yf );
464     }
465
466     /*  check that requested probe size is not bigger
467         than transmission function size (or too small)
468     */
469     if( (nxprobe > nx) || (nxprobe < 2) ) {
470         nxprobe = nx;
471         printf("Probe size reset to nx = %d\n", nxprobe);
472     }
473
474     if( (nyprobe > ny) || (nyprobe < 2) ) {
475         nyprobe = ny;
476         printf("probe size reset to ny = %d\n", nyprobe);
477     }
478
479     /*  calculate spatial frequencies for future use
480         (one set for transmission function and one for probe
481         wavefunction)
```

```
482     NOTE: zero freg is in the bottom left corner and
483         expands into all other corners - not in the center
484         this is required for FFT  */
485
486     kx  = (float*) malloc1D( nx, sizeof(float), "kx" );
487     ky  = (float*) malloc1D( ny, sizeof(float), "ky" );
488     kx2 = (float*) malloc1D( nx, sizeof(float), "kx2" );
489     ky2 = (float*) malloc1D( ny, sizeof(float), "ky2" );
490     xp  = (float*) malloc1D( nx, sizeof(float), "x2" );
491     yp  = (float*) malloc1D( ny, sizeof(float), "y2" );
492
493     freqn( kx, kx2, xp, nx, ax );
494     freqn( ky, ky2, yp, ny, by );
495
496     kxp  = (float*) malloc1D( nxprobe, sizeof(float), "kxp" );
497     kyp  = (float*) malloc1D( nyprobe, sizeof(float), "kyp" );
498     kxp2 = (float*) malloc1D( nxprobe, sizeof(float), "kxp2" );
499     kyp2 = (float*) malloc1D( nyprobe, sizeof(float), "kyp2" );
500
501     freqn( kxp, kxp2, xp, nxprobe, ax*((double)nxprobe)/nx );
502     freqn( kyp, kyp2, yp, nyprobe, by*((double)nyprobe)/ny );
503    printf("xp=%fax=%g\n",*xp,ax);
504 /*impose anti-aliasing bandwidth limit on transmission functions*/
505
506     sum = ((double)nx)/(2.0*ax);
507     k2maxp = ((double)ny)/(2.0*by);
508     if( sum < k2maxp ) k2maxp = sum;
509     k2maxp= BW * k2maxp;
510     printf("Bandwidth limited to a real space resolution of %f ↩
            Angstroms\n",
511                         1.0F/k2maxp);
512     printf("   (= %.2f mrad)  for symmetrical anti-aliasing.\n",
513          wavlen*k2maxp*1000.0F);
514     k2maxp = k2maxp * k2maxp;
515
516     /*  allocate some more arrays and initialize propagator */
517
518     propxr = (float*) malloc1D( nxprobe, sizeof(float), "propxr" );
519     propxi = (float*) malloc1D( nxprobe, sizeof(float), "propxi" );
520     propyr = (float*) malloc1D( nyprobe, sizeof(float), "propyr" );
521     propyi = (float*) malloc1D( nyprobe, sizeof(float), "propyi" );
522
523     /* calculate propagator functions with probe sample size
524         impose anti-aliasing bandwidth limit  */
525     tctx = 2.0 * tan(ctiltx);
526     tcty = 2.0 * tan(ctilty);
527
528     scale = pi * deltaz;
529     for( ix=0; ix<nxprobe; ix++) {
530         w = scale * ( kxp2[ix] * wavlen - kxp[ix]*tctx );
531         propxr[ix]= (float)  cos(w);
532         propxi[ix]= (float) -sin(w);
533     }
534
535     for( iy=0; iy<nyprobe; iy++) {
536         w = scale * ( kyp2[iy] * wavlen - kyp[iy]*tcty );
537         propyr[iy]= (float)  cos(w);
```

```
538         propyi[iy]= (float) -sin(w);
539     }
540 
541     nbeamp = 0;
542     for( iy=0; iy<nyprobe; iy++)
543     for( ix=0; ix<nxprobe; ix++) {
544         if( (kyp2[iy] + kxp2[ix]) < k2maxp ) nbeamp++;
545     }
546 
547     printf("Number of symmetrical anti-aliasing "
548            "beams in probe = %ld\n", nbeamp);
549 
550     /*  convert aperture dimensions */
551     k2min = (double*) malloc1D( ndetect, sizeof(double), "k2min" );
552     k2max = (double*) malloc1D( ndetect, sizeof(double), "k2max" );
553 
554     for( idetect=0; idetect<ndetect; idetect++) {
555         if( ADF == collectorMode[idetect] ) {
556             k2max[idetect] = 0.001 * almax[idetect]/wavlen;
557             k2max[idetect] = k2max[idetect] * k2max[idetect];
558             k2min[idetect] = 0.001 * almin[idetect]/wavlen;
559             k2min[idetect] = k2min[idetect] * k2min[idetect];
560         } else if( CONFOCAL == collectorMode[idetect] ) {
561             k2max[idetect] = almax[idetect] * almax[idetect];
562             k2min[idetect] = almin[idetect] * almin[idetect];
563         }
564     }
565 
566     /*  init the min/max record of total integrated intensity */
567 
568     totmin =  10.0;
569     totmax = -10.0;
570     detect  = (double***) malloc3D( nThick, ndetect, nprobes*nprobes,
571         sizeof(double), "detect" );
572     detect2 = (double***) malloc3D( nThick, ndetect, nprobes*nprobes,
573         sizeof(double), "detect2" );
574 sums = (double*) malloc1D( nprobes*nprobes, sizeof(double), "sums" );
575 rmin = (float**) malloc2D( nThick, ndetect, sizeof(float), "rmin" );
576 rmax = (float**) malloc2D( nThick, ndetect, sizeof(float), "rmax" );
577 
578 /* Define cuda FFT plans for probe and transmission arrays*/
579 cufftPlan2d(&planT, nx, ny, CUFFT_Z2Z);
580 cufftPlan2d(&planP, nxprobe, nyprobe, CUFFT_Z2Z);
581    trans = (cufftDoubleComplex*) malloc( nx*ny * sizeof(↩
       cufftDoubleComplex) );
582        if( NULL == trans ) {
583         printf("Cannot allocate trans array\n");
584         exit( EXIT_FAILURE );
585     }
586 
587 
588 
589 /* ---------- start here for a full image output ---------- */
590 /*
591   do one whole line at once NOT the whole image (which may be huge)
592 */
593     if( l1d == 0 ) {
```

```
594         printf("output file size in pixels is %d x %d\n",
595            nxout, nyout );
596         if( nprobes != nyout ) {
597             printf( "Error, nprobes=%d must be the same as"
598                 "nyout=%d, in image mode.\n", nprobes, nyout );
599             exit( 0 );
600         }
601 
602         pixr = (float***) malloc3D( ndetect*nThick, nxout, nyout,
603             sizeof(float), "pixr"  );
604         for( i=0; i<(nThick*ndetect); i++) {
605             for( ix=0; ix<nxout; ix++)
606             for( iy=0; iy<nyout; iy++)
607              pixr[i][ix][iy] = 0.0F;
608         }
609 
610         dx = (xf-xi)/((double)(nxout-1));
611         dy = (yf-yi)/((double)(nyout-1));
612        x = (double*) malloc1D( nprobes*nprobes, sizeof(double), "x");
613        y = (double*) malloc1D( nprobes*nprobes, sizeof(double), "y");
614 
615          /*  add random thermal displacements
616                 scaled by temperature if requested
617              remember that initial wobble is at 300K for
618                 each direction */
619          for( iwobble=0; iwobble<nwobble; iwobble++) {
620              if( lwobble == 1 ){
621                  scale = (float) sqrt(temperature/300.0) ;
622                  for( i=0; i<natom; i++) {
623                  xa2[i] = xa[i] +
624                      (float)(wobble[i]*rangauss(&iseed)*scale);
625                  ya2[i] = ya[i] +
626                      (float)(wobble[i]*rangauss(&iseed)*scale);
627                  za2[i] = za[i] +
628                          (float)(wobble[i]*rangauss(&iseed)*scale);
629                  occ2[i] = occ[i];
630                  Znum2[i] = Znum[i];
631                  }
632                  sortByZ( xa2, ya2, za2, occ2, Znum2, natom );
633                  printf("configuration # %d\n", iwobble+1 );
634                  printf( "The new range of z is %g to %g\n",
635                      za2[0], za2[natom-1] );
636              } else for( i=0; i<natom; i++) {
637                  xa2[i] = xa[i];
638                  ya2[i] = ya[i];
639                  za2[i] = za[i];
640                  occ2[i] = occ[i];
641                  Znum2[i] = Znum[i];
642              }
643              zmin = za2[0];  /* reset zmin/max after wobble */
644              zmax = za2[natom-1];
645 
646 
647    for( ix=0; ix<nxout; ix++) {
648       for( iy=0; iy<nyout; iy++) {
649 x[iy+(ix*nyout)] = xi + dx * ((double) ix) + sourcesize * rangauss(&↩
      iseed);
```

```
650 y[iy+(ix*nyout)] = yi + dy * ((double) iy) + sourcesize * rangauss(&↩
      iseed);
651 x[iy+(ix*nyout)] = periodic( x[iy+(ix*nyout)], ax );    /* put back in↩
       supercell */
652 y[iy+(ix*nyout)] = periodic( y[iy+(ix*nyout)], by );    /* if ↩
      necessary */
653
654                 }
655 }
656 /*Call main multislice function, STEMsignals*/
657 STEMsignals( x, y, nyout, detect, ndetect,
658 ThickSave, nThick, sums, fileoutpre, svbeam, nR, PixP, ixp, iwobble )↩
      ;
659 /* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
660 for( ix=0; ix<nxout; ix++) {
661           for( iy=0; iy<nyout; iy++) {
662             if( sums[iy] < totmin ) totmin = sums[iy];
663               if( sums[iy] > totmax ) totmax = sums[iy];
664                 for( it=0; it<nThick; it++){
665                    for( idetect=0; idetect<ndetect; idetect++){
666                       pixr[idetect + it*ndetect][ix][iy] += (float)
667                       (detect[it][idetect][iy+(ix*nyout)]/((double)↩
                             nwobble));
668          }
669              }
670                if( sums[iy] < 0.9)
671                printf("Warning integrated intensity too small, = "
672                "%g at x,y= %g, %g\n", sums[iy], x[iy], y[iy] );
673                   if( sums[iy] > 1.1)
674                printf("Warning integrated intensity too large, = "
675                   "%g at x,y= %g, %g\n", sums[iy], x[iy], y[iy] );
676                 }
677
678             } /* end for(ix...) */
679
680         } /* end for(iwobble... ) */
681
682         /*  output data files  */
683         for( it=0; it<nThick; it++)
684         for( i=0; i<ndetect; i++) {
685             rmin[it][i] = rmax[it][i] = pixr[i+it*ndetect][0][0];
686             for( ix=0; ix<nxout; ix++)
687             for( iy=0; iy<nyout; iy++) {
688                 temp = pixr[i+it*ndetect][ix][iy];
689                 if( temp < rmin[it][i] )rmin[it][i] = (float) temp;
690                 if( temp > rmax[it][i] )rmax[it][i] = (float) temp;
691             }
692         }
693 /*  Produce output file listing parameters for each image file */
694         sprintf( fileout, "%s.txt", fileoutpre );
695         fp = fopen( fileout, "w+" );
696         if( fp == NULL ) {
697             printf("Cannot open output file %s.\n", fileout );
698             exit( 0 );
699         }
700
701 fprintf(fp,"C\n");
```

```
fprintf(fp,"C   output of CuSTEM version %s\n", version);
fprintf(fp,"C\n");
fprintf(fp,"C   nslice= %d\n", nslice);
fprintf(fp,"deltaz= %g, file in= %s\n", deltaz, filein );
fprintf(fp,"V0= %g, Cs3= %g, Cs5= %g, df= %g\n", keV, Cs3, Cs5, df );
fprintf(fp,"Apert= %g mrad to %g mrad\n", apert1, apert2 );
fprintf(fp,"Transmision size x,y: %i, %i\n", nx, ny);
fprintf(fp,"Probe size x,y: %i, %i\n", nxprobe, nyprobe);
fprintf(fp,"Output size x,y: %i, %i\n", nxout, nyout);
fprintf(fp,"Scan range x,y (Angstoms): %g - %g, %g -%g\n",xi,xf,yi,yf↩
    );
 if( doConfocal == TRUE ) {
  fprintf(fp, "confocal lens Cs3= %g, Cs5= %g, df= %g\n",  Cs3C, Cs5C↩
      , dfC );
  fprintf(fp, "confocal lens apert= %g mrad to %g mrad\n", apert1C, ↩
     apert2C );
  fprintf(fp, "confocal dfa2C= %g, dfa2phiC= %g, dfa3C= %g, dfa3phiC↩
     =%g\n",
         dfa2C, dfa2phiC, dfa3C, dfa3phiC );
       }
fprintf(fp, "Crystal tilt x,y= %lg, %lg\n", ctiltx,ctilty);

   for(idetect=0; idetect<ndetect; idetect++) {
   if( ADF == collectorMode[idetect]  )
    fprintf(fp, "Detector %d, Almin= %g mrad, Almax= %g mrad\n",
    idetect, almin[idetect], almax[idetect] );
           else if( CONFOCAL == collectorMode[idetect] )
                fprintf(fp, "Detector %d, cmin= %g Angst, cmax= %g ↩
                   Angst.\n",
                           idetect, almin[idetect], almax[idetect] )↩
                               ;
        }

       fprintf(fp, "ax= %g A, by= %g A, cz= %g A\n", ax,by,cz);
       fprintf(fp, "Number of symmetrical anti-aliasing "
           "beams in probe wave function= %ld\n", nbeamp );
       fprintf(fp, "with a resolution (in Angstroms) = %g\n",
           1.0/sqrt(k2maxp) );
       if( lwobble == 1 ) {
          fprintf( fp,
           "Number of thermal configurations = %d\n", nwobble );
              fprintf( fp, "Source size = %g Ang. (FWHM) \n", ↩
                  sourceFWHM );
       }
   fprintf(fp, "The total integrated intensity range was:\n");
    fprintf(fp,"   %g to %g\n\n",  totmin, totmax );

    fprintf(fp,"CPU time = %g sec.\n", cputim()-timer);
#ifdef USE_OPENMP
    fprintf(fp,"wall time = %g sec.\n", walltim() - walltimer);
#endif

       fprintf( fp, "\n" );

        /*  store params plus min and max */
        param[pIMAX]    = 0.0F;
        param[pIMIN]    = 0.0F;
```

```
752         param[pXCTILT]  = (float) ctiltx;
753         param[pYCTILT]  = (float) ctilty;
754         param[pDEFOCUS] = (float) df;
755         param[pDX]      = (float) dx;
756         param[pDY]      = (float) dy;
757         param[pENERGY]  = (float) keV;
758         param[pOAPERT]  = (float) apert2;
759         param[pCS]      = (float) Cs3;
760         param[pWAVEL]   = (float) wavlen;
761         param[pNSLICES] = (float) -1.0;
762         param[35]       = (float) Cs5;
763
764         for( it=0; it<nThick; it++)
765         for( i=0; i<ndetect; i++) {
766             sprintf( fileout, "%s%03d%03d.tif", fileoutpre, i, it );
767             printf("%s: output pix range : %g to %g\n", fileout,
768                 rmin[it][i], rmax[it][i]);
769             param[pRMAX] = rmax[it][i];
770             param[pRMIN] = rmin[it][i];
771             param[pMINDET] = (float) ( almin[i] * 0.001 );
772             param[pMAXDET] = (float) ( almax[i] * 0.001 );
773             if( tcreateFloatPixFile( fileout, pixr[i+it*ndetect],
774                 (long) nxout, (long) nyout, 1, param ) != 1 ) {
775                 printf("Cannot write output file %s.\n", fileout );
776             }
777
778             if( ADF == collectorMode[i]  )
779                 fprintf(fp,"file: %s, detector= %g to %g mrad, "
780                         "thicknes= %g A, range= %g to %g\n", fileout,
781                         almin[i], almax[i], ThickSave[it], rmin[it][i↩
                            ], rmax[it][i]);
782             else if( CONFOCAL == collectorMode[i] )
783                 fprintf(fp,"file: %s, detector= %g to %g Angst., "
784                         "thicknes= %g A, range= %g to %g\n", fileout,
785                         almin[i], almax[i], ThickSave[it], rmin[it][i↩
                            ], rmax[it][i]);
786         }
787
788         fclose( fp );
789
790 ////////////////////////////////////////////////////
791 ////.....Double Precision Data Output.....////
792 ////////////////////////////////////////////////////
793 for(i=0; i<ndetect; i++){
794                 sprintf( fileout, "%sdet%i.dat", fileoutpre, i );
795       printf("output file= %s\n", fileout);
796       fp = fopen( fileout, "w+" );
797       if( fp == NULL ) {
798           printf("Cannot open output file %s.\n", fileout );
799             exit( 0 );
800       }
801       fprintf(fp, "C     x      y      signal\n");
802     for( ix=0; ix<nxout; ix++) {
803                 for( iy=0; iy<nyout; iy++) {
804                     x[iy+(ix*nyout)] = xi + dx * ((double) ix) ;
805                     y[iy+(ix*nyout)] = yi + dy * ((double) iy) ;
806    fprintf(fp, "%14.7g %14.7g", x[iy+(ix*nyout)], y[iy+(ix*nyout)]);
```

```
807     fprintf(fp, "%14.7g", pixr[i][iy][ix] );
808     fprintf(fp, "\n");
809 
810                 }/*end for iy...*/
811 } /*end for ix...*/
812         fclose( fp );
813 } /*end for i .. ndetect*/
814 ////////////////////////////////////////////////////
815 ////.........Double Precision Data Output.........////
816 ////////////////////////////////////////////////////
817 
818 
819 /* ---------------- start here for 1d line scan ------------------- */
820 
821     } else if ( l1d == 1 ) {
822 
823         dx = (xf-xi)/((double)(nout-1));
824         dy = (yf-yi)/((double)(nout-1));
825         x = (double*) malloc1D( nprobes, sizeof(double), "x" );
826         y = (double*) malloc1D( nprobes, sizeof(double), "y" );
827             for( ip=0; ip<nout; ip++) {
828                 for( it=0; it<nThick; it++)
829                 for( idetect=0; idetect<ndetect; idetect++)
830                     detect[it][idetect][ip] = 0.0;
831         }
832 
833         /*  add random thermal displacements scaled by temperature
834             if requested
835          remember that initial wobble is at 300K for each direction */
836         for( iwobble=0; iwobble<nwobble; iwobble++) {
837 
838             if( lwobble == 1 ){
839                 scale = (float) sqrt(temperature/300.0) ;
840                 for( i=0; i<natom; i++) {
841                     xa2[i] = xa[i] +
842                         (float)(wobble[i]*rangauss(&iseed)*scale);
843                     ya2[i] = ya[i] +
844                         (float)(wobble[i]*rangauss(&iseed)*scale);
845                     za2[i] = za[i] +
846                         (float)(wobble[i]*rangauss(&iseed)*scale);
847                     occ2[i] = occ[i];
848                     Znum2[i] = Znum[i];
849                 }
850                 printf("configuration # %d\n", iwobble+1 );
851                 sortByZ( xa2, ya2, za2, occ2, Znum2, natom );
852                 printf( "The new range of z is %g to %g\n",
853                     za2[0], za2[natom-1] );
854             } else for( i=0; i<natom; i++) {
855                 xa2[i] = xa[i];
856                 ya2[i] = ya[i];
857                 za2[i] = za[i];
858                 occ2[i] = occ[i];
859                 Znum2[i] = Znum[i];
860             }
861             zmin = za2[0];      /* reset zmin/max after wobble */
862             zmax = za2[natom-1];
863             for( ip=0; ip<nout; ip++) {
```

```
864                 x[ip] = xi + dx * ((double)ip)
865                             + sourcesize * rangauss(&iseed);
866                 y[ip] = yi + dy * ((double)ip)
867                             + sourcesize * rangauss(&iseed);
868                 x[ip] = periodic( x[ip], ax );   /* put back in ↩
                        supercell */
869                 y[ip] = periodic( y[ip], by );  /* if necessary */
870               }
871              printf( "Probe line %i of %i\n", (ix+1), nxout);
872               STEMsignals( x, y, nprobes, detect2, ndetect,
873                   ThickSave, nThick, sums, fileoutpre, svbeam, nR, PixP↩
                        , ixp, iwobble );
874               for( ip=0; ip<nprobes; ip++) {
875                   if( sums[ip] < totmin ) totmin = sums[ip];
876                   if( sums[ip] > totmax ) totmax = sums[ip];
877                   for( it=0; it<nThick; it++){
878                   for( idetect=0; idetect<ndetect; idetect++)
879                      detect[it][idetect][ip] +=
880                           detect2[it][idetect][ip]/((double)nwobble);
881                   }
882                   if( sums[ip] < 0.9)
883                   printf("Warning integrated intensity too small, = %g"
884                       " at x,y= %g, %g\n", sums[ip], x[ip], y[ip] );
885                   if( sums[ip] > 1.1)
886                   printf("Warning integrated intensity too large, = %g"
887                       " at x,y= %g, %g\n", sums[ip], x[ip], y[ip] );
888               }
889
890           }  /* end for(iwobble... */
891
892           /* ------ Output text data ------------------ */
893           sprintf( fileout, "%s.txt", fileoutpre );
894           printf("output file= %s\n", fileout);
895
896           fp = fopen( fileout, "w+" );
897           if( fp == NULL ) {
898               printf("Cannot open output file %s.\n", fileout );
899                 exit( 0 );
900           }
901
902           fprintf(fp, "C\n");
903           fprintf(fp, "C   output of CuSTEM version %s\n", version);
904           fprintf(fp, "C\n");
905           fprintf(fp, "C   nslice= %d\n", nslice);
906           fprintf(fp,"deltaz= %g, file in= %s\n", deltaz, filein );
907           fprintf(fp, "V0= %g, Cs3= %g, Cs5= %g, df= %g\n", keV, Cs3, ↩
                  Cs5, df );
908           fprintf(fp, "Apert= %g mrad to %g mrad\n", apert1, apert2 );
909           if( doConfocal == TRUE ) {
910              fprintf(fp, "confocal lens Cs3= %g, Cs5= %g, df= %g\n",  ↩
                     Cs3C, Cs5C, dfC );
911              fprintf(fp, "confocal lens apert= %g mrad to %g mrad\n", ↩
                     apert1C, apert2C );
912              fprintf(fp, "confocal dfa2C= %g, dfa2phiC= %g, dfa3C= %g, ↩
                     dfa3phiC=%g\n",
913                   dfa2C, dfa2phiC, dfa3C, dfa3phiC );
914           }
```

```
915         fprintf(fp, "Crystal tilt x,y= %lg, %lg\n", ctiltx,ctilty);
916
917         for(  idetect=0; idetect<ndetect; idetect++) {
918              if( ADF == collectorMode[idetect]  )
919                  fprintf(fp, "Detector %d, Almin= %g mrad, Almax= %g ↩
                         mrad\n",
920                                  idetect, almin[idetect], almax[idetect] )↩
                                     ;
921              else if( CONFOCAL == collectorMode[idetect] )
922                  fprintf(fp, "Detector %d, cmin= %g Angst, cmax= %g ↩
                         Angst.\n",
923                                  idetect, almin[idetect], almax[idetect] )↩
                                     ;
924         }
925
926         fprintf(fp, "ax= %g A, by= %g A, cz= %g A\n", ax,by,cz);
927         fprintf(fp, "Number of symmetrical anti-aliasing "
928             "beams in probe wave function= %ld\n", nbeamp );
929         fprintf(fp, "with a resolution (in Angstroms) = %g\n",
930             1.0/sqrt(k2maxp) );
931         if( lwobble == 1 ) {
932                fprintf( fp,
933                 "Number of thermal configurations = %d\n", nwobble );
934                    fprintf( fp, "Source size = %g Ang. (FWHM) \n", ↩
                           sourceFWHM );
935         }
936         fprintf(fp, "C     x      y     signal\n");
937
938         for( ip=0; ip<nprobes; ip++) {
939             /* recalculate mean x,y without source size wobble */
940             x[ip] = xi + dx * ((double)ip);
941             y[ip] = yi + dy * ((double)ip);
942             fprintf(fp, "%14.7g %14.7g", x[ip], y[ip]);
943             for(i=0; i<ndetect; i++)
944                 fprintf(fp, "%14.7g", detect[nThick-1][i][ip] );
945             fprintf(fp, "\n");
946         }
947
948         fclose( fp );
949
950 /* ------- next output xz image data ------------------- */
951         if( lxzimage == 1 ) {
952
953         /*  directory file listing parameters for each image file */
954              sprintf( fileout, "%sDATA.txt", fileoutpre );
955              fp = fopen( fileout, "w+" );
956              if( fp == NULL ) {
957                  printf("Cannot open output file %s.\n", fileout );
958                  exit( 0 );
959              }
960
961              /*  store params plus min and max */
962              param[pIMAX]    = 0.0F;
963              param[pIMIN]    = 0.0F;
964              param[pXCTILT]  = (float) ctiltx;
965              param[pYCTILT]  = (float) ctilty;
966              param[pDEFOCUS] = (float) df;
```

```
967             param[pDX]   = (float) dx;
968             param[pDY]   = (float) dy;
969             param[pENERGY]   = (float) keV;
970             param[pOAPERT]   = (float) apert2;
971             param[pCS]   = (float) Cs3;
972             param[pWAVEL]    = (float) wavlen;
973             param[pNSLICES] = (float) -1.0;
974             param[35]    = (float) Cs5;
975
976             pixout = (float**) malloc2D( nprobes*nprobes, nThick,
977                     sizeof(float), "pixout"  );
978
979             for( idetect=0; idetect<ndetect; idetect++){
980                 sprintf( fileout, "%s%03d.tif", fileoutpre, idetect )↩
                        ;
981                 printf("output file= %s\n", fileout);
982
983                 /* convert to float and fix pixel order */
984                 pmin = pmax = (float) detect[0][idetect][0];
985                 for( ix=0; ix<nprobes; ix++)
986                 for( iy=0; iy<nThick; iy++) {
987                     temp = pixout[ix][iy] = (float) detect[iy][↩
                            idetect][ix];
988                     if( temp < pmin )pmin = temp;
989                     if( temp > pmax )pmax = temp;
990                 }
991
992                 printf("%s: output pix range : %g to %g\n", fileout, ↩
                        pmin, pmax);
993                 param[pRMAX] = pmax;
994                 param[pRMIN] = pmin;
995                 if( collectorMode[idetect] == ADF ) {
996                     param[pMINDET] = (float) ( almin[idetect] * 0.001↩
                            );
997                     param[pMAXDET] = (float) ( almax[idetect] * 0.001↩
                            );
998                 } else if( collectorMode[idetect] == CONFOCAL ) {
999                     param[pMINDET] = (float) almin[idetect];
1000                    param[pMAXDET] = (float) almax[idetect];
1001                }
1002                if( tcreateFloatPixFile( fileout, pixout,
1003                    (long) nprobes, (long) nThick, 1, param ) != 1 ) ↩
                            {
1004                        printf("Cannot write output file %s.\n", ↩
                                fileout );
1005                }
1006                fprintf(fp,"file: %s, detector= %g to %g mrad, range=↩
                        %g to %g\n",
1007                fileout, almin[idetect], almax[idetect], pmin, pmax);
1008            }
1009            fclose( fp );
1010
1011        }  /* end if( lxzimage==1... */
1012
1013    } /* end if( l1d.. ) */
1014
1015    printf("Number of symmetrical anti-aliasing "
```

```
1016            "beams in trans. function = %ld\n", nbeamt);
1017
1018     /*  echo min/max of total integrated intensity */
1019     printf("The total integrated intensity range was:\n");
1020     printf("   %g to %g\n\n",  totmin, totmax );
1021
1022     printf("CPU time = %g sec.\n", cputim()-timer);
1023 #ifdef USE_OPENMP
1024     printf("wall time = %g sec.\n", walltim() - walltimer);
1025 #endif
1026     return( 0 );
1027
1028 }  /* end main() */
1029
1030 /*------------------------------ periodic() ------------------------------*/
1031 /*
1032      make probe positions periodic in the supercell
1033      in case some wobble off the edge with source size of user excess
1034
1035     pos = input position (x or y);
1036     size = supercell size ( 0 to size)
1037
1038     return positive value  0 <= x < size
1039 */
1040 double periodic( double pos, double size )
1041 {
1042     double x=pos;
1043     while( x < 0 ) x += size;
1044     x = fmod( x, size );
1045     return( x );
1046 }
1047
1048 /*------------------------------ STEMsignals() ------------------------------*/
1049 /*
1050   GPU multislice beam propagation
1051
1052   x[],y[]      = real positions of the incident probe
1053   npos         = int number of positions
1054   detect[][][]= real array to get signal into each detector
1055                 for each probe position and thickness
1056   ndetect      = number of detector geometries
1057   ThickSave[] = thicknesses at which to save data
1058   nThick       = number of thickness levels (including the last)
1059   sum          = real total integrated intensity
1060
1061   the assumed global variables are:
1062
1063   nxprobe,nyprobe    = int size of probe wavefunction in pixels
1064   nx,ny          = int size of transmission function in pixels
1065   layer[]        = int array with slice layer indecies
1066   prober[][], probei[][] = float real,image probe wavefunction
1067   transr[][], transi[][] = float real,imag transmission function
1068   propxr[][], propxi[][] = float real,imag propagator vs x
1069   propyr[][], propyi[][] = float real,imag propagator vs y
1070   ax,by,cz       = float unit cell size in Angs
1071   kxp[], kyp[]       = float spatial frequencies vs x, y
1072   kxp2[], kyp2[]     = float square of kxp[], kyp[]
```

```
1073   xp[], yp[]          = float real space positions in probe (confocal)
1074   apert1, apert2      = double min,max objective aperture (mrad)
1075   k2maxp              = double max spatial freq of probe squared
1076   pi                  = double constant PI
1077   wavlen              = double electron wavelength in Angs
1078   df                  = double defocus (in Ang)
1079   Cs3,Cs5             = double spherical aberration (in mm)
1080
1081   xa[],ya[],za[]      = atom coordinates
1082   occ[]           = atomic occupancy
1083   Znum[]          = atomic numbers
1084   natom           = number of atoms
1085   deltaz          = slice thickness
1086   v0              = beam energy
1087   nbeamt          = number of beams in transmission function
1088   zmin, zmax          = range of z coord. of the atoms
1089   nslice          = number of slices
1090 */
1091 void STEMsignals( double x[], double y[], int npos,
1092          double ***detect, int ndetect,
1093          double ThickSave[], int nThick, double sum[], char ↩
                fileoutpre[], int svbeam, int nR, int PixP[], int ixp, ↩
                int iwobble )
1094 {
1095     int ix, iy, ixt, iyt, idetect, *ixoff,*iyoff, ixmid, iymid, ox, ↩
           oy, ndone;
1096     int istart, na, ip, i, j, jt, it, iR, bm, pid, offst;
1097     FILE *fp;
1098     long nxprobel, nyprobel, nxl, nyl;
1099     char fileout[NCMAX];
1100     float scale, prr, pri, tr, ti;
1101     cufftDoubleComplex *cpix, *dcpix;
1102     cufftResult c1,c2,ct;
1103     double *xoff, *yoff, chi1, chi2, chi3, k2maxa, k2maxb, chi,
1104         w, k2, phi, phirms, phil;
1105     double sum0, sum1, delta, zslice, totalz;
1106    float *k2mp;
1107     /* extra for confocal */
1108     float hr, hi;
1109     double chi2C, chi3C, k2maxaC, k2maxbC, r2, rx2, phix, phiy;
1110    int width, height, depth;
1111    cudaExtent extent;
1112    cudaPitchedPtr devPitchedPtr;
1113     /* ------- make sure x,y are ok ------- */
1114     for( ip=0; ip<(npos*npos); ip++) {
1115         if( (x[ip] < 0.0) || (x[ip] > ax) ||
1116             (y[ip] < 0.0) || (y[ip] > by) ) {
1117             sum[ip] = 0.0f;
1118             printf("bad x=%f,y=%f in STEMsignals()\n", x[ip], y[ip]);
1119             return;
1120         }
1121     }
1122
1123     ixmid = nxprobe/2;
1124     iymid = nyprobe/2;
1125     chi1 = pi * wavlen;
1126     chi2 = 0.5 * Cs3 * 1.0e7*wavlen*wavlen;
```

```
1127     chi3 = Cs5 * 1.0e7 * wavlen*wavlen*wavlen*wavlen /3.0;
1128     k2maxa = apert1 * 0.001/wavlen;
1129     k2maxa = k2maxa *k2maxa;
1130     k2maxb = apert2 * 0.001/wavlen;
1131     k2maxb = k2maxb * k2maxb;
1132
1133     /* extra for confocal */
1134     chi2C = 0.5 * Cs3C * 1.e7*wavlen*wavlen;
1135     chi3C = Cs5C * 1.0e7 * wavlen*wavlen*wavlen*wavlen /3.0;
1136     k2maxaC = apert1C * 0.001/wavlen;
1137     k2maxaC = k2maxaC *k2maxaC;
1138     k2maxbC = apert2C * 0.001/wavlen;
1139     k2maxbC = k2maxbC * k2maxbC;
1140
1141     ixoff = (int*) malloc1D( npos*npos, sizeof(int), "ixoff" );
1142     iyoff = (int*) malloc1D( npos*npos, sizeof(int), "iyoff" );
1143     xoff = (double*) malloc1D( npos*npos, sizeof(double), "xoff" );
1144     yoff = (double*) malloc1D( npos*npos, sizeof(double), "yoff" );
1145
1146 //Uncomment for openMP
1147 //#pragma omp parallel for private(ix,iy,j,sum0,k2,w,phi,chi,scale,tr↩
        ,ti)
1148
1149     nxprobel = (long) nxprobe;
1150     nyprobel = (long) nyprobe;
1151
1152     nxl = (long) nx;
1153     nyl = (long) ny;
1154
1155     scale = 1.0F / ( ((float)nx) * ((float)ny) );
1156
1157     zslice = 0.75*deltaz;  /*  start a little before top of unit cell↩
            */
1158     istart = 0;
1159     nslice = 0;
1160     it = 0;      /* thickness level index */
1161
1162     if( zmax > cz ) totalz = zmax;
1163         else totalz = cz;
1164     printf( "specimen range is 0 to %g Ang.\n", totalz );
1165
1166
1167
1168 //////////BUILD TRANS LAYERS/////////////////////
1169 ////CUDAMALLOCS all up front to avoid memory leaks
1170 /*Determine number of trans slices */
1171 nslice=floor( (totalz+0.25*deltaz)/deltaz +1) ;
1172 if ( cudaSuccess !=  cudaMalloc((void**)&dtrans, sizeof(↩
        cufftDoubleComplex)*nslice*nx*ny))printf( "Error!\n" );
1173 if ( cudaSuccess !=   cudaMalloc((void**)&dtranswork, sizeof(↩
        cufftDoubleComplex)*nx*ny)   ) printf( "Error!\n" );
1174         cudaMalloc((void**)&dkx2, sizeof(float)*nx);
1175         cudaMalloc((void**)&dky2, sizeof(float)*ny);
1176         cudaMalloc((void**)&dk2max, sizeof(float));
1177         k2mp = (float*) malloc( 1 * sizeof(float) );
1178
1179 istart=0;
```

```
1180 for(is=0; is<nslice; is++){ //for over each layer
1181 offst=nx*ny*is;
1182 zslice = (0.75+is)*deltaz;
1183    na = 0;
1184        for(i=istart; i<natom; i++)
1185         if( za2[i] < zslice ) na++; else break;
1186           if( na > 0 ){
1187 /*build the trans array for this layer*/
1188     trlayer(  &xa2[istart], &ya2[istart], &occ2[istart],
1189             &Znum2[istart], na, (float)ax, (float)by, (float)keV,
1190             trans, nxl, nyl, &phirms, &nbeamt, (float) k2maxp );
1191    }
1192    istart += na;
1193 /*Copy trans layer to device and bandwidth limit*/
1194  k2mp[0]= (float) k2maxp;
1195 cudaMemcpy(dk2max, k2mp, sizeof(float), cudaMemcpyHostToDevice);
1196 cudaMemcpy(dkx2, kx2, sizeof(float)*nx, cudaMemcpyHostToDevice);
1197 cudaMemcpy(dky2, ky2, sizeof(float)*ny, cudaMemcpyHostToDevice);
1198 cudaMemcpy(dtranswork, trans, sizeof(cufftDoubleComplex)*nx*ny, ↩
       cudaMemcpyHostToDevice);
1199 cufftExecZ2Z(planT, dtranswork, dtranswork, CUFFT_INVERSE);
1200 //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1201 /*Hard set number of threads per block, never likely
1202    to use size below 16x16 so this shouldn't cause memory
1203    problem but is more efficient than 1x1
1204    NB THESE VALUES MUST BE ADJUSTED TO SUIT GPU,
1205    THESE ARE FOR USE ON NVIDIA TESLA C2075*/
1206    dim3 layerthreadsPerBlock(16, 16);
1207 /*Set number of blocks so that the total number of threads is
1208 the size of the probe array*/
1209    dim3 transnumBlocks(nx / (layerthreadsPerBlock.x), ny / (↩
          layerthreadsPerBlock.y));
1210 //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1211
1212    cudalayer<<<transnumBlocks, layerthreadsPerBlock>>>(dtranswork, ↩
          dkx2, dky2, dk2max);
1213    ct=cufftExecZ2Z(planT, dtranswork, dtrans+offst, CUFFT_FORWARD);//↩
          Last FFT and copy to storage array
1214    printf("ct=%d\n",ct);
1215 }
1216 cudaFree(dkx2);
1217 cudaFree(dky2);
1218 cudaFree(dtranswork);
1219 //////////End of building Trans layers/////////////
1220
1221 /*Copy propagator to GPU*/
1222 if ( cudaSuccess != cudaMalloc((void**)&dpropxr, sizeof(float)*↩
       nxprobe)) printf( "Errordtemp\n" );
1223 if ( cudaSuccess != cudaMalloc((void**)&dpropxi, sizeof(float)*↩
       nxprobe)) printf( "Errordtemp\n" );
1224 if ( cudaSuccess != cudaMalloc((void**)&dpropyr, sizeof(float)*↩
       nxprobe)) printf( "Errordtemp\n" );
1225 if ( cudaSuccess != cudaMalloc((void**)&dpropyi, sizeof(float)*↩
       nxprobe)) printf( "Errordtemp\n" );
1226 cudaMemcpy(dpropxr, propxr, sizeof(float)*nxprobe, ↩
       cudaMemcpyHostToDevice);
1227 cudaMemcpy(dpropxi, propxi, sizeof(float)*nxprobe, ↩
```

```
    cudaMemcpyHostToDevice);
1228 cudaMemcpy(dpropyr, propyr, sizeof(float)*nxprobe, ↩
    cudaMemcpyHostToDevice);
1229 cudaMemcpy(dpropyi, propyi, sizeof(float)*nxprobe, ↩
    cudaMemcpyHostToDevice);
1230 if ( cudaSuccess != cudaMalloc((void**)&dkxp2, sizeof(float)*nxprobe)↩
    ) printf( "Errordtemp\n" );
1231 if ( cudaSuccess != cudaMalloc((void**)&dkyp2, sizeof(float)*nxprobe)↩
    ) printf( "Errordtemp\n" );
1232 cudaMemcpy(dkxp2, kxp2, sizeof(float)*nxprobe, cudaMemcpyHostToDevice↩
    );
1233 cudaMemcpy(dkxp2, kxp2, sizeof(float)*nxprobe, cudaMemcpyHostToDevice↩
    );
1234
1235
1236 ndone=0;
1237 //.....MAIN LOOP OVER EACH INCIDENT PIXEL POSITION.....//
1238 //UNCOMMENT for openMP
1239 /*#pragma omp parallel for private(ip,sum0,ix,j,iy,k2,phix,phiy,phil,
1240 dtemp,w,phi,dtranswork,chi,it,probe,prr,scale,pri,is,delta,offst,↩
    idetect,
1241 hr,hi,cpix,dcpix,rx2,r2,sum1,istart,zslice,na,dprobe,tr,ti,ixt,iyt,jt↩
    ,bm,
1242 iR,fileout,fp,dtranspara,transpara,ixp,i)
1243 */
1244
1245 /*Loop over incident focussed beam positions*/
1246 for( ip=0; ip<npos*npos; ip++) {
1247 ////////////GENERATE probe////////////////
1248 printf("%i / %i done\n", ndone,npos*npos);
1249 if ( cudaSuccess !=  cudaMalloc((void**)&dtranspara, sizeof(int)*4))↩
     printf( "Errordtranspara\n");
1250 if ( cudaSuccess !=  cudaMalloc((void**)&dtranswork, sizeof(↩
    cufftDoubleComplex)*nx*ny)  ) printf( "Errordtranswork\n" );
1251 if ( cudaSuccess != cudaMalloc((void**)&dprobe, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe) )  printf( "Errordprobe\n" ↩
    );
1252 if ( cudaSuccess != cudaMalloc((void**)&dcpix, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe)  ) printf( "Errordcpix↩
    \n" );
1253 if ( cudaSuccess !=  cudaMalloc((void**)&dtemp, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe)  ) printf( "Errordtemp\n"↩
     );
1254   transpara = (int*) malloc( 4 * sizeof(int) );
1255       cpix = (cufftDoubleComplex*) malloc( nxprobe*nyprobe * sizeof↩
          (cufftDoubleComplex) );
1256  probe = (cufftDoubleComplex*) malloc( nxprobe*nyprobe * sizeof(↩
     cufftDoubleComplex) );
1257       if( NULL == probe ) {
1258           printf("Cannot allocate probe array\n");
1259           exit( EXIT_FAILURE );
1260       }
1261
1262       ixoff[ip] = (int) floor( x[ip]*((double)nx) / ax ) - ixmid; ↩
          /*floor returns largest integer less than operand*/
1263       xoff[ip]  = x[ip] - ax*((double)ixoff[ip])/((double)nx);
1264
```

```
1265         iyoff[ip] = (int) floor( y[ip]*((double)ny) / by ) - iymid;
1266         yoff[ip]  = y[ip] - by*((double)iyoff[ip])/((double)ny);
1267         sum0 = 0.0;
1268         for( ix=0; ix<nxprobe; ix++) {
1269             j = ix*nyprobe;
1270             for( iy=0; iy<nyprobe; iy++) {
1271                 k2 = kxp2[ix] + kyp2[iy];
1272                 if( (k2 >= k2maxa) && (k2 <= k2maxb) ) {
1273       phix=ix;
1274     phiy=iy;
1275       if( ix > ixmid ) phix = (double) (ix-nx);
1276     if( iy > iymid ) phiy = (double) (iy-ny);
1277       phil= atan2(phiy, phix);
1278                     w = 2.*pi* ( xoff[ip]*kxp[ix] + yoff[ip]*kyp[iy] ↩
                          );
1279                     phi = atan2( ky[iy], kx[ix] );
1280                     chi = chi1*k2* ( (chi2 + chi3*k2)*k2 - df
1281                         + dfa2*sin( 2.0*(phi-dfa2phi) )
1282                         + 2.0F*dfa3*wavlen*sqrt(k2)*
1283                         sin( 3.0*(phi-dfa3phi) )/3.0 );
1284           chi= - chi + w + (l*phil);
1285                     probe[iy + j].x = tr = (float) cos( chi );
1286                     probe[iy + j].y = ti = (float) sin( chi );
1287                     sum0 += (double) (tr*tr + ti*ti);
1288                 } else {
1289                     probe[iy + j].x = 0.0F;
1290                     probe[iy + j].y = 0.0F;
1291                 }
1292             }
1293         }  /* end for( ix... */
1294        scale = (float) ( 1.0/sqrt(sum0) );
1295         for( ix=0; ix<nxprobe; ix++) {
1296             j = ix*nyprobe;
1297             for( iy=0; iy<nyprobe; iy++) {
1298                 probe[j+iy].x *= scale;
1299                 probe[j+iy].y *= scale;
1300             }
1301         }
1302 //////Beam wavefunction generated/////////
1303
1304        transpara[0]= (int) nx; /* Pointer needed for cuda transfer*/
1305         transpara[1]= (int) ny;
1306         transpara[2]= (int) ixoff[ip];
1307         transpara[3]= (int) iyoff[ip];
1308 /*Copy specimen slices to GPU memory*/
1309 if ( cudaSuccess != cudaMemcpy(dtranspara, transpara, sizeof(int)*4, ↩
      cudaMemcpyHostToDevice)) printf("errorcopy1\n") ;
1310 cudaMemcpy(dprobe, probe, sizeof(cufftDoubleComplex)*nxprobe*nyprobe,↩
       cudaMemcpyHostToDevice);
1311 /*Set number of threads per block, never likely to use size
1312    below 16x16 so this shouldn't cause memory problem but
1313    is more efficient than 1x1 */
1314    dim3 transthreadsPerBlock(16, 16);
1315 /*Set number of blocks so that the total number of threads is
1316    the size of the probe array*/
1317         dim3 transnumBlocks(nxprobe /(transthreadsPerBlock.x), ↩
             nyprobe /(transthreadsPerBlock.y));
```

```
1318
1319 ////////////////////Propagate probe through all layers////////////
1320 istart=0;
1321 for(is=0; is<nslice;is++){
1322 zslice = (0.75+is)*deltaz;
1323 offst=nx*ny*is;
1324
1325 for(i=istart; i<natom; i++)
1326             if( za2[i]<zslice ) na++; else break;
1327                if( na > 0 ){
1328 if ( cudaSuccess !=  cudaMemcpy(dtranswork, dtrans+offst, sizeof(↩
    cufftDoubleComplex)*nx*ny, cudaMemcpyDeviceToDevice)) printf("↩
    errorcopy3\n");
1329
1330   c1=cufftExecZ2Z(planP, dprobe, dprobe, CUFFT_FORWARD);
1331
1332   /*Call kernel*/
1333   cudatrans<<<transnumBlocks, transthreadsPerBlock>>>(dtranswork, ↩
       dprobe, dtemp, dtranspara);
1334   /*Execute forward FFT*/
1335   c2=cufftExecZ2Z(planP, dprobe, dprobe, CUFFT_INVERSE);
1336   /*Copy transformed dprobe array to probe array on host*/
1337 if(c1+c1!=0){ printf("cufft1: %d, cufft2: %d\n", c1, c2);}
1338
1339 }
1340           /*  multiply by the propagator function on GPU*/
1341  cudaprop<<<transnumBlocks, transthreadsPerBlock>>>(dprobe, dtemp, ↩
     dkxp2,dkyp2,dpropxr,dpropxi,dpropyr,dpropyi,dk2max);
1342
1343 /*If exit layer, copy to host*/
1344        for( it = 0; it<nThick; it++ )
1345        if( fabs(ThickSave[it]-zslice)<fabs(0.5*deltaz)) {
1346 if ( cudaSuccess !=      cudaMemcpy(probe, dprobe, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe, cudaMemcpyDeviceToHost)) ↩
    printf("errorcopy4\n");
1347   sum[ip]=0.0;
1348         for(ix=0; ix<ndetect; ix++) detect[it][ix][ip] = 0.0;
1349 /*sum intensity incident on the ADF detector and
1350 calculate total integrated intensity*/
1351                for( ix=0; ix<nxprobe; ix++) {
1352                    j = ix*nyprobe;
1353                    for( iy=0; iy<nyprobe; iy++) {
1354                      prr = probe[iy + j].x;
1355                        pri = probe[iy + j].y;
1356                        delta = prr*prr + pri*pri;
1357                        sum[ip] += delta;
1358                        k2 = kxp2[ix] + kyp2[iy];
1359                        for( idetect=0; idetect<ndetect; idetect++) {
1360                            if( ADF == collectorMode[idetect] ) {
1361                                if( (k2 >= k2min[idetect] ) &&
1362                                (k2 <= k2max[idetect] ) ){
1363                                detect[it][idetect][ip] += delta;
1364                     }
1365                            }
1366                      }
1367                    } /* end for(iy..) */
1368                } /* end for(ix...) */
```

```
1369
1370 /*transform back if confocal needed
1371 - use copy of probe so original can continue in use*/
1372                 if( doConfocal == TRUE ) {
1373                     sum0 = 0;
1374                     for( ix=0; ix<nxprobe; ix++) {
1375                         j = ix*nyprobe;
1376                         for( iy=0; iy<nyprobe; iy++) {
1377                             k2 = kxp2[ix] + kyp2[iy];
1378                             if( (k2 >= k2maxaC) && (k2 <= k2maxbC) ) ↩
                                    {
1379                                 phi = atan2( ky[iy], kx[ix] );
1380    /*  offset defocus by zslice so both lens referenced to
1381    entrance surface of specimen  */
1382                                 chi = chi1*k2* ( (chi2C + chi3C*k2)*↩
                                        k2 - dfC + zslice
1383                                     + dfa2C*sin( 2.0*(phi-dfa2phiC) )
1384                                     + 2.0F*dfa3C*wavlen*sqrt(k2)*
1385                                     sin( 3.0*(phi-dfa3phiC) )/3.0 );
1386                                 chi= - chi;
1387                                 hr = (float) cos( chi );
1388                                 hi = (float) sin( chi );
1389                                 prr = probe[iy + j].x;  /* real */
1390                                 pri = probe[iy + j].y;  /* imag */
1391                                 cpix[iy + j].x = prr*hr -pri*hi;
1392                                 cpix[iy + j].y = prr*hi +pri*hr;
1393                                 sum0 += prr*prr + pri*pri;
1394                             } else {
1395                                 cpix[iy + j].x = 0.0F;
1396                                 cpix[iy + j].y = 0.0F;
1397                             }
1398                         }  /*  end for( iy... )  */
1399                     }  /*  end for( ix... )  */
1400
1401
1402 if ( cudaSuccess !=  cudaMemcpy(dcpix, cpix, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe, cudaMemcpyHostToDevice)) ↩
    printf("errorcopy5\n");
1403
1404 if(0 !=  cufftExecZ2Z(planP, dcpix, dcpix, CUFFT_FORWARD))printf("↩
    cuffter3\n");
1405
1406 if ( cudaSuccess !=  cudaMemcpy(cpix, dcpix, sizeof(↩
    cufftDoubleComplex)*nxprobe*nyprobe, cudaMemcpyDeviceToHost)) ↩
    printf("errorcopy6\n");
1407                     scaleW( cpix, nxprobe, nyprobe );
1408
1409                     /* find normalization constant
1410                       i.e. correct for constants in the FFT */
1411                     sum1 = 0.0;
1412                     for( ix=0; ix<nxprobe; ix++) {
1413                         j = ix*nyprobe;
1414                         for( iy=0; iy<nyprobe; iy++) {
1415                             prr = cpix[iy + j].x;
1416                             pri = cpix[iy + j].y;
1417                             sum1 += prr*prr + pri*pri;
1418                         }
```

```
1419                    }
1420
1421 /*~~~~ integrate wavefunction over real space detector ~~~~*/
1422                    for( ix=0; ix<nxprobe; ix++) {
1423                            rx2 = xoff[ip] - xp[ix];
1424                            rx2 = rx2*rx2;
1425                            j = ix*nyprobe;
1426                            for( iy=0; iy<nyprobe; iy++) {
1427                                r2 = yoff[ip] - yp[iy];
1428                                r2 = rx2 + r2*r2;
1429                                prr = cpix[iy + j].x;
1430                                pri = cpix[iy + j].y;
1431                                delta = prr*prr + pri*pri;
1432                                for( idetect=0; idetect<ndetect; ↩
                                       idetect++) {
1433                                 if( CONFOCAL == collectorMode[↩
                                        idetect] ) {
1434                                   if( (r2 >= k2min[idetect] ) &&
1435                                             (r2 <= k2max[idetect] ↩
                                                 ) )
1436                                      detect[it][idetect][ip] += ↩
                                          delta*(sum0/sum1);
1437                                }
1438                            }
1439                        } /* end for( iy... )*/
1440                    } /*  end for( ix....) */
1441
1442                                } /* end if( doConfocal==TRUE) */
1443 istart+=na;
1444  } /*for over detector recording depths, iTHick*/
1445  } /* for loop over slices*/
1446 free(cpix);
1447 free(transpara);
1448 free(probe);
1449 cudaFree(dtranswork);
1450 cudaFree(dprobe);
1451 cudaFree(dcpix);
1452 cudaFree(dtemp);
1453 cudaFree(dtranspara);
1454 ndone++;
1455         } /* for loop over probe positions*/
1456
1457     free( ixoff );
1458     free( iyoff );
1459     free( xoff );
1460     free( yoff );
1461
1462 cudaDeviceReset();
1463
1464     return;
1465
1466 }/* end STEMsignals() */
1467
1468
1469 /*------------------------ trlayer() ---------------------------*/
1470 /*
1471   Calculate complex specimen transmission function
```

```
1472   for one layer using real space projected atomic potentials
1473
1474   x[],y[] = real array of atomic coordinates
1475   occ[]   = real array of occupancies
1476   Znum[]  = array of atomic numbers
1477   natom   = number of atoms
1478   ax, by  = size of transmission function in Angstroms
1479   kev     = beam energy in keV
1480   transr  = 2D array to get real part of specimen
1481         transmission function
1482   transi  = 2D array to get imag part of specimen
1483         transmission function
1484   nx, ny  = dimensions of transmission functions
1485   *phirms = average phase shift of projected atomic potential
1486   *nbeams = will get number of Fourier coefficients
1487   k2max   = square of max k = bandwidth limit
1488
1489 */
1490 void trlayer( const float x[], const float y[], const float occ[],
1491     const int Znum[], const int natom,
1492     const float ax, const float by, const float kev,
1493     cufftDoubleComplex *trans, const long nx, const long ny,
1494     double *phirms, long *nbeams, const float k2max)
1495 {
1496     int idx, idy, i, j, ixo, iyo, ix, iy, ixw, iyw, nx1, nx2, ny1, ↩
        ny2;
1497     float k2, *k2mp, *dk2max;
1498     double r, rx2, rsq, vz, rmin, rmin2, sum, scale, scalex, scaley;
1499 /* max atomic radius in Angstroms */
1500     const double rmax=3.0, rmax2=rmax*rmax;
1501
1502     scale = sigma( kev ) / 1000.0;  /* in 1/(volt-Angstroms) */
1503     scalex = ax/nx;
1504     scaley = by/ny;
1505     /* min radius to avoid  singularity */
1506     rmin = ax/((double)nx);
1507     r = by/((double)ny);
1508     rmin =  0.25 * sqrt( 0.5*(rmin*rmin + r*r) );
1509     rmin2 = rmin*rmin;
1510
1511     idx = (int) ( nx*rmax/ax ) + 1;
1512     idy = (int) ( ny*rmax/by ) + 1;
1513     for( ix=0; ix<nx; ix++) {
1514         j = ix*ny;
1515         for( iy=0; iy<ny; iy++)
1516            trans[j++].x = 0.0F;
1517     }
1518
1519     for( i=0; i<natom; i++) {
1520         ixo = (int) ( x[i]/scalex ); //position in pixels
1521         iyo = (int) ( y[i]/scaley );
1522         nx1 = ixo - idx;
1523         nx2 = ixo + idx;
1524         ny1 = iyo - idy;
1525         ny2 = iyo + idy;
1526
1527     /* add proj. atomic potential at a local region near its center
```

```
1528        taking advantage of small range of atomic potential */
1529
1530         for( ix=nx1; ix<=nx2; ix++) {
1531             rx2 = x[i] - ((double)ix)*scalex;
1532             rx2 = rx2 * rx2;
1533             ixw = ix;
1534             while( ixw < 0 ) ixw = ixw + nx;
1535             ixw = ixw % nx;
1536             j = ixw*ny;
1537             for( iy=ny1; iy<=ny2; iy++) {
1538                 rsq = y[i] - ((double)iy)*scaley;
1539                 rsq = rx2 + rsq*rsq;
1540                 if( rsq <= rmax2 ) {
1541                   iyw = iy;
1542                   while( iyw < 0 ) iyw = iyw + ny;
1543                   iyw = iyw % ny;
1544                   if( rsq < rmin2 ) rsq = rmin2;
1545                   vz = occ[i] * vzatomLUT( Znum[i], rsq );
1546                   trans[iyw + j].x += (float) vz;
1547                 }
1548             } /* end for(iy... */
1549        }  /* end for(ix... */
1550
1551     } /* end for(i=0... */
1552
1553     /* convert phase to a complex transmission function */
1554     sum = 0;
1555     for( ix=0; ix<nx; ix++) {
1556         j = ix*ny;
1557         for( iy=0; iy<ny; iy++) {
1558             vz = scale * trans[j].x;
1559             sum += vz;
1560             trans[j].x = (float) cos( vz );
1561             trans[j++].y = (float) sin( vz );
1562         }
1563     }
1564
1565     *phirms = sum / ( ((double)nx)*((double)ny) );
1566     return;
1567
1568  }  /* end trlayer() */
1569
1570
1571 //////////CUDALAYER Kernel ////////////////
1572 ////////////////////////////////////////////
1573 __global__ void cudalayer(cufftDoubleComplex *dtranswork, float *dkx2↩
        , float *dky2, float *dk2max ){
1574
1575    int ny= gridDim.y * blockDim.y;
1576    int nx= gridDim.x * blockDim.x;
1577    int idx= blockIdx.x * blockDim.x + threadIdx.x;
1578    int idy= blockIdx.y * blockDim.y + threadIdx.y;
1579    int j = idx * ny;
1580    float k2 = dky2[idy] + dkx2[idx];
1581    float scale= 1.0F/( nx * ny);
1582
1583    /*Bandwidth limit trans function*/
```

```
1584     if (k2 >= dk2max[0]) dtranswork[idy + j].x = dtranswork[idy + j].y↩
             = 0.0F;
1585
1586     /*Post FFT scaling */
1587     dtranswork[idy+j].x = dtranswork[idy+j].x * scale;
1588     dtranswork[idy+j].y = dtranswork[idy+j].y * scale;
1589
1590 } /*END of cudalayer*/
1591
1592 //////////CUDATRANS Kernel ////////////////
1593 ////////////////////////////////////////////
1594 __global__ void cudatrans(cufftDoubleComplex *dtranswork, ↩
        cufftDoubleComplex *dprobe, cufftDoubleComplex *dtemp, int *↩
        dtranspara){
1595
1596     int nx= dtranspara[0];
1597     int ny = dtranspara[1];
1598     int dxoff = dtranspara[2];
1599     int dyoff = dtranspara[3];
1600     int nyprobe= gridDim.y * blockDim.y;
1601     int nxprobe= gridDim.x * blockDim.x;
1602     int idx= blockIdx.x * blockDim.x + threadIdx.x;
1603     int idy= blockIdx.y * blockDim.y + threadIdx.y;
1604     int ixt = blockIdx.x * blockDim.x + threadIdx.x + dxoff;
1605     int iyt = blockIdx.y * blockDim.y + threadIdx.y + dyoff;
1606     float scale= 1.0F/( nxprobe * nyprobe);
1607
1608     /*Introduce periodicity*/
1609     if( idx+dxoff >= nx ) ixt = idx+dxoff - nx;
1610     if( idx+dxoff < 0 ) ixt = idx+dxoff + nx;
1611     if( idy+dyoff >= ny ) iyt = idy+dyoff - ny;
1612     if( idy+dyoff < 0 ) iyt = idy+dyoff + ny;
1613
1614     int jprobe = idx * nyprobe;
1615     int jtrans = ixt * ny;
1616
1617     /*Post FFT scaling */
1618     dprobe[idy+jprobe].x = dprobe[idy+jprobe].x * scale;
1619     dprobe[idy+jprobe].y = dprobe[idy+jprobe].y * scale;
1620
1621     /*Transmission multiplication*/
1622     dtemp[idy+jprobe].x = dprobe[idy+jprobe].x * dtranswork[iyt+jtrans↩
            ].x - dprobe[idy+jprobe].y * dtranswork[iyt+jtrans].y;
1623     dtemp[idy+jprobe].y = dprobe[idy+jprobe].x * dtranswork[iyt+jtrans↩
            ].y + dprobe[idy+jprobe].y * dtranswork[iyt+jtrans].x;
1624
1625     dprobe[idy+jprobe]=dtemp[idy+jprobe];
1626
1627 } /*END of cudatrans*/
1628
1629 //////////CUDAPROP Kernel ////////////////
1630 ////////////////////////////////////////////
1631 __global__ void cudaprop(cufftDoubleComplex *dprobe, ↩
        cufftDoubleComplex *dtemp, float *dkxp2, float *dkyp2, float *↩
        dpropxr, float *dpropxi, float *dpropyr, float *dpropyi, float *↩
        dk2max){
1632
```

```
1633         int nyprobe= gridDim.y * blockDim.y;
1634         int nxprobe= gridDim.x * blockDim.x;
1635         int idx= blockIdx.x * blockDim.x + threadIdx.x;
1636         int idy= blockIdx.y * blockDim.y + threadIdx.y;
1637         int jprobe = idx * nyprobe;
1638 
1639         dtemp[idy+jprobe].x= dprobe[idy+jprobe].x * dpropyr[idy] - ↩
                dprobe[idy+jprobe].y * dpropyi[idy];
1640         dtemp[idy+jprobe].y= dprobe[idy+jprobe].x * dpropyi[idy] + ↩
                dprobe[idy+jprobe].y * dpropyr[idy];
1641 
1642         dprobe[idy+jprobe].x=dtemp[idy+jprobe].x * dpropxr[idx] - ↩
                dtemp[idy+jprobe].y * dpropxi[idx];
1643         dprobe[idy+jprobe].y=dtemp[idy+jprobe].x * dpropxi[idx] + ↩
                dtemp[idy+jprobe].y * dpropxr[idx];
1644 
1645 } /*END of cudaprop*/
```

3 Cluster source

A Fortran90 program for producing face centred cubic cluster models with five-fold twinning.

```
!!!!!!!!!!!!!!!!!Cluster modelling program!!!!!!!!!!!!!!!!!
!!This fortran program can be used to model icosahedral !!!
!!and decahedral nanocluster systems. It works by first !!!
!!defining a tetrahedral structure and then multipling  !!!
!!it about various axis to produce the whole cluster.!!!!!!
!!!!!!!!!!!!Written by R Aveyard 06/04/2011 !!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!Edited 7/04/2011 to include an extra subroutine which
!aligns the primary axis of the nanocluster with the
!cartesian z-axis!

PROGRAM Clustersauce
    IMPLICIT NONE
    INTEGER :: Na=0, i=0, st=0, Ns=0, f=0, q=0, s=0, ip=0
!Na is the number of atoms in the cluster.
!i is the index used to identify each atom in the cluster.
!st is a variable used to define the structure geometry required
!Ns is the number of shells required, entered by the user.
!s is the summation in the equation to determine the number of
!atoms in a cluster from the number of shells entered by the user.
!f, q and ip are used integers used for DO loops

    DOUBLE PRECISION :: a
!a is the atomic separation in the cluster, entered by the user.
    DOUBLE PRECISION, ALLOCATABLE , DIMENSION(:,:) :: coord
!coord is the main array containing the x, y and z coordinates of
!each atom in the system. Its size is defined by the number of
!atoms in the cluster which is determined once the user has entered
!the required number of shells.


    PRINT*, 'Enter number of shells'
!Reading user inputs for the number of shells,
!atomic separation and cluster geometry required.
    READ*, Ns
    PRINT*, 'Enter bond length'
    READ*, a
    print*, 'Choose cluster geometry: 1)Icosahedron, 2)Decahedron, 3)Inodecahedron, 4)Marks decahedron'
    READ*, st


    DO f=1, Ns
!DO loop used to calculate a summation to give s. s is then
!used to determine the number of atoms in the cluster from
!the required number of shells.
    s=10*(f**2)+2
    END DO
```

```
48    Na=1+ s      !Defining the number of atoms so that
49 ! the size of the main array can be defined.
50 !Define size of array containing coordinates.
51    ALLOCATE( coord(Na*6, 3))
52    coord=0      !Initialization of coord array.
53
54
55    SELECT CASE (st)
56 !Case used to determine which subroutines are needed
57 !depending on the required geometry.
58
59       CASE (1)
60 !Case for icosahedral structures which require more
61 !subroutines than decahedra
62
63          CALL Tetrahedron(Na, a, coord, i, Ns)
64 !Subroutine which constructs a tetrahedral unit cell which will
65 !be rotated in various ways to produce the entire cluster.
66
67          CALL Mirror(coord, Na, i, Ns, a)
68 !Subroutine adds a mirror image to produce a double tetrahedral cell.
69
70          CALL Rotate(coord, Na, i, Ns)
71 !Subroutine appends rotated versions of the double tetrahedral cell
72 !to give the top half of five fold cluster.
73
74          CALL Lower(coord, Na, i, Ns)
75 !Mirror top half of particle to produce bottom half
76
77          CALL Join(coord, Na, i, Ns)
78 !Combine two half particles to produce complete particle
79
80       CASE (2:)
81 ! Case calling the subroutines needed to produce decahedral cluster
82 !only 2 subroutines are needed as the tetrahedral makes both the
83 !top and bottom halves of the cluster.
84          CALL Tetrahedron(Na, a, coord, i, st, Ns)
85
86          CALL Rotate(coord, Na, i, Ns)
87
88    END SELECT
89
90    CALL Cancel(coord, Na, a)
91 !Subroutine scans through the coordinate array for overlapping
92 !atoms and sets repeated atoms to a null value of 0,0,0 to avoid
93 !double counting of atoms which occurs as a result of rotating
94 !a section of the cluster to produce the whole thing.
95
96    CALL Align(coord, Na, i)
97
98    OPEN(unit=1, file='xyz')
99 !Initiates the output file 'xyz' in which the atomic coordinates
100 !will be recorded.
101    WRITE(1,*) '0.0000000000000000   0.0000000000000000   ↩
          0.0000000000000000'
102 !Write origin atom as first in output file. This is done outside
103 !of the main output loop below as the loop is constructed so that
```

```
104 !atoms as the origin are not recorded as this is the null value.
105
106    ip=i
107 !a parameter ip is needed as the upper limit of the output DO loop
108 !to follow. The atomic index, i, cannot be used as it will be altered
109 !in the loop when null atoms are not included in the output file.
110
111    Do q=2, ip
112 !Write all coordinates to output file
113
114       IF (coord(q,1) == 0.0000000000000000 .AND. coord(q,2) == ↩
              0.0000000000000000 .AND. coord(q,3) == 0.0000000000000000) ↩
              THEN
115 !IF statement needed to avoid writing null atomic coordinates to
116 ! output file.
117       i=i-1
118 !If the array contains a null coordinate it will not be recorded
119 !in the output file so the atomic index is reduced for each of
120 !these cases.
121       ELSE
122       WRITE(1,*) coord(q,1), coord(q,2), coord(q,3)
123 !For non-null atoms the coordinates are output to file xyz
124       END IF
125    END DO
126    PRINT*,'The cluster model has been produced. It contains',i, '↩
          atoms'
127 !Print statement indicating that the program has finished running.
128
129
130 END PROGRAM Clustersauce
131
132
133 SUBROUTINE Tetrahedron(Na, a, coord, i, st, Ns)
134    IMPLICIT NONE
135    INTEGER :: n=0, m=0, p=0, tr=0, tr2=0
136 ! n, m and p are coefficiants of the 3 primitive axes of the
137 !tetrahedral. They are used multiply the system along the
138 !crystallographic axes to produce the entire cluster. Tr and
139 !Tr2 are truncation values which are entered by the user in
140 !order to truncate a regular decahedra to produce ino or
141 !Marks decahedra.
142    INTEGER, INTENT(IN) :: Na, st, Ns
143    DOUBLE PRECISION, INTENT(IN) :: a
144    DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
145    INTEGER, INTENT(OUT) :: i
146    i=0
147
148    SELECT CASE (st)
149 !Case used to produce different tetrahedral structures depending
150 !on the required geometry.
151       CASE (:2)
152 !This case is the basic tetrahedral used for both the icosahedral
153 !and regular decahedral structures
154             DO p= 0, Ns
155 ! DO loops used to add multiple atoms along the tetrahedral axes
156                DO m= 0, (Ns-p)
157
```

```
158                 DO n= 0, (Ns-p-m)
159                       i=i+1
160 coord(i,1)= (n + 0.447*m + 0.449*p)*a
161 !'x' coordinates for all atoms in the do loop in terms of the
162 !tetrahedral axes.
163 coord(i,2)= (0.894*m + 0.278*p)*a
164 !'y' coordinates added to the main array.
165 coord(i,3)= (0.849*p)*a
166 !'z' coordinates.
167
168                    END DO
169                 END DO
170              END DO
171
172       CASE (3)
173 !Case for inodecahedral structures, same as above but
174 !introduces truncation.
175          PRINT*, 'Enter degree of truncation'
176 !User input to define the extent of truncation.
177          READ*, tr
178          DO p= 0, tr
179 !DO loop limited by truncation value.
180
181             DO m= 0, (tr-p)
182
183                DO n= 0, (Ns-p-m)
184                   i=i+1
185
186                   coord(i,1)= (n + 0.447*m + 0.449*p)*a
187                   coord(i,2)= (0.894*m + 0.278*p)*a
188                   coord(i,3)= (0.849*p)*a
189
190                END DO
191             END DO
192          END DO
193
194       CASE (4)
195 !Case for Marks decahedral structures, same as above but
196 !introduces further truncation.
197          PRINT*, 'Enter two degrees of truncation'
198 !User input to define two truncation variables.
199          READ*, tr, tr2
200
201
202          DO p= 0, tr
203
204             DO m= 0, (tr2-p)
205
206                DO n= 0, (Ns-p-m)
207                   i=i+1
208
209                   coord(i,1)= (n + 0.447*m + 0.449*p)*a
210                   coord(i,2)= (0.894*m + 0.278*p)*a
211                   coord(i,3)= (0.849*p)*a
212
213                END DO
214             END DO
```

```
215           END DO
216
217       END SELECT
218
219
220
221
222
223
224
225
226 END SUBROUTINE Tetrahedron
227
228 SUBROUTINE Mirror(coord, Na, i, Ns, a)
229 !Subroutine to mirror the tetrahedral structure about one of its
230 !facets to produce a double tetrahedral structure needed to model
231 !icosahedral geometry (This subroutine is not needed fo decahedra)
232       IMPLICIT NONE
233       INTEGER, INTENT(IN) :: Na, Ns
234       DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
235       INTEGER :: n, m, p
236 !n, m and p are coefficiants of the 3 primitive axes of
237 !the tetrahedral.
238       INTEGER, INTENT(INOUT) :: i
239       DOUBLE PRECISION, INTENT(IN) :: a
240
241       DO p= 0, Ns
242 !Do loop atoms to the tetrahedral to mirror it about one
243 !of the facets.
244
245          DO m= 0, (Ns-p)
246
247             DO n= 0, (Ns-p-m)
248                i=i+1
249                coord(i,1)= (0.5*n + 0.5*m - 0.445*p)*a
250 !Atomic positions of mirrored atoms calculated and
251 !stored in main array.
252                coord(i,2)= (0.265*n + 0.866*m + 0.724*p)*a
253                coord(i,3)= (0.824*n + 0.525*p)*a
254
255
256             END DO
257          END DO
258       END DO
259
260 END SUBROUTINE Mirror
261
262 SUBROUTINE Rotate(coord, Na, i, Ns)
263 !So far, one fifth of the five-fold symmetry of the cluster has been
264 !produced. This subroutine rotates that structure 4 times to give a
265 !full five-fold structure.
266 !(For icosahedral this is only the top half of the cluster).
267       IMPLICIT NONE
268       INTEGER, INTENT(IN) :: Na, Ns
269       DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
270       DOUBLE PRECISION, DIMENSION(3,3) :: R
271 !R is a rotation matrix used to rotate a five-fold section of the
```

```
!cluster about the x- axis.
   DOUBLE PRECISION :: shft=1.256637, SIN, COS
!shft is the angular shift needed to rotate a five-fold section of
!the cluster to add another section, it is ~72'
   INTEGER, INTENT(INOUT) :: i
   DOUBLE PRECISION, DIMENSION(3,1) :: M
!M is a temporary matrix which will contain the x, y and z
!coordinates of each atom as they pass through a DO loop.
!It is needed to facilitate matrix algebra
   INTEGER :: ifold, o, q
!ifold is the index of the last atom in the cluster so far.
!The cluster so far is one of the five-fold sections so
!this is to be rotated five times to produce the final
!cluster. ifold is needed to give the upper limit of the
!do loop in which each atom is rotated.


   ifold=i
!ifold is set to equal i, the last atom index so far.
!i could be used for this value, however, it couldn't then
!be used to index the new atoms added in the do loop as this
!would result in an infinite loop. It is easier to follow
!if 'i' is kept as the atom index throughout
!the program so a constant is needed for the loop bound.

   DO o=1, 4
!Do loop to give 4 different rotations of the five fold section.
      R(1,1)=1
!Define rotation matrix to rotate five-fold section about the x-axis
!to give the top half of the cluster.
      R(1,2)=0
      R(1,3)=0
      R(2,1)=0
      R(2,2)=COS(o*shft)
      R(2,3)=SIN(o*shft)
      R(3,1)=0
      R(3,2)=-SIN(o*shft)
      R(3,3)=COS(o*shft)

      Do q=1, ifold
!Do loop to perform rotation on all atoms in the first
! five-fold section

         i=i+1
!Continue updating the atom index with each iteration
         M(1,1)=coord(q,1)
!Defining M to be a matrix containing the x, y and z
!coordinates of each atom
         M(2,1)=coord(q,2)
         M(3,1)=coord(q,3)
         M=MATMUL(R,M)
!matrix multiplication to rotate atoms into next section
         coord(i,1)=M(1,1)
!Store rotated atom in main coordinate array.
         coord(i,2)=M(2,1)
         coord(i,3)=M(3,1)

```

```
329         END DO
330       END DO
331 END SUBROUTINE Rotate
332
333
334 SUBROUTINE Lower(coord, Na, i, Ns)
335 !DO loop needed for icosahedral structures. For these structures the
336 !previous routines only produce the top half of the cluster, this
337 !subroutine mirrors the top half to produce the bottom half.
338 !(Only used for icosahedra)
339     IMPLICIT NONE
340     INTEGER, INTENT(IN) :: Na, Ns
341     DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
342     DOUBLE PRECISION, DIMENSION(3,3) :: R
343     INTEGER, INTENT(INOUT) :: i
344     INTEGER :: ihalf, q
345     DOUBLE PRECISION, DIMENSION(3,1) :: M
346 !M is a temporary matrix which will contain the x, y
347 !and z coordinates of each atom as they pass through
348 ! a DO loop. It is needed to facilitate matrix algebra
349
350     R(1,1)=-1
351 !Define rotation matrix to rotate five-fold section
352 !about the x-axis to give the top half of the cluster.
353     R(1,2)=0
354     R(1,3)=0
355     R(2,1)=0
356     R(2,2)=1
357     R(2,3)=0
358     R(3,1)=0
359     R(3,2)=0
360     R(3,3)=1
361
362     ihalf=i
363
364     Do q=1, ihalf
365 !Do loop to perform rotation on all atoms in the first
366 !five-fold section
367
368           i=i+1
369 !Continue updating the atom index with each iteration
370           M(1,1)=coord(q,1)
371 !Defining M to be a matrix containing the x, y and z coordinates
372 !of each atom
373           M(2,1)=coord(q,2)
374           M(3,1)=coord(q,3)
375           M=MATMUL(R,M)
376 !matrix multiplication to rotate atoms into next section
377           coord(i,1)=M(1,1)
378 !Store rotated atom in main coordinate array.
379           coord(i,2)=M(2,1)
380           coord(i,3)=M(3,1)
381
382     END DO
383
384
385 END SUBROUTINE Lower
```

```
386
387 SUBROUTINE Join(coord, Na, Ns, i)
388 ! The upper half and lower half of the icosahedral structures have
389 !been produced but are mirror images of each over. This routine
390 !rotates the bottom half by 72'/2=36' so that the halves join
391 !correctly.
392    IMPLICIT NONE
393    INTEGER, INTENT(IN) :: Na, Ns
394    DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
395    DOUBLE PRECISION, DIMENSION(3,3) :: R
396    INTEGER, INTENT(INOUT) :: i
397    INTEGER :: ihalf, q
398 !ihalf is used to determine the point in the coord array at which the
399 !upper half of the cluster ends and the bottom half begins so that
400 !only the bottom half is included in the rotation DO loop.
401    DOUBLE PRECISION, DIMENSION(3,1) :: M
402 !M is a temporary matrix which will contain the x, y and z
403 !coordinates
404 !of each atom as they pass through a DO loop.
405 !It is needed to facilitate matrix algebra
406    DOUBLE PRECISION :: shft=0.628319, SIN, COS
407 !shft is ~36 in radians, the angle the bottom half of the cluster
408 !must be rotated by to fit the top half.
409
410    R(1,1)=1
411 !Define rotation matrix to rotate five-fold section about the
412 !x-axis to give the top half of the cluster.
413    R(1,2)=0
414    R(1,3)=0
415    R(2,1)=0
416    R(2,2)=COS(shft)
417    R(2,3)=SIN(shft)
418    R(3,1)=0
419    R(3,2)=-SIN(shft)
420    R(3,3)=COS(shft)
421
422    ihalf=i/2
423
424    Do q=ihalf, i
425 !Do loop to perform rotation on all atoms in the first
426 !five-fold section
427
428          M(1,1)=coord(q,1)
429 !Defining M to be a matrix containing the x, y and z
430 !coordinates of each atom
431          M(2,1)=coord(q,2)
432          M(3,1)=coord(q,3)
433          M=MATMUL(R,M)
434 !matrix multiplication to rotate atoms into next section
435          coord(q,1)=M(1,1)
436 !Store rotated atom in main coordinate array.
437          coord(q,2)=M(2,1)
438          coord(q,3)=M(3,1)
439
440    END DO
441 END SUBROUTINE Join
442
```

```
443 SUBROUTINE Cancel(coord, Na, a)
444 !The method of constructing a cluster from rotations of an
445 !initial tetrahedral results in overlapping atoms
446 !(double counting). This subroutine detects repeated atoms
447 !and gives them coordinates of 0,0,0 (i.e. the origin) so
448 !that they can be ignored when the output file is created.
449       IMPLICIT NONE
450       DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
451       DOUBLE PRECISION :: R
452 !Remanence when the squared coordinates of one atom are
453 !subtracted from those of another, effectively the atomic
454 !separation squared.
455 !This is used to determine whether any two atoms are overlapping.
456       DOUBLE PRECISION, INTENT(IN) :: a
457       INTEGER, INTENT(IN) :: Na
458       INTEGER :: k, h
459 !Integers used to perform DO loops over all atoms in the system.
460
461
462 louter: DO k= 2, 2*Na
463 !The outer DO loop cycles through each atom in the coordinate
464 !array so that its position can be compared with each over
465 !atom in the inner DO loop.
466       DO h=(k+1), 2*Na
467       R = (coord(k,1)-coord(h,1))**2 + (coord(k,2)-coord(h,2))**2 + (↩
              coord(k,3)-coord(h,3))**2
468 !calculation of the remanence between atoms h and k.
469
470       IF (R .LT. (a**2)*0.8) THEN
471 !IF statement used to select cases where atoms overlap so that
472 !their centres are closer than 0.9 times the normal
473 !atomic separation.(0.8=0.9 squared)
474
475          coord(k,1)=0.0000000000000000
476 ! If an atom overlaps another, the latter is given the null
477 !origin coordinates and will not be included in the output.
478          coord(k,2)=0.0000000000000000
479          coord(k,3)=0.0000000000000000
480
481          CYCLE louter
482 !If an atom has been found to overlap and has been set to the
483 !null value, no further comparisons are needed so the outer DO
484 !loop cycles to move on to the next atom for consideration.
485
486          ELSE
487 !In the case where no overlaps are found, the atomic coordinates
488 !remain stored in the coord array.
489          coord(k,1)=coord(k,1)
490          coord(k,2)=coord(k,2)
491          coord(k,3)=coord(k,3)
492       END IF
493       END DO
494 END DO louter
495
496
497
498 END SUBROUTINE Cancel
```

```
499
500 SUBROUTINE Align(coord, Na, i)
501 !This subroutine is used to align the primary axis of the cluster
502 !with the z-axis of the cartesian system for external convenience.
503    INTEGER, INTENT(IN) :: Na
504    DOUBLE PRECISION, INTENT(INOUT), DIMENSION(Na*6,3) :: coord
505    DOUBLE PRECISION, DIMENSION(3,3) :: R
506    INTEGER, INTENT(INOUT) :: i
507    INTEGER :: q
508    DOUBLE PRECISION, DIMENSION(3,1) :: M
509 !M is a temporary matrix which will contain the x, y and z
510 !coordinates of each atom as they pass through a DO loop.
511 !It is needed to facilitate matrix algebra
512    DOUBLE PRECISION :: shft=1.570796, SIN, COS
513 !shft is ~90' in radians, the angle cluster must be rotated about
514 !the y-axis to align the primary axis of the cluster.
515
516    R(1,1)=COS(shft)
517 !Define rotation matrix to rotate cluster about the y-axis
518 !to align primary cluster axis with z-axis.
519    R(1,2)=0
520    R(1,3)=SIN(shft)
521    R(2,1)=0
522    R(2,2)=1
523    R(2,3)=0
524    R(3,1)=-SIN(shft)
525    R(3,2)=0
526    R(3,3)=COS(shft)
527
528
529
530    Do q=1, i
531 !Do loop to perform rotation on all atoms in the first
532 !five-fold section
533
534          M(1,1)=coord(q,1)
535 !Defining M to be a matrix containing the x, y and z
536 !coordinates of each atom
537          M(2,1)=coord(q,2)
538          M(3,1)=coord(q,3)
539          M=MATMUL(R,M)
540 !matrix multiplication to rotate atoms into next section
541          coord(q,1)=M(1,1)
542 !Store rotated atom in main coordinate array.
543          coord(q,2)=M(2,1)
544          coord(q,3)=M(3,1)
545
546    END DO
547
548 END SUBROUTINE Align
```

4 Image analysis code

A script to be executed in Matlab to perform a quantitative analysis of zone-axis HAADF-STEM images.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%::''''''''''''''''''''  PEAKFINDER  ''''''''''''''''''':%%%%%%
 
% Script uses cross-correlation to determine peak positions
% Initial image data array should be named 'input'
% A Gaussian template is used for cross-correlation,
%parameters must be adjusted to approximate the Gaussians in the
%initial image in order to get good results.

%%%\_____ Gaussian parameters _____/%%%
%These parameters must be adjusted so that the gaussian template
%used in the cross correlation resembles those in the image.
%During cross-correlations the template will be normalized so the
%absolute intensity values are of less importance than the ratio
%of peak to background. It shouldn't matter which spot in the image
%the template is initially optimized to match, as long as all the
%spots are of reasonably similar shape. It is often useful to set
%the last few pixels of the image array equal to those of the gauss
%array so that the template is superimposed in the image array which
%can then be visually inspected to judge the quality of the template.
%To do this use:
% image((L-lg+1):L, (L-lg+1):L) = gauss(1:lg, 1:lg);

sptsz=2; %size of gaussian in angstroms.
theta=0; %rotate Gaussian for use with anisotropic peaks
origin=[0,0]; %DON'T CHANGE!
qgauss='y'; %y for gauss n for sinc fn
sl=0.1;

%%%\_____ Tif. image analysis or STEM output dat file _____/%%
dfile='y';  % y for dat file format n for image file format
model='y'; % y to correlate output with model structure file 'struct'

%%%\_____ Wiener filter parameters _____/%%%
% A Wiener filter is used to reduce noise allowing more reliable
%peak locating. The parameters should be optimized so that all peaks
%are found. This can be checked by plotting the array 'peakmap' which
%displays a map of the located peaks.

filter='n'; %Use wiener filter y/n
wienersig = 1.3; %Point spread function for use in Wiener filter
NSR=50; %noise-to-signal power ratio for Wiener filter


%%%\_____ Wiener filter parameters _____/%%%
% The cross-correlation produces an array of the same size as
%the input image. Each point in the array carries a value between
%-1 and 1 which is a measure of how well matched that point is to
```

```
%the centre of the template Gaussian. The quality factor below takes
%values between zero and one and is used to select the pixels in
%the array which match the template well. A higher value will accept
%only better matches

QF=0.4; %0.4Cross-correlation quality factor
Intoff=0.005; %Remove peaks with intensities less than this
%fraction of max of peaks.

%%%\_____ Intensity histogram _____/%%%
% After determining the location of the peaks and their intensities
%the frequency with which given intensities occur can be plotted,
%this can be used to find the relationship between intensity and
%number of atoms. In order to perform this analysis the intensity
%spectrum must be quantized into bins. If the number of bins is too
%large the intensity resolution will be high but the frequency of
%each intensity bin will be reduced

nbins=100;

%%%\_____ Model atom counting _____/%%%
% Determine number of atoms per column in the structure model and
%relate real-space coordinates to pixels in data out put so that
%column number and intensity can be examined visually.
imgx1=25;% Scan range of STEM simulation in Angstroms
imgx2=50;
imgy1=25;
imgy2=50;
sepmax=15; % Maximum separation within which atoms are considered
%to be in the same column (pixels).

%%%%%%::.....End of inpu Parameters.....::%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

imgsz= imgx2-imgx1;

if dfile == 'y'
L=sqrt(length(input));

for i=1:L
    for j=1:L


image(i,j)=input(L*(i-1)+j,3);
    end
end

elseif dfile == 'n'
      %image=im2double(imagein);
      L=length(image);
end


% Arrange input image to make it amenable to Matlab
%setup pointspreadfunction for filter
PSF = fspecial('gaussian', L,wienersig);

```

```
106
107 pixperang=L/imgsz;
108 atmsz=round(pixperang*sptsz);
109 gsize=[atmsz,atmsz];
110 sigmax=round(atmsz/3.5); %deviations in x & y axes
111 sigmay=round(atmsz/3.5);
112 peak=max(image);
113 peak=max(peak)-std(peak);
114 background=mean(image);
115 background=mean(background);
116
117 if qgauss == 'y'
118 gauss = customgauss(gsize, sigmax, sigmay, theta, background, peak, ↩
      origin);
119
120
121 elseif qgauss == 'n'
122   for j=1:atmsz
123
124     for i=1:atmsz
125         is=i*sl;
126         js=j*sl;
127         r=sqrt( ((atmsz/2*sl)-is)^2  +  ((atmsz/2*sl)-js)^2  );
128         gauss(j,i)=peak*sinc(r);
129
130     end
131   end
132
133 end
134 lg= length(gauss);
135 if filter=='y'
136 image= deconvwnr(image,PSF,NSR); %Performs Wiener filter
137 end
138 cc = normxcorr2(gauss,image);
139 pp=1; %initiate peaks output index
140
141 for i=1:L
142     for j=1:L
143         if cc(i,j) > QF
144
145             peaks(pp,1)=i;
146             peaks(pp,2)=j;
147             pp=pp+1;
148         end
149
150     end
151 end
152 peaks(:)=peaks(:)- (gsize(1,1)/2);
153
154 numpeaks=length(peaks);
155
156 %The cross-correlation will often return a number of pixels near the
157 %peak of the gaussian, the centre of mass of these will be taken as
158 %the true peak around which integrals will be performed. The
159 %positions of the centres of mass will be recorded in the peakcent
160 %array.
161
```

```
for q=1:numpeaks
    peakcent(q,1) = peaks(q,1);
    peakcent(q,2) = peaks(q,2);
    numpix=1;
    for h=1:numpeaks
        distx= (peaks(h,1)-peaks(q,1))^2;
        disty= (peaks(h,2)-peaks(q,2))^2;
        rsq=distx+disty;
        rsqmax= (gsize(1,1)/2)^2;
   if  0<rsq && rsq<rsqmax %This value should be high enough to find
     %single pixel centre of mass, if it leaves more than one pixel
     %per peak, increase rsqmax. This can be checked using fig. 1.

            peakcent(q,1) = peakcent(q,1)+ peaks(h,1);
            peakcent(q,2) = peakcent(q,2)+ peaks(h,2);
            numpix= numpix+1;
        end
    end
peakcent(q,1)= round( peakcent(q,1)/numpix );
peakcent(q,2)= round( peakcent(q,2)/numpix );
end

%For each peak in the peakcent array there will now be numpix
%repetitions of that value so to distill the peakcent array these
%will be removed using the unique function.

%N.B. THE OPERATION OF THE UNIQUE FUNCTION IS DIFFERENT FOR DIFFERENT
%MATLAB VERSIONS, THIS WORKS ON VERSION R2010B 7.11.0.584.

peakcent= unique(peakcent(:,:),'rows');


%Roving disk integration- integrate over a disk of area of 1/4
%atomic radius centred at various points near the position of each
%peak given by cross-correlation. The true peak is defined as the
%centre of the disc with highest average intensity.

 for q=1: length(peakcent) %for loop over each column

    peakcent(q,3)=0;
    for k= 0:atmsz/2; %for loop over COI1
        for l= 0:atmsz/2 %for loop over COI2
   COI=[round(peakcent(q,1)-atmsz/4+k),round(peakcent(q,2)-atmsz/4+l)↩
      ];
                N=0;
                int=0;
%for loop over scan region1
for i= COI(1,1)-2*atmsz : COI(1,1)+2*atmsz
%for loop over scan region2
for j= COI(1,2)-2*atmsz : COI(1,2)+2*atmsz

                    distx= (i-COI(1,1))^2;
                    disty= (j-COI(1,2))^2;
                    rsq=distx+disty;
                        if rsq < (atmsz) && i>0 && j>0
                        int=int+image(i,j);
                        N=N+1;
```

```
                        end
                    end %for loop over scan region2

                end %for loop over scan region1
                int=int/N;
                if int>peakcent(q,3)
                        peakcent(q,3)=int;
%true peak centre is centre of highest intensity ring
          TCOI=COI;
                end
            end %for loop over COI2
        end %for loop over COI1

        peakcent(q,1)=TCOI(1,1);
        peakcent(q,2)=TCOI(1,2);
    end %column loop

    peakf=max(peakcent(:,3));
    for q= length(peakcent) :-1:1
        if peakcent(q,3) < Intoff*peakf
            peakcent(q,:) = [];
        end
    end

peakcent= unique(peakcent(:,:),'rows');

%peakmap allows for a plot of the position of the peak centre
%pixels. The intensities of the input image will be integrated
%within a radius of the peak.
peakmap(1:L,1:L)=zeros;
    for h=1: length(peakcent)
          peakmap( peakcent(h,1)+1 , peakcent(h,2)+1 )= peakcent(h,3);

    end

% Convert model structure coordinates to pixels so that they can
%be aligned with image data NB: Not yet integer pixel values,
%just scaled.

if model == 'y'
    for h=2: (length(struct)-1);
fstruct( h, 1) = (struct( h, 2)-imgx1)* (L/imgsz) ;
fstruct( h, 2) = (struct( h, 3)-imgy1)* (L/imgsz) ;
    end


%Count number of atoms in column by finding number of atoms
%within sepmax of each other.
sepmaxsq= sepmax^2;
for q=1: length(fstruct);
          if (fstruct(q,1)==0)
              continue
          end
          fstruct(q,3)=1;
      for k=1: length(fstruct);
          sepsq= (fstruct(k,1) - fstruct(q,1))^2 + (fstruct(k,2) - ↩
              fstruct(q,2))^2;
```

```
274         if ( k~=q && sepsq <= sepmaxsq )
275             fstruct(q,3) = fstruct(q,3) +1;
276             fstruct(k,:)=0;
277         end
278     end
279
280 end
281 fstruct= unique(fstruct(:,:), 'rows');
282 %Round column coordinate values to integer pixels.
283 fstruct= round(fstruct);
284 end
285
286 %%%Voronoi integration
287 peakcent(:,5)=0;
288 peakcent(:,6)=0;
289
290
291
292 for i=1:length(image)
293     for j=1:length(image)
294
295 IDX = knnsearch([peakcent(:,1),peakcent(:,2)], [i,j] );
296     for q=1:length(IDX)
297        peakcent(IDX(q),5)=peakcent(IDX(q),5)+image(i,j);
298        peakcent(IDX(q),6)=peakcent(IDX(q),6)+1;
299     end
300     clear IDX
301
302     end
303 end
304
305
306 %%Output figures
307
308
309  imagesc(image);figure(1);
310  colormap(hot);
311  for h=1: length(peakcent)
312      text(peakcent(h,2),peakcent(h,1),[num2str(peakcent(h,5))],...
313   'VerticalAlignment','bottom',...
314    'HorizontalAlignment','center',...
315    'FontSize',24, 'color', [0,0,1])
316  end
317  if model == 'y'
318   for h=1: length(fstruct)
319      text(  (fstruct(h,1))   ,   (fstruct(h,2))   ,[num2str(fstruct(h,3)↩
          )],...
320   'VerticalAlignment','top',...
321    'HorizontalAlignment','center',...
322    'FontSize',24, 'color', [0,0.25,0])
323  end
324  end
325  axis equal
326  axis tight
327  hold on;
328  voronoi( peakcent(:,2), peakcent(:,1));
329 hold off;
```

```
330
331 figure(2);imagesc(peakmap);
332 % Analysing the peak intensity frequencies
333
334
335 [his,histx] = hist(peakcent(:,5), nbins);
336 figure; bar(histx, his, 'grouped');
337
338 peakcent= unique(peakcent(:,:),'rows');
339
340 %Model column - Image peak correlator to associate the peak
341 %intensities integrated from the image with the atom count
342 % from the model structure.
343 if model == 'y'
344 for j=1: length(peakcent)
345     proxmin=50000; %initialize proxmin to a suitably large value.
346         for k=1: length(fstruct)
347             prox= (fstruct(k,2) - peakcent(j,1))^2 + (fstruct(k,1) - ↩
                    peakcent(j,2))^2;
348         if prox < proxmin
349             proxmin=prox;
350             peakcent(j,4)=fstruct(k,3);
351         end
352     end
353     if proxmin > sepmax
354         disp('Error in image-model correlation')
355     end
356 end
357 end
358 if model == 'y'
359 lp=length(peakcent);
360 figure; scatter(peakcent(1:lp,4), peakcent(1:lp,5));
361
362 end
363 disp('Analysis complete')
364
365 clear COI
366 clear Intoff
367 clear L
368 clear N
369 clear NSR
370 clear PSF
371 clear QF
372 clear TCOI
373 clear atmsz
374 clear background
375 clear cc
376 clear dfile
377 clear distx
378 clear disty
379 clear filter
380 clear fstruct
381 clear gauss
382 clear gsize
383 clear h
384 clear his
385 clear histx
```

```
386 clear i
387 clear imgsz
388 clear imgx1
389 clear imgx2
390 clear imgy1
391 clear imgy2
392 clear int
393 clear j
394 clear k
395 clear l
396 clear lg
397 clear lp
398 clear model
399 clear nbins
400 clear numpeaks
401 clear numpix
402 clear origin
403 clear peak
404 clear peakf
405 clear peakmap
406 clear peaks
407 clear pixperang
408 clear pp
409 clear prox
410 clear proxmin
411 clear q
412 clear qgauss
413 clear rsq
414 clear rsqmax
415 clear sepmax
416 clear sepmaxsq
417 clear sepsq
418 clear sigmax
419 clear sigmay
420 clear sl
421 clear sptsz
422 clear theta
423 clear textdata
424 clear wienersig
425
426 % CUSTOMGAUSS
427 % Generate a custom 2D gaussian
428 %
429 %Function by Thomas Dideriksen 2006
430 %
431 %
432 %gauss = customgauss(gsize, sigmax, sigmay, theta, offset,
433 % factor, center)
434 %
435 %gsize      Size of the output 'gauss', should be a 1x2 vector
436 %sigmax     Std. dev. in the X direction
437 %sigmay     Std. dev. in the Y direction
438 %theta      Rotation in degrees
439 %offset     Minimum value in output
440 %factor     Related to maximum value of output, should be
441 % different from zero
442 % center     The center position of the gaussian, should be a
```

```
443 %1x2 vector
444 function ret = customgauss(gsize, sigmax, sigmay, theta, offset, ↩
        factor, center)
445 ret      = zeros(gsize);
446 rbegin   = -round(gsize(1) / 2);
447 cbegin   = -round(gsize(2) / 2);
448 for r=1:gsize(1)
449     for c=1:gsize(2)
450         ret(r,c) = rotgauss(rbegin+r,cbegin+c, theta, sigmax, sigmay,↩
                offset, factor, center);
451     end
452 end
453
454
455 function val = rotgauss(x, y, theta, sigmax, sigmay, offset, factor, ↩
        center)
456 xc       = center(1);
457 yc       = center(2);
458 theta    = (theta/180)*pi;
459 xm       = (x-xc)*cos(theta) - (y-yc)*sin(theta);
460 ym       = (x-xc)*sin(theta) + (y-yc)*cos(theta);
461 u        = (xm/sigmax)^2 + (ym/sigmay)^2;
462 val      = offset + factor*exp(-u/2);
```

Bibliography

[1] E. van Putten, D. Akbulut, J. Bertolotti, W. Vos, A. Lagendijk, and A. Mosk. Scattering lens resolves sub-100nm structures with visible light. *Physical Review Letters*, 106(19):1–4, 2011.

[2] L. D. Broglie. XXXV. A tentative theory of light quanta. *Philosophical Magazine Series 6*, 47(278):446–458, 1924.

[3] C. Davisson and L. Germer. Diffraction of electrons by a crystal of nickel. *Physical Review*, 30(6):705–740, 1927.

[4] Plücker. Ueber die einwirkung des magneten auf die elektrischen entladungen in verdnnten gasen. *Annalen der Physik*, 179(1):88–106, 1858.

[5] H. Busch. Zur schleiermacherschen methode der messung des wärmeleitvermögens von gasen. *Annalen der Physik*, 385(9):33–42, 1926.

[6] E. Ruska. Die elektronenmikroskopische abbildung elektronenbestrahlter oberflächen. *Zeitschrift für Physik*, 83(7-8):492–497, 1933.

[7] E. J. Kirkland. *Advanced computing in electron microscopy*. Springer US, Boston, MA, 2nd edition, 2010.

[8] M. Ardenne. Das elektronen-rastermikroskop. *Zeitschrift für Physik*, 109(9-10):553–572, 1938.

[9] A. V. Crewe, J. Wall, and J. Langmore. Visibility of single atoms. *Science*, 168(3937):1338–1340, 1970.

[10] Jeol. Jem-2100 electron microscope instructions. 2004.

[11] D. B. Williams and C. B. Carter. *Transmission electron microscopy*. Springer, New York, 2nd edition edition, 2009.

[12] E. Rutherford. The scattering of α and β particles by matter and the structure of the atom. *Philosophical Magazine*, 495(1909):1–24, 1911.

[13] N. F. Mott. The collision between two electrons. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 126(801):259–267, 1930.

[14] J. A. Ibers. Atomic scattering amplitudes for electrons. *Acta Crystallographica*, 11(3):178–183, 1958.

[15] A. Howie. Image contrast and localized signal selection techniques. *Journal of Microscopy*, 117(1):11–23, 1979.

[16] M. Haider, H. Müller, S. Uhlemann, J. Zach, U. Loebau, and R. Hoeschen. Prerequisites for a cc/cs-corrected ultrahigh-resolution TEM. *Ultramicroscopy*, 108(3):167–78, 2008.

[17] T. Sasaki, H. Sawada, F. Hosokawa, Y. Kohno, T. Tomita, T. Kaneyama, Y. Kondo, K. Kimoto, Y. Sato, and K. Suenaga. Performance of low-voltage STEM/TEM with delta corrector and cold field emission gun. *Journal of Electron Microscopy*, 59 Suppl 1:S7—-S13, 2010.

[18] C. Maunders, C. Dwyer, P. C. Tiemeijer, and J. Etheridge. Practical methods for the measurement of spatial coherence–a comparative study. *Ultramicroscopy*, 111(8):1437–46, 2011.

[19] J. LeBeau, A. DAlfonso, S. Findlay, S. Stemmer, and L. Allen. Quantitative comparisons of contrast in experimental and simulated bright-field scanning transmission electron microscopy images. *Physical Review B*, 80(17):1–5, 2009.

[20] J. Strutt. Xv. on the light from the sky, its polarization and colour. *Philosophical Magazine Series 4*, 41(271):107–120, 1871.

[21] O. Scherzer. Uber einige fehler von elektronenlinsen. *Physikalische Zeitschrift*, 101:593–603, 1936.

[22] O. Scherzer. Spharische und chromatische korrektur von elektronen-linsen. *Optik*, 2, 1947.

[23] R. Seeliger. Ein neues verfahren zur bestimmung des ffnungsfehlers von elektronenlinsen. *Optik*, 4, 1948.

[24] G. Möllenstedt and H. Düker. Beobachtungen und messungen an biprisma-interferenzen mit elektronenwellen. *Zeitschrift für Physik*, 145(3):377–397, 1956.

[25] J. Deltrap. Correction of spherical aberration of electron lenses. *PhD thesis, University of Cambridge*, 1964.

[26] J. Zach. Design of a high-resolution low-voltage scanning electron microscope. *Optik*, 83(1):30–40, 1989.

[27] M. Haider, H. Rose, S. Uhlemann, E. Schwan, B. Kablus, and K. Urban. A spherical-aberration-corrected 200kv transmission electron microscope. *Ultramicroscopy*, 75(1):53–60, 1998.

[28] O. L. Krivanek, N. Dellby, A. J. Spence, R. A. Camps, and L. M. Brown. Aberration correction in STEM. In J. M. Rodenburg, editor, *Electron Microscopy and Analysis 1997*, pages 35–40, Bristol, 1997. Institute of Physics.

[29] N. Dellby, O. L. Krivanek, P. D. Nellist, P. E. Batson, and A. R. Lupini. Progress in aberration-corrected scanning transmission electron microscopy. *Journal of Electron Microscopy*, 50(3):177–185, 2001.

[30] P. E. Batson, N. Dellby, and O. L. Krivanek. Sub-Ångstrom resolution using aberration corrected electron optics. *Nature*, 418(6898):617–620, 2002.

[31] D. E. Jesson and S. J. Pennycook. Incoherent imaging of crystals using thermally scattered electrons. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 449(1936):273–293, 1995.

[32] S. Hillyard and J. Silcox. Detector geometry, thermal diffuse scattering and strain effects in ADF STEM imagin. *Ultramicroscopy*, 58(1):6 – 17, 1995. Microscopy with Field Emission Electron Sources.

[33] J. LeBeau, S. Findlay, L. Allen, and S. Stemmer. Quantitative atomic resolution scanning transmission electron microscopy. *Physical Review Letters*, 100(20):1–4, 2008.

[34] N. Young, Z. Li, Y. Chen, S. Palomba, M. Di Vece, and R. Palmer. Weighing supported nanoparticles: Size-selected clusters as mass standards in nanometrology. *Physical Review Letters*, 101(24):28–31, 2008.

[35] Z. Y. Li, N. P. Young, M. Di Vece, S. Palomba, R. E. Palmer, A. L. Bleloch, B. C. Curley, R. L. Johnston, J. Jiang, and J. Yuan. Three-dimensional atomic-scale structure of size-selected gold nanoclusters. *Nature*, 451(7174):46–8, 2008.

[36] J. M. LeBeau, S. D. Findlay, L. J. Allen, and S. Stemmer. Standardless atom counting in scanning transmission electron microscopy. *Nano Letters*, 11(1):310–310, 2011.

[37] S. Van Aert, K. J. Batenburg, M. D. Rossell, R. Erni, and G. Van Tendeloo. Three-dimensional atomic imaging of crystalline nanoparticles. *Nature*, 470(7334):374–377, 2011.

[38] S. Bals, M. Casavola, M. A. van Huis, S. Van Aert, K. J. Batenburg, G. Van Tendeloo, and D. Vanmaekelbergh. Three-dimensional atomic imaging of colloidal core-shell nanocrystals. *Nano letters*, 11(8):3420–4, 2011.

[39] A. De Backer, G. T. Martinez, A. Rosenauer, and S. Van Aert. Atom counting in HAADF STEM using a statistical model-based approach: Methodology, possibilities, and inherent limitations. *Ultramicroscopy*, 134:23–33, 2013.

[40] Van Aert, S. and De Backer, A. and Martinez, G. T. and Goris, B. and Bals, S. and Van Tendeloo, G. and Rosenauer, A. Procedure to count atoms with trustworthy single-atom sensitivity. *Physical Review B*, 87(6):064107, 2013.

[41] D. G. Stroppa, R. D. Righetto, L. A. Montoro, L. Houben, J. Barthel, M. A. L. Cordeiro, E. R. Leite, W. Weng, C. J. Kiely, and A. J. Ramirez. Assessment of a nanocrystal 3-d morphology by the analysis of single HAADF-HRSTEM images. *Nanoscale Research Letters*, 8, 2013.

[42] S. Bals, S. Van Aert, C. P. Romero, K. Lauwaet, M. J. Van Bael, B. Schoeters, B. Partoens, E. Yücelen, P. Lievens, and G. Van Tendeloo. Atomic scale dynamics of ultrasmall germanium clusters. *Nature communications*, 3(may):897, 2012.

[43] S. Pennycook. Z-contrast stem for materials science. *Ultramicroscopy*, 30(1-2):58–69, 1989.

[44] P. M. Voyles, D. A. Muller, J. L. Grazul, P. H. Citrin, and H.-J. L. Gossmann. Atomic-scale imaging of individual dopant atoms and clusters in highly n-type bulk si. *Nature*, 416(6883):826–829, 2002.

[45] O. L. Krivanek, M. F. Chisholm, V. Nicolosi, T. J. Pennycook, G. J. Corbin, N. Dellby, M. F. Murfitt, C. S. Own, Z. S. Szilagyi, M. P. Oxley, and Others. Atom-by-atom structural and chemical analysis by annular dark-field electron microscopy. *Nature*, 464(7288):571–574, 2010.

[46] S. Bals, R. Kilaas, and C. Kisielowski. Nonlinear imaging using annular dark field tem. *Ultramicroscopy*, 104(3-4):281–9, 2005.

[47] D. Van Dyck. Persistent misconceptions about incoherence in electron microscopy. *Ultramicroscopy*, 111(7):894–900, 2011.

[48] R. F. Loane, P. Xu, and J. Silcox. Thermal vibrations in convergent-beam electron diffraction. *Acta Crystallographica Section A*, 47(3):267–278, 1991.

[49] P. Voyles, D. Chadi, P. Citrin, D. Muller, J. Grazul, P. Northrup, and H.-J. Gossmann. Evidence for a new class of defects in highly n-doped si: Donor-pair-vacancy-interstitial complexes. *Physical Review Letters*, 91(12):125505, 2003.

[50] V. Grillo. The effect of surface strain relaxation on haadf imaging. *Ultramicroscopy*, 109(12):1453–64, 2009.

[51] F. Fujimoto. Periodicity of crystal structure images in electron microscopy with crystal thickness. *physica status solidi (a)*, 45(1):99–106, 1978.

[52] K. Kambe. Visualization of bloch waves of high energy electrons in high resolution electron microscopy. *Ultramicroscopy*, 10(3):223–227, 1982.

[53] J. M. Cowley and A. F. Moodie. The scattering of electrons by atoms and crystals. i. a new theoretical approach. *Acta Crystallographica*, 10(10):609–619, 1957.

[54] J. Verbeeck, A. Béché, and W. Van den Broek. A holographic method to measure the source size broadening in STEM. *Ultramicroscopy*, 120:35–40, 2012.

[55] D. A. Muller, B. Edwards, E. J. Kirkland, and J. Silcox. Simulation of thermal diffuse scattering including a detailed phonon dispersion curve. *Ultramicroscopy*, 86(3-4):371–80, 2001.

[56] B. Forbes, A. Martin, S. Findlay, A. DAlfonso, and L. Allen. Quantum mechanical model for phonon excitation in electron diffraction and imaging using a born-oppenheimer approximation. *Physical Review B*, 82(10):1–8, 2010.

[57] A. J. Logsdail, Z. Y. Li, and R. L. Johnston. Development and optimization of a novel genetic algorithm for identifying nanoclusters from scanning transmission electron microscopy images. *Journal of computational chemistry*, 33(4):391–400, 2012.

[58] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–297, 1965.

[59] Nvidia. Cuda cufft library, 2007.

[60] U. Leonhardt. Optical metamaterials: Invisibility cup. *Nature Photonics*, 1(4):207–208, 2007.

[61] M. Faraday. Experimental relations of gold (and other metals) to light. *Philosophical Transactions of the Royal Society of London*, 147(1857):145–181, 1857.

[62] J. P. Wilcoxon and B. L. Abrams. Synthesis, structure and properties of metal nanoclusters. *Chemical Society reviews*, 35(11):1162–94, 2006.

[63] A. J. Logsdail. Computational characterisation of gold. *Springer theses*, 2013.

[64] F. Baletto, R. Ferrando, A. Fortunelli, F. Montalenti, and C. Mottet. Crossover among structural motifs in transition and noble-metal clusters. *The Journal of Chemical Physics*, 116(9):3856, 2002.

[65] T. P. Martini. Shells of atoms. *Physics Reports*, 273(4):199–241, 1996.

[66] L. D. Marks. Surface structure and energetics of multiply twinned particles. *Philosophical Magazine A*, 49(1):81–93, 1984.

[67] A. S. Barnard, N. P. Young, A. I. Kirkland, M. A. van Huis, and H. Xu. Nanogold: a quantitative phase map. *ACS Nano*, 3(6):1431–1436, 2009.

[68] K. Koga, T. Ikeshoji, and K.-i. Sugawara. Size- and temperature-dependent structural transitions in gold nanoparticles. *Physical Review Letters*, 92(11):2–5, 2004.

[69] C. L. Johnson, E. Snoeck, M. Ezcurdia, B. Rodríguez-González, I. Pastoriza-Santos, L. M. Liz-Marzán, and M. J. Hÿtch. Effects of elastic anisotropy on strain distributions in decahedral gold nanoparticles. *Nature materials*, 7(2):120–4, 2008.

[70] M. J. Walsh, K. Yoshida, A. Kuwabara, M. L. Pay, P. L. Gai, and E. D. Boyes. On the structural origin of the catalytic properties of inherently strained ultrasmall decahedral gold nanoparticles. *Nano Letters*, 12(4):2027–2031, 2012.

[71] P. Nachimuthu, W.-C. Shih, R.-S. Liu, L.-Y. Jang, and J.-M. Chen. The study of nanocrystalline cerium oxide by x-ray absorption spectroscopy. *Journal of Solid State Chemistry*, 149(2):408–413, 2000.

[72] L.-M. Peng, G. Ren, S. L. Dudarev, and M. J. Whelan. Debye-Waller factors and absorptive scattering factors of elemental crystals. *Acta Crystallographica Section A Foundations of Crystallography*, 52(3):456–470, 1996.

[73] N. Wang, S. I. Rokhlin, and D. F. Farson. Nonhomogeneous surface premelting of au nanoparticles. *Nanotechnology*, 19(41):415701, 2008.

[74] L. Lewis, P. Jensen, and J.-L. Barrat. Melting, freezing, and coalescence of gold nanoclusters. *Physical Review B*, 56(4):2248–2257, 1997.

[75] J.-H. Shim, B.-J. Lee, and Y. W. Cho. Thermal stability of unsupported gold nanoparticle: a molecular dynamics study. *Surface Science*, 512(3):262–268, 2002.

[76] Y. Wang, S. Teitel, and C. Dellago. Melting and equilibrium shape of icosahedral gold nanoparticles. *Chemical Physics Letters*, 394(4-6):257–261, 2004.

[77] T. Li, Y. Ji, S. Yu, and G. Wang. Melting properties of noble metal clusters. *Solid State Communications*, 116(10):547–550, 2000.

[78] Y. Wang, S. Teitel, and C. Dellago. Surface-driven bulk reorganization of gold nanorods. *Nano letters*, 5(11):2174–8, 2005.

[79] N. Y. Sdobnyakov, P. V. Komarov, D. N. Sokolov, and V. M. Samsonov. Study of the thermodynamic characteristics of gold nanoclusters using a gupta many body potential. *The Physics of Metals and Metallography*, 111(1):15–22, 2011.

[80] C. Feigl, G. Grochola, G. Opletal, I. Snook, and S. Russo. A theoretical study of size and temperature dependent morphology transformations in gold nanoparticles. *Chemical Physics Letters*, 474(1-3):115–118, 2009.

[81] P. Buffat. Electron diffraction and HRTEM studies of multiply-twinned structures and dynamical events in metal nanoparticles: facts and artefacts. *Materials Chemistry and Physics*, 81(2-3):368–375, 2003.

[82] M. Haruta. Size- and support-dependency in the catalysis of gold. *Catalysis Today*, 36(1):153–166, 1997.

[83] M. Valden. Onset of catalytic activity of gold clusters on titania with the appearance of nonmetallic properties. *Science*, 281(5383):1647–1650, 1998.

[84] G. Bond and D. Thompson. Gold-catalysed oxidation of carbon monoxide. *Gold Bulletin*, 33(2):41–50, 2000.

[85] X. Lai and D. Goodman. Structure-reactivity correlations for oxide-supported metal catalysts: new perspectives from STM. *Journal of Molecular Catalysis A: Chemical*, 162(1-2):33–50, 2000.

[86] N. Lopez. On the origin of the catalytic activity of gold nanoparticles for low-temperature CO oxidation. *Journal of Catalysis*, 223(1):232–235, 2004.

[87] M. Chen and D. Goodman. Structure-activity relationships in supported Au catalysts. *Catalysis Today*, 111(1-2):22–33, 2006.

[88] A. Herzing, C. J. Kiely, A. F. Carley, P. Landon, and G. J. Hutchings. Identification of active gold nanoclusters on iron oxide supports for CO oxidation. *Science*, 321(5894):1331–5, 2008.

[89] H. Häkkinen, S. Abbet, A. Sanchez, U. Heiz, and U. Landman. Structural, electronic, and impurity-doping effects in nanoscale chemistry: supported gold nanoclusters. *Angewandte Chemie (International ed. in English)*, 42(11):1297–300, 2003.

[90] Z. Yan, S. Chinta, A. Mohamed, J. P. Fackler, and D. W. Goodman. The role of F-centers in catalysis by Au supported on MgO. *Journal of the American Chemical Society*, 127(6):1604–5, 2005.

[91] M. Haruta and M. Daté. Advances in the catalysis of Au nanoparticles. *Applied Catalysis A: General*, 222(1-2):427–437, 2001.

[92] Y. Xu and M. Mavrikakis. Adsorption and dissociation of O_2 on gold surfaces: effect of steps and strain. *The Journal of Physical Chemistry B*, 107(35):9298–9307, 2003.

[93] B. R. Cuenya. Synthesis and catalytic properties of metal nanoparticles: Size, shape, support, composition, and oxidation state effects. *Thin Solid Films*, 518(12):3127–3150, 2010.

[94] F. Cleri and V. Rosato. Tight-binding potentials. *Physical Review B*, 48(1), 1993.

[95] J. C. Slater and G. F. Koster. Simplified LCAO method for the periodic potential problem. *Physical Review*, 94:1498–1524, 1954.

[96] I. T. Todorov, W. Smith, K. Trachenko, and M. T. Dove. Dl_poly_3: new dimensions in molecular dynamics simulations via massive parallelism. *Journal of Materials Chemistry*, 16(20):1911, 2006.

[97] J. Diao, K. Gall, and M. L. Dunn. Surface-stress-induced phase transformation in metal nanowires. *Nature materials*, 2(10):656–60, 2003.

[98] B. Curley, R. Johnston, N. Young, Z. Li, M. DiVece, R. Palmer, and A. Bleloch. Combining theory and experiment to characterize the atomic structures of surface-deposited Au309 clusters. *Journal of Physical Chemistry C*, 111(48):17846–17851, 2007.

[99] E. Aprà, F. Baletto, R. Ferrando, and A. Fortunelli. Amorphization mechanism of icosahedral metal nanoclusters. *Physical Review Letters*, 93(6):10–13, 2004.

[100] F. Baletto and R. Ferrando. Structural properties of nanoclusters: Energetic, thermodynamic, and kinetic effects. *Reviews of Modern Physics*, 77, 2005.

[101] H. E, K. E. Macarthur, T. J. Pennycook, E. Okunishi, A. J. D'Alfonso, N. R. Lugg, L. J. Allen, and P. D. Nellist. Probe integrated scattering cross sections in the analysis of atomic resolution HAADF STEM images. *Ultramicroscopy*, 133:109–19, 2013.

[102] J. Miller, A. Kropf, Y. Zha, J. Regalbuto, L. Delannoy, C. Louis, E. Bus, and J. Vanbokhoven. The effect of gold particle size on Au-Au bond length and reactivity toward oxygen in supported catalysts. *Journal of Catalysis*, 240(2):222–234, 2006.

[103] D. Klenov, S. Findlay, L. Allen, and S. Stemmer. Influence of orientation on the contrast of high-angle annular dark-field images of silicon. *Physical Review B*, 76(1):1–7, 2007.

[104] T. Plamann and M. J. Hÿtch. Tests on the validity of the atomic column approximation for STEM probe propagation. *Ultramicroscopy*, 78:153–161, 1999.

[105] E. Abe, S. J. Pennycook, and P. Tsai. Direct observation of a local thermal vibration anomaly in a quasicrystal. *Nature*, 421(6921):347–50, 2003.

[106] M. Haruta, H. Kurata, H. Komatsu, Y. Shimakawa, and S. Isoda. Effects of electron channeling in HAADF-STEM intensity in La_2CuSnO_6. *Ultramicroscopy*, 109(4):361–7, 2009.