# A Molecular Dynamics Study of Segregation and Diffusion in FCC Nanocrystals using the Sutton-Chen Potential

**By**

**Cornelia van der Walt**

**M.Sc**

*A thesis presented in fulfilment of the requirements of the degree*

**DOCTOR PHILOSOPHIAE**

**in the Department of Physics**

**at the University of the Free State**

**Republic of South Africa**

**Supervisor: J.J. Terblans**

**Co-supervisor: H.C. Swart**

February 2017

*To Andrew; it’s possible, it’s possible – anything is possible…*

# Acknowledgements

The author would like to express her deepest gratitude and sincere appreciation to the following people:

Funding from the **Nano Cluster** of the University of the Free State and the **NRF** is gratefully acknowledged. Without the financial support, this study would not have been possible.

To my husband, **Andrew T. Nash**, for endless hours listening to my troubles, staying up late just to keep me company, countless cups of tea and coffee, and boundless patience, understanding and support. Great thanks are also due to you for the help with setting up SQL databases to deal with the big data, patient explanations of software architecture, and help with formatting graphs and hours of “monkey work”. I couldn’t have done it without you!

To my mother, **Wilna**, for your concern, support, and little messages and pictures cards of encouragement. To my brother, **Marthinus** for your support and encouragement through the shared experience of struggling to complete a post-graduate project.

To **Mom Anne** and **Dad Roger**, for letting us stay with you for the final few months, and helping us through this difficult time with healthy meals, coffee early in the mornings, regular walks and stimulating talks. Special thanks are due to Mom Anne for help with reading my Greek and editing the grammar.

To **Tannie Belinda**, for letting us stay with you every visit to campus, for the lovely meals, braais, and gin and tonics. Thank you so much for giving us a home away from home.

Great thanks are due to my promoter, **Prof. J.J. (Koos) Terblans**, for hours and hours spent puzzling over the many challenges of this study. Thank you for making the time, even though you are busy, and always with a smile or a joke. I am amazed at your ability to come up with the most simple and elegant solutions, always reducing my complex ideas down to their core, and seeing through to the heart of the matter. Without your guidance, I would have been utterly lost. Thanks are also due to my co-promoter, **Prof. Hendrik C. Swart**, for his encouragement, support, and quips and jokes that sometimes stopped my heart.

Special thanks are owed to **Leon Wessels** for his help in implementing his modules in the program, advice in program errors, and valuable and stimulating discussion. It was a great help in overcoming the fear of undertaking something new and intimidating.

Thanks are due to the **staff** and fellow **students** of the Physics Department of the University of the Free State, for help, support, and friendliness. Discussions at tea time allowed me to return to my desk fresh and renewed.

Thanks, are also due to the **FSM**, I could only achieve this by Your powers combined.

# Keywords

Alloy
Aluminium
Berendsen thermostat
Binding energy
Cohesion energy
Copper
Diffusion
Diffusion activation energy
Diffusion mechanism
Dissipative particle dynamics
FCC metals
Gibbs free energy
Migration energy
Molecular dynamics
Nanocrystal
Nanocube
Nanoparticle
Nickel
Palladium
Platinum
Schottky defect
Segregation
Segregation energy
Silver
Surface orientation
Sutton-Chen potential
Temperature dependence
Vacancy formation energy
Velocity Verlet scheme
Verlet algorithm

# Abstract

Nanotechnology research has expanded notably, with a wide range of applications from catalysis in fuels, to optics. A key factor in manufacturing these particles is understanding diffusion and segregation of dopants and impurities in the nanocrystals, as segregation of these impurities influences which atom is exposed to the surface of the nano-particle, and able to react. Understanding these processes in terms of the shape and size of the particle, as well as the effects of temperature, are all important factors for nano-material manufacture.

Molecular Dynamics software is uniquely able to study the dynamics inside particles of up to several thousand atoms. The Sutton-Chen potential, in particular, is able to simulate the reactions of face-centred cubic (FCC) metals and model bulk modulus, elastic constants, lattice parameters, surface energies, phonon dispersion, cohesion energy and vacancy formation energy. It is ideally suited for studying the diffusion and segregation dynamics of the large clusters of atoms that make up nanocrystals.

In this study, a Molecular Dynamics model using the Sutton-Chen potential was built. This model implements the Verlet Velocity scheme to simulate the kinetics of the atoms, and uses the Berendsen thermostat to keep the system at a constant temperature. The model was tested on six FCC metals, namely Al, Ni, Cu, Pd, Ag and Pt, and, making use of periodic boundaries in order to simulate bulk crystals, calculated the cohesion energy to confirm the effectiveness of the model. The model further confirmed surface orientation dependence for low index surfaces. The relationship for vacancy formation energy of $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$ applied to all the FCC metals studied. The effects of temperature on other diffusion-related energies in the crystals were also studied. It was further found that the diffusion activation energy of FCC metals has the same relationship of $Q^{(110)} < Q^{(100)} < Q^{(111)}$.

Equipped with this information, the model was used for in-depth analysis of Cu, and later Ag, nano-cubes, -rhombicuboctahedrons and -octahedrons. A thorough analysis of the surface orientation dependence, size dependence, shape dependence and temperature dependence of key energies involved in diffusion, created a complete picture of nanoparticle stability and surface reactivity. It was found that larger particles are more stable, and that

surface reactivity indicates that nano-rhombicuboctahedrons are more reactive than perfect cubes, and that octahedrons are the least surface-reactive.

The final part of this study calculated the segregation energy in Ag/Cu systems to confirm the ability of the mixed Sutton-Chen potential to simulate segregation in alloys. In the Ag/Cu system, Ag is known to segregate to the surface, while Cu desegregates, and the model was able to demonstrate this. As this model can successfully reproduce that segregation, it can become a powerful tool for the study of diffusion dynamics in FCC alloy nano-materials.

# Samevatting

Nanotegnologie navorsing het geweldig uitgebrei, met 'n wye verskeidenheid van toepassings van katalise in brandstof, tot optika. 'n Belangrike faktor in die vervaardiging van hierdie deeltjies is die begrip van diffusie en segregasie van doteermiddels en onsuiwerhede in nano-kristalle, aangesien segregasie van hierdie onsuiwerhede beïnvloed watter atoom word blootgestel op die oppervlak van die nano-deeltjie, en in staat is om te reageer. Begrip van hierdie prosesse in terme van die vorm en grootte van die deeltjies, sowel as die effekte van temperatuur is almal belangrike faktore vir die vervaardiging van nano-materiale.

Molekulêre Dinamika sagteware is uniek in staat daarin om die dinamika binne deeltjies van tot 'n paar duisend atome te bestudeer. Die Sutton-Chen potensiaal, in besonder, is in staat om die reaksies, volumemodulus, elastisiteitskonstantes, roosterkonstantes, oppervlakenergie, fononverstrooiing, kohesie-energie en leemte-vormingsenergie van vlak-gesentreerde kubiese metale te modelleer. Dit is ideaal vir die bestudering van diffusie en segregasie dinamika van groot groepe atome waaruit die nano-kristalle bestaan.

In hierdie studie is 'n Molekulêre Dinamika model, wat van die Sutton-Chen potensiaal gebruik maak, ontwikkel. Hierdie model maak gebruik van die Verlet Snelheid skema om die kinetika van die atome te simuleer, en maak gebruik van die Berendsen termostaat om die sisteem teen 'n konstante temperatuur te hou. Die model is getoets op ses FCC metale, naamlik Al, Ni, Cu, Pd, Ag en Pt, en deur van periodiese-randvoorwaardes gebruik te maak om grootmaatkristalle te simuleer, die kohesie-energie is bereken om die doeltreffendheid van die model bevestig. Die model bevestig verder die oppervlak afhanklikheid vir lae-indeks oppervlaktes. Die verhouding vir leemte-vormingsenergie van $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$ is van toepassing op al die FCC metale wat bestudeer was. Die effek van temperatuur op ander diffusie-verwante energieë in die kristalle is ook bestudeer. Dit is verder bevind dat die diffusie-aktiveringsenergie vir FCC metale het dieselfde verhouding dat $Q^{(110)} < Q^{(100)} < Q^{(111)}$.

Met hierdie inligting is die model gebruik vir in-diepte analise van Cu, en later Ag, nano-kubussse, -rombiese-oktaëders en -oktaëders. 'n Deeglike ontleding van die oppervlakafhanklikheid, grootteafhanklikheid, vormafhanklikheid en temperatuurafhanklikheid van die energieë betrokke by diffusie en segregasie het 'n volledige beeld van nano-deeltjie se stabiliteit en oppervlak-reaktiwiteit gegee. Daar is bevind dat groter deeltjies meer stabiel is, en die oppervlak-reaktiwiteit getoon het dat nano-rombiese-oktaëders meer reaktief is as perfekte kubusse, en dat oktaëders die minste oppervlak-reaktief is.

In die laaste gedeelte van hierdie studie is die segregasie energie in 'n Ag/Cu sisteem bereken ten einde die vermoë te bevestig dat die gemengde Sutton-Chen potensiaal segregasie in legerings kan simuleer. In die Ag/Cu sisteem, is Ag bekend daarvoor dat dit segregeer na die oppervlak terwyl Cu de-segregeer, en die model was in staat om dit te demonstreer. Aangesien hierdie model suksesvol segregasie kan weergee, kan dit 'n kragtige instrument vir die studie van diffusie dinamika in FCC allooi nano-materiale word.

# Table of Contents

# Table of Figures

# Commonly Used Symbols, Acronyms, and Abbreviations

| | |
|---|---|
| $\Delta G$ | Change in Gibbs Free Energy; segregation energy. |
| $E_{coh}$ | Cohesion energy. |
| $E_{extr}$ | Extraction energy; binding energy of an atom in the bulk. |
| $E_{m}$ | Migration energy. |
| $E_{surf}$ | Surface binding energy; binding energy of an adatom on the crystal surface. |
| $E_{v}$ | Vacancy formation energy. |
| $Q$ | Diffusion activation energy. |
| Ag | Silver. |
| Al | Aluminium. |
| Cu | Copper. |
| Ni | Nickel. |
| Pd | Palladium. |
| Pt | Platinum. |
| DFT | Density Functional Theory. |

| | |
|---|---|
| FCC | Face-centred cubic. |
| HCP | Hexagonal close-packed. |
| LJ | Lennard-Jones. |
| MD | Molecular Dynamics. |
| MSM | Multi-Section Method. |
| NSC | Classical Sutton-Chen parameters. |
| PE | Potential energy. |
| QSC | Quantum Sutton-Chen parameters. |
| RTS | Rafii-Tabar Sutton parameters. |
| SC | Sutton-Chen. |

# Chapter 1 - Introduction

## 1.1 Rationale and Motivation

Nanocrystals can be described as any material with at least one dimension less than or equal to 100 nm that is a single crystal [1], such as those seen in Figure 1.1 [2]. Nanocrystals are massed atoms ranging from a few hundred to tens of thousands which form a "cluster". A typical nanocrystal is around ten nanometers in diameter and is larger than molecules but smaller than bulk solids and consequently exhibits physical and chemical properties somewhere between both. A nanocrystal is mostly surface with little volume and its properties can change noticeably as the crystal grows in size [3]. For instance, doped nanocrystals are a new class of materials whose quantum efficiency, a measure of the efficacy of wavelength absorption, increases with decreasing size of the particles [4]. In order to develop doped crystals for a variety of applications, it is also important to understand the properties and behaviour of the impurities in the semiconductor nanocrystals, such as the segregation and diffusion of the dopants.

Doped semiconductor nanocrystals have a range of uses, such as in video displays or windows as electrochromic materials [5], usage in materials requiring a well-defined optical absorption at mid-infrared wavelengths and possible application in high-density optical data storage [6]. With n-type doping the photoluminescence of doped semiconductor nanocrystals is quenched and this has applications in chemical synthesis for molecular biology. These materials have a variety of applications because of the range of semiconductor compositions available and the ability to tune their electronic and optical properties by changing particle size and the impurities with which they are doped.

The shape, such as corner and edge sharpness, or availability of corners, kinks and steps, can also increase catalytic productivity [7]. Further, the spectra of surface plasmon resonance (SPR) peaks are red-shifted by sharp corners of custom nanoparticle shapes, which give them a range of useful plasmonic properties [7, 8].

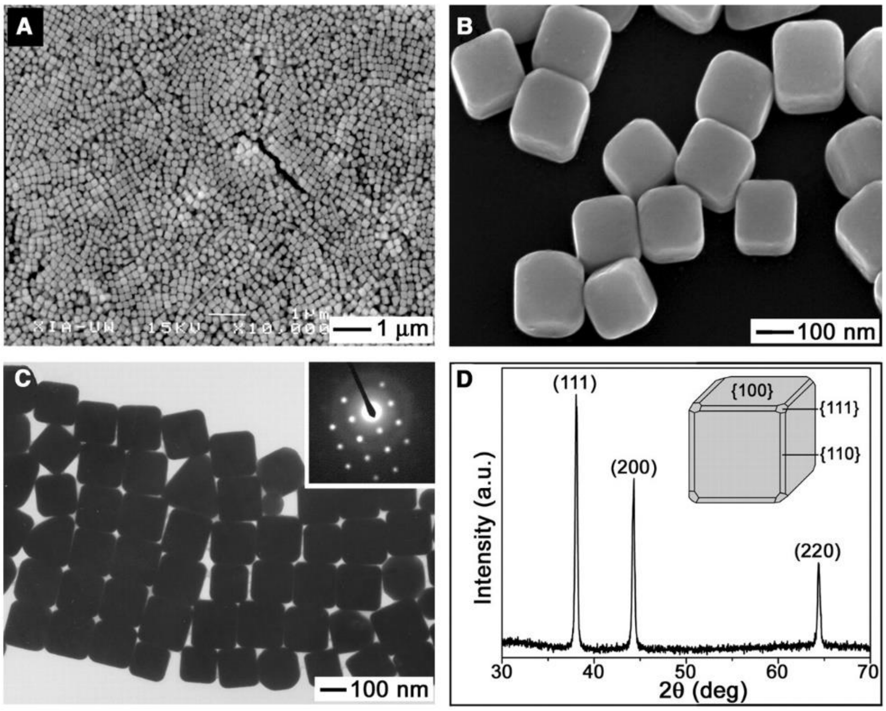


Figure 1.1 (a-d). (a) and (b) show SEM images of nano-crystals. These crystals are cubic in shape and slightly truncated. (c) shows a TEM image of the same crystals. (d) shows an XRD spectrum of the Ag nano-crystals, in arbitrary units [2].

In recent years, the search for energy-efficient fuels has spurred on research into nanoparticle catalysts [9, 10]. Extensive research is being done into the synthesis of nanoparticles of particular shapes and sizes. Size [11], shape [12], composition [13] and active sites [14] all affect nanoparticle properties [7], which are useful in various applications ranging from use as organic catalysis in cancer treatment [15], to enhancing the growth of more complex nanoparticle shapes [16]. For instance, Pt alloy nanoparticles are of interest because of their use as electrocatalysts in fuel cells. Figure 1.2 on the following page shows a variety of nanocrystals and their associated applications [17].

Manufacture of effective catalysts requires in-depth understanding of phase-segregation and alloying in bimetallic nanoparticles, including the effects of various factors such as phase-structures on the resulting catalytic activity [9]. An experimental study of Au-Pt, and density functional theory modelling, indicated a size and temperature dependence of phase segregation, indicating greater Au segregation at higher temperatures.

In another case, Cu-Au alloying allowed shape control and greater versatility in manufacturing catalysts [18], and the existence of Cu-Au nanocubes was first predicted by computational models. An analytical and computational study examined the effects of particle shape and size on segregation in Ag-Au, Cu-Au and Au-Pd nanoparticles [19]. In nanoparticles, the role of the bulk volume as an almost infinite sink or reservoir of atoms, and the shape dynamics of the nanoparticle influence dynamics. The Wulff model used in the study lacked the ability to predict particular atomistic details, but the study suggested that numerical models, like the Sutton-Chen potential, could elucidate such details.

Silver nanocubes have been synthesized as seen in Figure 1.1 [2]. Ag nanocrystals have particularly bright colloidal colours because of its emission spectra [7]. Cu nanocrystals are of particular use because of its excellent conductivity and catalytic properties [7, 20]. Wang et al. have reported synthesis of Cu nanoparticles with six (100) facets in perfect cubes [8]. An imperfect cube, or rhombicuboctahedron [21], the shape and construction of which is described in more detail later in this thesis, also displays (110) orientations on edges and (111) orientations on corners which are useful in synthesis of more complicated nanoparticle shapes.

Nanocubes are uniquely suited as building blocks in self-assembling structures [7, 18]. Nanocubes form large-scale regular lattices, and the shape of the final assembled structures can be controlled by application of hydrophilic and hydrophobic monolayers to the six faces. Nanocubes further provide information-rich characteristic hotspots for Raman scattering. These functions give shape controlled nanocrystals sensing applications as well. Nanocubes and truncated nanocubes can also be used in the manufacture of new complex nanoparticles, from 1-dimensional structures such as nanorods and nanorice, to more intricate structures such as branched nanocrystals and nanocages [8, 12]. Cuboctahedra can be used as seeds to grow octahedra with exposed {111} free surfaces [20]. It has been shown that the surface orientation of the particles have a significant effect on the reactivity of the catalyst, such as the aforementioned {111} faces of octahedra available for Heck coupling. Heck coupling is a reaction of an unsaturated halide with an alkene in the presence of a base and a palladium catalyst to form a substituted alkene.

The creation of noble metal nanocatalysts with base metal cores is also of interest to manufacturers for economic reasons. One method is to use a seed such as a base metal nanocube and to grow an epitaxial layer of noble metal on the activated sites [16]. Another possible method is to use surface segregation to grow a coating of the reactive metal on the surface. Another use of controlled segregation is the ability to change surface composition [22]. It has been shown that alloy composition of the surface can influence catalytic

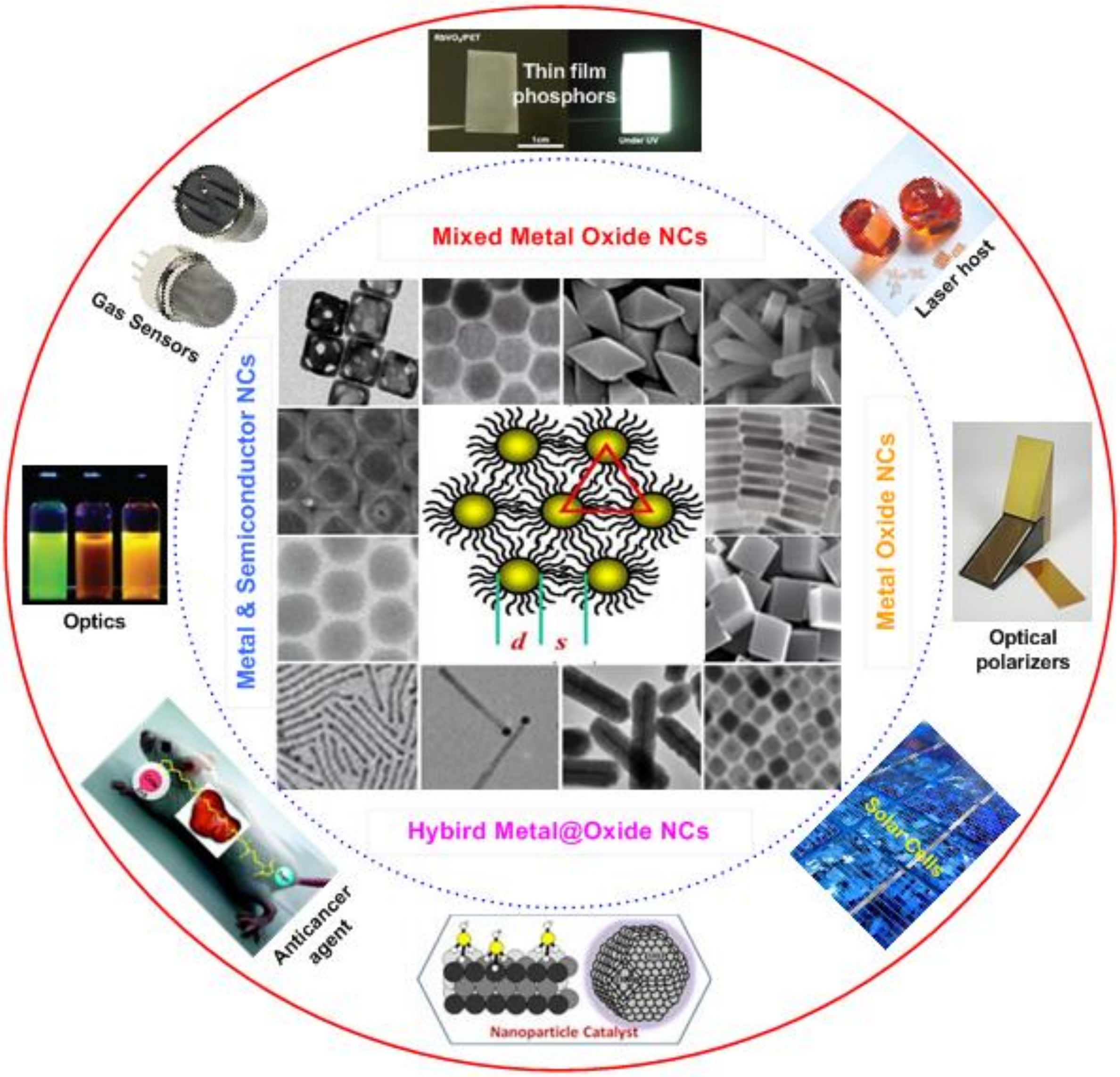


Figure 1.2 shows a representation of various shapes of nanocrystals that have been synthesized, and their possible applications in nano-systems for use in catalysis, optics, electronics and medicine [17].

reactivity, such as the composition of Cu-Pt surfaces in nanocubes used in electrocatalytic $CO_2$ reduction [13].

To develop doped crystals for a variety of applications it is also important to understand the properties and behaviour of the dopants in the semiconductor nanocrystals, which will be aided by the creation of a model describing diffusion and segregation in nanocrystals. To know which metal will segregate to the surface and be available for catalytic reaction, segregation in the system needs to be well understood. It is important to know whether diffusion and segregation of these metals used as catalysts show temperature dependence or surface orientation dependence, as this may provide vital information for effective catalyst manufacturing, characterization, and use.

Segregation refers to the enriching of a material with an impurity or dopant material at a free surface or an internal interface. Segregation sites can include surfaces, a dislocation, or a grain boundary [3]. Segregation to surfaces also has significant consequences involving the purity of samples. Some impurities have favourable segregation to the surface of the material resulting in a relatively small concentration of impurity in the bulk of the sample and significant coverage of the impurity on the cleaved surfaces. Absorption theories for the solid-solid interface and the solid-vacuum surface are directly comparable to well-known theories in the field of gas absorption on the free surfaces of solids [23]. Some previous studies have shown that segregation coefficients of impurities are found to depend on the growth conditions of the nanocrystals [24].

In this study, a model is constructed to explain the behaviour of impurities in nanomaterials. This will be achieved by utilizing Molecular Dynamics to learn more about the migration-, diffusion- and segregation energies for an atom migrating in the bulk and surface layers of nanocrystals.

## 1.2 Aim of this study

The project aims to create a model that describes diffusion and segregation in nanocrystals. To do this, the model must accurately describe the interactions between the atoms in the nanocrystal and the impurities. Interaction potentials and forces need to be calculated that accurately describe the system and a set of realistic initial conditions must be chosen. By means of solving the classical equations of motion, the behaviour of impurities can be predicted, such as whether the impurities remain in the bulk of the nanocrystal or segregate to the surface. The behaviours predicted by the model must be confirmed using experimental data obtained from literature.

In order to look at the properties and physical behaviour of impurities in nanocrystals, the activation barriers were considered, activation energies for vacancy formation and migration for dilute impurities were determined. These energies yield the diffusion rates of impurities by the mono-vacancy mechanism [25]. Diffusion involves the rate of change of the density of the diffusing substance at a fixed point in space. In nanocrystals with little bulk and mostly surface, boundary conditions need to be carefully regarded [26].

In nanocrystals, the surface to bulk ratio is significantly larger than in bulk crystals, increasing the significance of studying the effects of the surface on segregation. There are few studies, focused on bulk materials, that consider the impact of surface orientation on segregation energy [27]. A few models studied alloys using modelling software [28-35], often with the use of Monte Carlo modelling software, which is not able to model the dynamics of the system. The current study aims to thoroughly investigate diffusion to different surface orientations in not only bulk materials, but also nano-materials, using Molecular Dynamics simulations. The study will compare vacancy formation energies, an important diffusion parameter, for various face-centred cubic (FCC) metals, in bulk. It then aims to investigate in detail the effects of surface orientations, nano-crystal size, nano-crystal shape, and temperature on these same quantities in a particular metal, namely copper, before applying these insights to study the influence of several of these parameters on segregation in the Ag/Cu system.

The model used to quantify the above-mentioned properties was first developed using the Lennard-Jones potential, Verlet Velocity Scheme and a small number of atoms. The program was put together and tested with a few variables to ensure it runs smoothly. Once the model was able to make the necessary calculations, the Sutton Chen potential was substituted as the interaction potential, as it is more effective in predicting the properties of FCC metals [36]. Extensive calculations were done to determine the behaviour of the materials under various conditions, using first bulk materials and then nanomaterials. After each set of tests, the model was recalibrated or improved to further improve the ability of the calculated results to simulate the materials being studied. Finally, the properties of impurities in the nanocrystal were studied. The results were collected in a dissertation and published in papers.

If the model can simulate segregation in the Ag-Cu system, and describe in detail the segregation and diffusion dynamics in nanocrystals, the model can be applied to a range of mixed materials, and can study the in-depth segregation dynamics of other bimetallic systems which can be used in the manufacture of surface-coated catalytic nanoparticles. It could be useful in predicting the temperature, shape, size, and other necessary parameters for a desired result in nanoparticle manufacture. It may also yield further insights into diffusion and segregation of impurities in nanomaterials in order to better understand their effects on the properties of semiconductor nanocrystals and their possible applications in optics and catalysis.

## 1.3 Thesis layout

Presented in this section is a layout of the chapters in the thesis along with a short description of each chapter. In total the thesis consists of 8 chapters, a conclusion, plus 2 appendixes; appendix A and B.

#### 1.3.1.1 Chapter 2 -Diffusion and Segregation Theory

This chapter covers the theory of diffusion and segregation which is being investigated in this study.

#### 1.3.1.2 Chapter 3 -Molecular Dynamics – The Calculations

Chapter 3 looks at the calculations that the Molecular Dynamics model needs to do, including the algorithms used to calculate interatomic forces, positions, velocities and accelerations, and temperature control algorithms.

#### 1.3.1.3 Chapter 4 -The Code and Implementation

This section investigates the various challenges and pitfalls of implementing the above calculations, and the particular practical solutions implemented to overcome them during this study.

#### 1.3.1.4 Chapter 5 -Confirmation of the Model Using Bulk Crystals

This section presents the first results of this study; bulk calculations for a range of FCC metals which can be compared to experimental data to confirm the voracity of the working model. The model compares the cohesion energy and vacancy formation energy of Al, Ni, Cu, Pd, Ag and Pt to that of experimental values.

#### 1.3.1.5 Chapter 6 -Comparative Study of Cu Nanocubes at 0 K

Chapter 6 looks at Cu nano-materials and investigates the effects of particle size and shape on the parameters from the previous chapter. It compares the results to the bulk values as well as the experimental values from the previous chapters.

#### 1.3.1.6 Chapter 7 -Thermodynamics in Cu Nanocubes

Chapter 7 adds an important variable to the investigation begun in Chapter 6, by studying the properties of the Cu nanocrystals over a range of temperatures. It pays particular attention to surface energies, but also looks at melting behaviour and vacancy formation energy.

#### 1.3.1.7 Chapter 8 -Diffusion Dynamics in Pure Cu and Ag Crystals

This section adds migration energy measurements to the body of data on Cu, and investigates Ag as well, in preparation to compare the pure metal interactions to mixed interactions in the next chapter. With migration energy and a new metal included in the study, the Sutton-Chen parameters are also revisited and recalibrated.

#### 1.3.1.8 Chapter 9 -Diffusion and Segregation in Alloy Cu and Ag Crystals

The culmination of this study: Ag and Cu are mixed, and the various crystal properties are measured and investigated, in particular, the segregation energies of Cu in Ag, and Ag in Cu. The chapter ends with a long relaxation run with a mixed Cu and Ag nanoparticle, to see if the model will successfully model spontaneous segregation.

#### 1.3.1.9 Conclusion

This final chapter draws a conclusion outlining the results obtained in this study.

#### 1.3.1.10 Appendices

Appendix A: Computer code

Appendix B: Publications and conferences attended

## 1.4 References

[1] B. D. Fahlman, *Materials Chemistry*, (Mount Pleasant MI: Springer, 2007).

[2] Y. Sun, Y. Xia, *Science* **298** (2002): 2176.

[3] R. Kolb, (2001). “Nanocrystals: the shapes of things to come,” *Berkeley Lab research review*, accessed October 7, 2010, http://www.lbl.gov/Science-Articles/Research-Review/Magazine/2001/Fall/features/02Nanocrystals.html

[4] R. N. Bhargava, D. Gallagher, X. Hong and A. Nurmikko, *Physical Review Letters* **72** (1994): 3.

[5] “Doped Semiconductor Nanocrystals,” Kauffman Innovation Network, Inc, University of Chicago (2006). Accessed October 7, 2010. http://www.ibridgenetwork.org/uctech/doped-semiconductor-nanocrystals

[6] J. W. Chon, J. Moser, and M. Gu, “Use of Semiconductor Nanocrystals for Spectrally Encoded High-Density Optical Data Storage,” (International Symposium on Optical Memory and Optical Data Storage (ISOM/ODS), Honolulu, Hawaii, 2005)

[7] Y. Xia, Y. Xiong, B. Lim, and S. E. Skrabalak, *Angewandte Chemie International Edition* **48** (2009): 60.

[8] Y. H. Wang, P. L. Chen and M. H. Liu, *Nanotechnology* **17** (2006): 6000.

[9] B. N. Wanjala, J. Luo, R. Loukrakpam, B. Fang, D. Mott, P. N. Njoki, M. Engelhard, H. R. Naslund, J. K. Wu, L. Wang, O. Malis, and C. J. Zhong, *Chemistry of Matererials* **22** (2010): 4282.

[10] J. J. Terblans and G. N. van Wyk, *Radiation Effects and Defects in Solids* **156** (2006): 87.

[11] B. Li, R. Long, X. Zhong, Y. Bai, Z. Zhu, X. Zhang, M. Zhi, J. He, C. Wang, Z. Li and Y. Xiong, *Small* **8** (2012): 1710.

[12] R. Long, S. Zhou, B. J. Wiley and Y. Xiong, *Chemical Society Reviews* **43** (2014): 6288.

[13] X. Zhao, B. Luo, R. Long, C. Wang and Y. Xiong, *Journal of Materials Chemistry A* **3** (2015): 4134.

[14] S. Bai, X. Wang, C. Hu, M. Xie, J. Jiang and Y. Xiong, *Chemical Communications* **50** (2014): 6094.

[15] R. Long, K. Mao, X. Ye, W. Yan, Y. Huang, J. Wang, Y. Fu, X. Wang, X. Wu, Y. Xie, and Y. Xiong, *Journal of American Chemical Society* **135** (2013): 3200.

[16] Y. Bai, R. Long, C. Wang, M. Gong, Y. Li, H. Huang, H. Xu, Z. Li, M. Deng and Y. Xiong, *Journal of Materials Chemistry A* **1** (2013): 4228.

[17] T.-D. Nguyen and T.-O. Do, “Size- and Shape-Controlled Synthesis of Monodisperse Metal Oxide and Mixed Oxide Nanocrystals,” in Nanocrystal, Dr. Yoshitake Masuda, ed. (InTech, 2011). Accessed January 12, 2017. http://www.intechopen.com/books/nanocrystal/size-and-shape-controlled-synthesis-of-monodisperse-metal-oxide-and-mixed-oxide-nanocrystals

[18] Y. Liu and A. R. Hight Walker, *Angewandte Chemie International Edition* **49** (2010): 6781.

[19] L. Peng, E. Ringe, R. P. Van Duyne and L, D. Marks, *Physical Chemistry Chemical Physics* **17** (2015): 27940.

[20] R. Long, D. Wu, Y. Li, Y. Bai, C. Wang, L. Song and Y. Xiong, *Nano Research* **8** (2015): 2115.

[21] R. A. Molecke, “Characterization, modeling, and simulation of Multiscale directed-assembly systems” (PhD diss., The University of New Mexico, 2011).

[22] L. Rubinovich, M. I. Haftel, N. Bernstein, and M. Polak, *Physical Review B* **74** (2006): 035405.

[23] D. Briggs, M. P. Seah, eds., *Practical Surface Analysis by Auger and X-ray Photoelectron Spectroscopy*, (Chichester: John Wiley & Sons, 1983).

[24] H. Kodera, *Japanese Journal of Applied Physics* **2** (1963): 212.

[25] J. B. Adams, S. M. Foiles and W. G. Wolfer, *Journal of Materials Research* **4** (1989): 102.

[26] W. Rice, “Diffusion of impurities during epitaxy," paper presented in *Proceedings of the IEEE* **52** (1964): 284.

[27] J. Y. Wang, J. du Plessis, J. J. Terblans and G. N. van Wyk, *Surface and Interface Analysis* **28** (1999): 73.

[28] H. H. Kart, M. Tomak and T. Çağin, *Turkish Journal of Physics* **30** (2006): 311.

[29] G. Bozzolo, J. E. Garcés and G. N. Derry, *Surface Science* **601** (2007): 2038.

[30] A. Çoruh, Y. Saribek, M. Tomak, and T. Çağin, *AIP Conference Proceedings* **899** (2007): 243.

[31] S. Özdemir Kart, A. Erbay, H. Kılıç, T. Çağin and M. Tomak *Journal of Achievements in Materials and Manufacturing Engineering* **31** (2008): 41.

[32] C. H. Claassens, J. J. Terblans, M. J. H. Hoffman and H. C. Swart, *Surface and Interface Analysis* **37** (2005): 1021.

[33] G. Wang, M. A. Van Hove, P. N. Ross and M. I. Baskes, *Journal of Chemical Physics* **122** (2005): 024706.

[34] G. Wang, M. A. van Hove, P. N. Ross and M. I. Baskes, *Progress in Surface Science* **79** (2005): 28.

[35] J. Y. Wang, J. du Plessis, J. J. Terblans and G. N. van Wyk, *Surface and Interface Analysis* **28** (1999): 73.

[36] D. Cameron, (1992). “Implementation of Sutton-Chen Potential for Molecular Dynamics,” accessed February 6, 2011, http://www.genetical.com/dc/ScientificResearch/Rowland/MolecDynamED/SuttonChen/Potential.html.

# Chapter 2 - Diffusion and Segregation Theory

## 2.1 Segregation in Binary Alloys

Segregation has a great many applications. This chapter investigates the theory behind segregation, diffusion, and the particular method of atomic transport which will be modelled in the study.

Segregation in metal alloys is the separation of the constituent metals, such that surface enrichment of one of the metals causes the reduction of the Gibbs free energy of the system [1-3]. The minimizing of the Gibbs free energy is a driving force for many processes. When surface segregation occurs, atoms jump randomly from one site into another vacant one, but the rate at which the enriching metal atoms jump into the surface layer exceeds the rate at which that same metal's surface atoms jump into the bulk [4].

The total change in energy of surface segregation of a closed system in thermodynamic equilibrium can be expressed as

$$\Delta E = T\Delta S - P\Delta V + \Delta G, \tag{2.1}$$

where $T$ is the temperature, $S$ the entropy, $P$ the pressure and $V$ the volume of the system and the Gibbs free energy is given by $G$ [5, 6]. Thus, the Gibbs free energy depends on temperature and pressure. However, the Gibbs free energy in a system depends on the chemical bonds, as well as the phase of the system and the concentration of the constituent metals. Where pressure is not varied, the change in can also be expressed in terms of the entropy and enthalpy of the system [7, 8]:

$$\Delta G = \Delta H - T\Delta S. \tag{2.2}$$

$H$ is the internal energy of the system. The driving forces for spontaneous reactions lean in the direction of decreased internal enthalpy, $-\Delta H$, an increase in entropy, $+\Delta S$, and the release of heat, or exothermic, $+T$. In the case where a reaction decreases enthalpy, increases entropy, and is exothermic, $\Delta G$ is negative. Therefore, negative change in Gibbs free energy indicates a spontaneous reaction. Where a reaction is not exothermic, an increase in temperature may overcome the energy barrier and produce a negative change in Gibbs free energy to drive segregation.

In a system where the pressure and temperature remain constant:

$$\Delta E = \Delta G \tag{2.3}$$

from equation 2.1 which indicates that the change in energy of surface segregation can be expressed as a change in Gibbs free energy $G$ [5, 6]. The Gibbs free energy may be represented as a function of the chemical potential [5, 6, 9, 10]. This chemical potential is the energy per atom in the crystal. The difference in the chemical potential energy between the multi-layers can be considered as the cause of the driving force behind segregation [5, 11-13], or the energy cost of transferring the enriching metal to the surface [14, 15]. If the change in Gibbs free energy between layers of a crystal is negative, it would indicate a driving force for spontaneous segregation between those layers.

Being able to simulate surface segregation may be useful, in that measuring surface enrichment over time or temperature can allow the determination of other important variables, such as the diffusion factor $D_0$ or activation energy $Q$, which determine diffusion. Diffusion is discussed in the next section [16-22].

## 2.2 Diffusion

### 2.2.1 Fick's Law of Flux

Bulk diffusion limits segregation, so understanding bulk diffusion is important in measuring segregation [23]. Diffusion is often driven by a gradient of some sort, whether a concentration gradient, thermal energy gradient, stress gradient or electro-magnetic gradient. Fick's law of flux in steady-state crystals, which uses a change in concentration gradient to measure diffusion, gives the number of particles $\partial n$ over time through a unit surface area $A$ as

$$\partial n = -D \frac{\partial C}{\partial x} A \partial t, \tag{2.4}$$

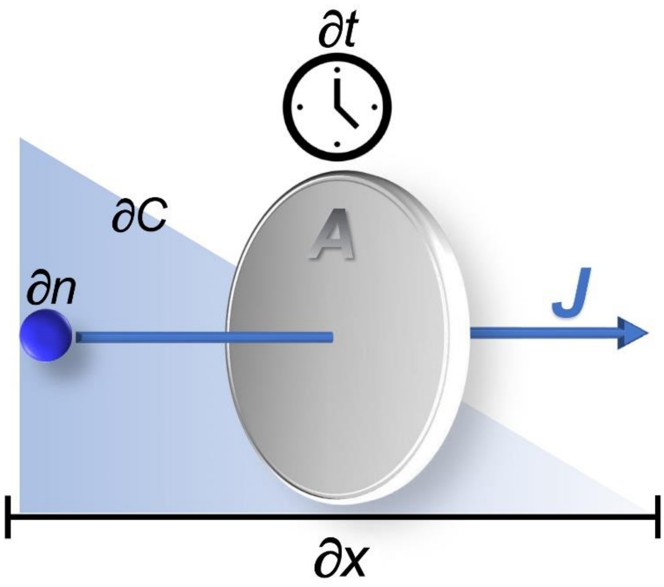


Figure 2.1. A diagram representing flux; the change in particles over time through an area is proportional to the concentration gradient which drives the flux.

with $D$ the diffusion coefficient and the concentration gradient given by

$$\frac{\partial C}{\partial x} = \left(C_2 - C_1 / x_2 - x_1\right). \tag{2.5}$$

Then flux $J$ is given by

$$J = -\frac{1}{A}\frac{\partial n}{\partial t} = -D\frac{\partial C}{dx}, \tag{2.6}$$

which is represented in Figure 2.1.

In non-steady state diffusion where the concentration gradient changes over time, Fick's law can be expressed as [24, 25]

$$\frac{\partial C}{\partial t} = \frac{\partial C}{\partial x}\left[D\left(\frac{\partial C}{\partial x}\right)\right]. \tag{2.7}$$

Where the diffusion coefficient $D$ is concentration independent:

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2}. \tag{2.8}$$

This relation has the solution of

$$C\left(x,t\right) = \text{const} - C_0 \text{erf}\left(\frac{x}{2\sqrt{Dt}}\right). \tag{2.9}$$

For further reading on a detailed derivation of the solution for the flux equation, refer to reference [24].

The coefficient of diffusion $D$ is famously given by the Arrhenius relationship

$$D = D_0 \exp\left( \frac{-Q}{k_B T} \right). \tag{2.10}$$

Where $Q$ is the activation energy for diffusion, $T$ is the temperature, $k_B$ is Boltzmann's constant and $D_0$ is the pre-exponential frequency factor [26-28].

### 2.2.2 Diffusion Mechanisms – the Vacancy Mechanism

Diffusion can occur in a variety of ways: there is volume diffusion, where there is bulk movement; surface diffusion along a free surface of a crystal; grain boundary diffusion; diffusion through interstitial crystal positions; and diffusion through the vacancy mechanism [27-29]. All crystals contain point defects, namely vacancies.

The vacancy mechanism is the most common method by which diffusion takes place, especially in metal FCC, BCC and HCP crystals, as there are always vacancies available, and the activation energy for the vacancy mechanism is less than that of the interstitial diffusion mechanism.

Using the probability term $P_v$, the number of vacancies in a crystal, at a given temperature, can be determined by [6, 25, 26, 28, 30]:

$$N_v = N_0 \exp\left( \frac{-E_v}{k_B T} \right), \tag{2.11}$$

where the activation energy for vacancy formation is $E_v$, $T$ is the temperature, $k_B$ is the Boltzmann's constant and $N_0$ is the number of lattice sites.

In the vacancy mechanism, the bulk diffusion coefficient is determined by two factors. The first factor is the probability that the diffusing atom has enough energy to cause the crystal matrix to deform as it moves to a new location. The second factor is the probability that the new location is vacant.

The activation energy of diffusion $Q$ can be deconstructed into two separate energies; the vacancy formation energy and the activation energy for atom migration $E_m$ [25, 28, 30, 31], where

$$Q = E_v + E_m \ . \tag{2.12}$$

These two factors are part of the diffusion equation [32]

$$D = D_0 \left[ \exp\left( \frac{-E_m}{k_B T} \right) \right]_{P_m} \left[ \exp\left( \frac{-E_v}{k_B T} \right) \right]_{P_v} , \tag{2.13}$$

where the $P_m$ term is the probability that an atom has the required energy to move from one lattice position to another. The $P_v$ term is the probability that there is a vacancy available to which the atom can move.

The vacancy formation energy is thus an important factor in atomic transport, and is used to describe diffusion kinetics in materials. The vacancy formation energy ( $E_v$ ) has been shown to be dependent on surface orientation in both Al and Cu [31, 33], but this has not been extensively studied as a function of temperature. In the following sections, the calculations used to determine the vacancy formation and migration activation energies will be discussed.

## 2.2.3 The Vacancy Formation Energy

Point defects are often referred to as Schottky defects, not to be confused with the anion-cation vacancy pair from ionic crystals [34]. In the case of a neutral vacancy in a metal crystal, the term serves to distinguish between a vacancy-adatom, and vacancy-interstitial defects, the Schottky defect being the one with an adatom.

The Schottky mechanism for vacancy formation is shown in Figure 2.2 where a simplified perfect FCC crystal whose only defect is the free surface represented by the top layer of atoms, forms a point defect. The Schottky mechanism is demonstrated here by atoms moving into open defect positions, such as the surface (b) or existing vacancies (c).

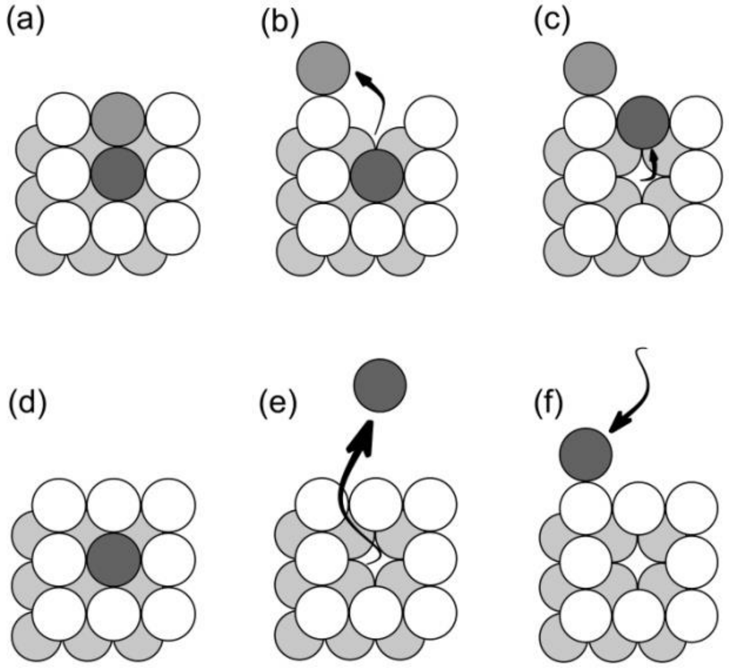


Figure 2.2 (a-f). The Schottky mechanism for vacancy formation. (a) and (d) show a simplified perfect FCC crystal containing only a surface defect at the top. The Schottky mechanism involves atoms moving into open defect positions, (b) and (c). A simplified version of the vacancy-adatom pair formation which only represents the initial (d) and final (f) crystal states approximates the process and allows effective calculations of the vacancy formation energy. Thus in (e) the atom is extracted straight from the middle of the crystal and deposited on the surface in (f).

The calculation of the vacancy formation energy ( $E_v$ ) for the Schottky defect has previously been simplified to the energy difference between the energy needed to extract an atom from inside the crystal bulk and the energy obtained from adding an atom to the crystal surface [31-33]. The vacancy formation energy can be calculated with:

$$E_v = E_{extr} - E_{surf} . \tag{2.14}$$

$E_{extr}$ is the energy needed for an atom to be removed from the bulk of a crystal to a position infinitely far away, and $E_{surf}$ is the energy gained for adding the extracted atom (adatom) on the surface. For simplification, these energies will be defined as the extraction energy $E_{extr}$ and the adatom energy $E_{surf}$ .

The extraction energy was evaluated by determining the potential energy of a perfect crystal ($U_{perfect}^{total}$), finding the potential energy for the crystal once a vacancy had been formed in the middle by extracting an atom ($U_{vacancy}^{total}$) and finding the difference:

$$E_{extr} = U_{perfect}^{total} - U_{vacancy}^{total}. \tag{2.15}$$

To find the binding energy for the adatom $E_{surf}$ the system energy was evaluated for the crystal with a vacancy in the centre, as an adatom was moved closer to the surface from infinity (see Figure 2.2 (f)).

This was done for all possible positions on the surface to find the most preferred binding positions. Preferred binding sites were chosen as the points where the potential energy of the system from adding the adatom to the surface was a minimum. Using likely bonding sites, the average surface-adatom binding energy ($U_{surf}^{total}$) for each crystal was determined and used to calculate the adatom energy:

$$E_{surf} = U_{surf}^{total} - U_{vacancy}^{total}. \tag{2.16}$$

Finally, the vacancy formation energy is calculated using equation 2.14.

### 2.2.4 The Migration Energy

There is activation energy needed for the motion of an atom from a lattice site to a point defect. In the process of migrating, the atom comes closer to other crystal atoms and needs to overcome repulsive Pauli forces.

The difference in energy between the state where the atom is in a lattice position, and when it is being maximally repelled by neighbouring atoms is the magnitude of the energy barrier that needs to be overcome. This energy barrier is the migration activation energy, represented in Figure 2.3. By measuring the total energy of a crystal as an atom is migrating from its original position to the neighbouring vacancy, the change in crystal energy can be used to determine the amplitude of this energy barrier. This activation energy barrier is the migration energy, $E_m$, and makes up the second part of the diffusion activation energy. The method for calculating the migration energy is explained in greater detail in section 4.8.

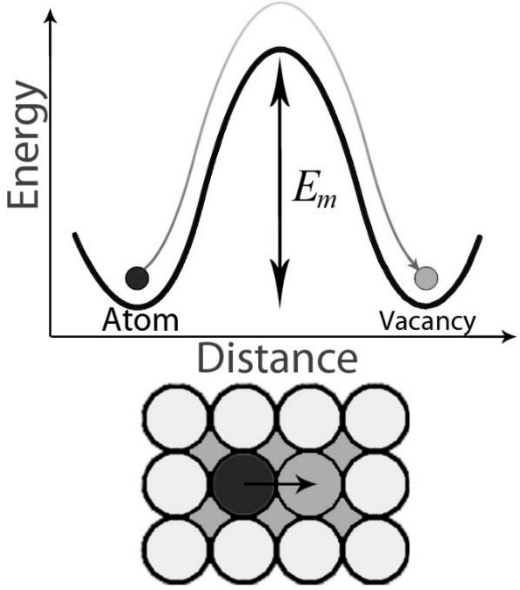


Figure 2.3. The migration energy is obtained from measuring the amplitude of the peak of change in energy as an atom moves from one lattice position into a vacancy.

## 2.3 Summary

The diffusion activation energy $Q$ limits diffusion and segregation, which consists of two parts: the vacancy formation energy $E_v$, and the migration energy $E_m$. A gradient in the crystal's free energy drives diffusion, and the diffusing atom migrates through an interstitial mechanism to an available vacancy. The energy needed to create such a Schottky vacancy in the crystal is $E_v$, which is created by an abbreviated two-step process. $E_m$ is obtained by noting the change in crystal energy when migrating the atom to the neighbouring vacancy. Using the Sutton-Chen potential, these energies can be measured by calculating the energy differences in the crystal directly when creating vacancies, and when migrating an atom to a vacancy.

In the next chapter, the various algorithms used to model atomic interactions and calculate energies are discussed, and in Chapter 4, the practical aspects of performing the calculations described in this chapter are more fully explored.

## 2.4 References

[1] P. A. Dowben, A. H. Miller and R.W. Vook, *Gold Bulletin* **20** (1987): 3.

[2] G. N. Lewis, M. Randall, K. S. Pitzer and L. Brewer, *Thermodynamics*, 2$^{nd}$ ed. (New York: McGraw-Hill, 1961).

[3] S. Hofman, *Scanning Electronic Microscopy* **3** (1985): 1071.

[4] H. D. Joubert, H. C. Swart and J. J. Terblans, *Surface and Interface Analysis* **37** (2005): 1027.

[5] J. Du Plessis, “Diffusion and Defect Data,” *Solid State Phenomena B*, **11** (Brookfield: Sci-Tech Publications, 1990).

[6] C. H. P. Lupis, *Chemical Thermodynamics of Materials* (Amsterdam: North-Holland, 1983).

[7] M. Polak, and L. Rubinovich, *Surface Science Reports* **38** (2000): 127.

[8] G. Klein, “Techniques for the Estimation and Prediction of Surface Segregation Occurring in Alloys,” (Ph.D diss., Carnegie Mellon University, 2011).

[9] G. Job and F. Hermann, *European Journal of Physics* **27** (2006): 353.

[10] S. Stølen and T. Grande, *Chemical Thermodynamics of Materials* (Chichester: John Wiley & Sons Ltd, 2004), 395.

[11] L. S. Darken, *Trans. Advances in Mechanical Engineering* **180** (1949): 430.

[12] H. Viefhaus and M. Rüsenberg, *Surface Science* **159** (1985): 1.

[13] J. K. Strohl, "Modeling alloy catalyst surfaces and bimetallic catalyst particles," *Retrospective Theses and Dissertations*, paper 9733 (1988).

[14] A. V. Ruben and H. L.Skriver, *Computation Materials Science* **15** (1999): 127.

[15] G. Bozzolo, J. Ferrante, R. D. Noebe, B. Good, F. S. Honecy and P. Abel, *Computation Materials Science* **15** (1999): 177.

[16] G. Dhanaraj, K. Byrappa, V. Prasad and M. Dudley, eds., *Springer Handbook of Crystal Growth*, 1st ed, (Berlin, Heidelberg: Springer, 2010).

[17] J. du Plessis, and G. N. van Wyk, *Journal of Physics and Chemistry of Solids* **50** (1988): 251.

[18] J. du Plessis, G. N. van Wyk, and E. Taglauer, *Surface Science* **220** (1989): 381.

[19] G. N. van Wyk, J. du Plessis, and E. Taglauer, *Surface Science* **254** (1991): 73.

[20] J. du Plessis, and P. E. Viljoen, *Surface Science* **276** (1992): 7.

[21] J. du Plessis, *Surface Science* **287-288** (1993): 857.

[22] E. C. Viljoen, J. du Plessis, H. C. Swart, and G. N. Van Wyk, *Surface Science* **342** (1995): 110.

[23] J. J. Terblans and G. N. van Wyk, *Radiation Effects and Defects in Solids* **156** (2006): 87.

[24] J. Crank, *The Mathematics of Diffusion*, 2nd ed (Oxford: Claredon Press, 1975).

[25] D. R. Askeland and P. P. Phulé, *The Science and Engineering of Materials*, 5th ed. (Toronto: Thomson, 2006).

[26] Y. Kraftmakher*, Physics Reports* **299** (1998): 79.

[27] P. Heitjans and J. Karger, eds., *Diffusion in Condensed Matter* (Berlin: Springer, 2005).

[28] R. J. Borg, G. J. Dienes, *Introduction to Solid State Diffusion*, 1st ed. (New York: Academic Press, 1988).

[29] P. G. Shewmon, *Diffusion in solids*, (New York: McGraw-Hill Book Company, 1963).

[30] M. E. Glicksman, *Diffusion in Solids: Field Theory, Solid State Principles, and Applications*, 1st ed. (NewYork: John Wiley & Sons, 2000).

[31] J. J. Terblans, *Surface and Interface Analysis* **33** (2002): 767.

[32] J. J. Terblans, W. J. Erasmus, E. C. Viljoen and J du Plessis, *Surface and Interface Analysis* **28**, (1999): 70.

[33] J. J. Terblans*, Surface and Interface Analysis* **35** (2003): 548.

[34] C. E. Housecroft and A. G. Sharpe, *Inorganic Chemistry*, 3rd ed. (Harlow: Pearson Prentice Hall, 2008).

# Chapter 3 - Molecular Dynamics – The Calculations

## 3.1 Introduction

The original simulation software was first developed using the Lennard-Jones potential, with a Velocity integrator and thermostat, before substituting the Lennard-Jones for the Sutton-Chen potential. Molecular Dynamics (MD) simulations can make use of a wide variety of algorithms, each with their own unique advantages and shortcomings.

## 3.2 Lennard-Jones Potential

### 3.2.1 Introduction

The Lennard-Jones potential (LJ) determines the potential between two or more uncharged atoms [1]. The LJ potential is best used for closed shell systems, such as with the rare gases Ar or Kr. LJ is not suitable for use with open shell systems such as in localized covalent bonds or in metals [2]. The LJ standard potential can be used for solids, liquids, surfaces and clusters amongst others. The lowest energy arrangement of an infinite number of atoms calculated by the LJ potential is hexagonal close packed (HCP), and by raising the temperature, simulated material first becomes cubic and then liquid [3]. It is ideal for looking at fundamental issues but is not suitable for describing the properties of a metal.

The LJ potential is written as:

$$\phi_{ij}^{LJ} = 4\varepsilon_{ij}^{LJ}\left[\left(\frac{\sigma_{ij}^{LJ}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}^{LJ}}{r_{ij}}\right)^{6}\right], \tag{3.1}$$

where $\phi_{ij}$ is the potential between particles $i$ and $j$, $\varepsilon$ is the depth of the energy well, $r$ is the distance between the particles and $\sigma$ is a finite distance at which the interparticle potential is zero [4]. The $r^{-12}$ term is strongly repulsive, representing the Pauli repulsion between particles at close distances due to the overlap of electron orbitals [1]. A more

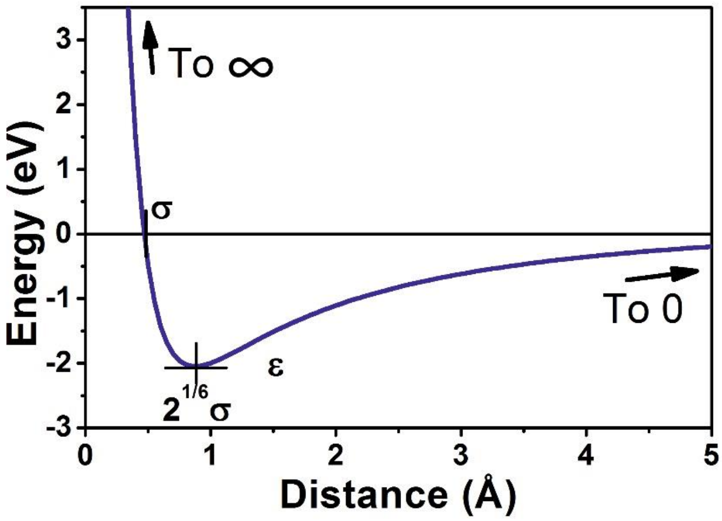


Figure 3.1. The force exerted by the molecules (or atoms) upon one another is the derivative of the potential energy, with respect to distance between them.

appropriate representation of the behaviour of the potential, at small distances of $r < 2^{1/6}\sigma$ is represented by an exponential term. Term $r^{-12}$ shows Pauli interactions at $r < 2^{1/6}\sigma$, and $r^{-9}$ terms can also be used [2, 5]. The $r^{-6}$ term describes a weakly attractive potential over larger distances of $r > 2^{1/6}\sigma$, due to the Van der Waals forces that arise from induced dipole – induced dipole interactions [1]. The LJ potential can be represented in a diagram as shown in Figure 3.1. The depth of the energy well is also indicated as $\varepsilon$ in the diagram [6].

For simplicity $2^{1/6}\sigma$ defined in Figure 3.1 can be named $r_m$ , and equation 3.1 can be simplified to:

$$\phi^{LJ} = \varepsilon\left[\left(\frac{r_m}{r}\right)^{12} - 2\left(\frac{r_m}{r}\right)^{6}\right]. \tag{3.2}$$

Its simple form is exactly what makes it ideal for creating new simulation software to test and refine before beginning more complicated calculations.

## 3.2.2 The Truncated LJ Potential

It is practical to consider a cut-off distance $R_c$ [7] where after particles are regarded to no longer interact, so that the LJ potential is given by:

$$V(r) = \begin{cases} \phi^{LJ}(r) - \phi^{LJ}(R_c) & \text{if } r \le R_c, \\ 0 & \text{if } r > R_c. \end{cases} \tag{3.3}$$

This saves computing power and prevents a jump discontinuity at $r = R_c$. The truncated part of the potential does affect the cohesion energy and pressure of the system, and in solids with periodic boundaries a small constant contribution can be made to compensate for this. In a system with free surfaces and lower symmetry the truncation effect may, however, be much larger. Potentials for metals and semiconductors are designed such that $\phi(R_c) = 0$, and at least for the first two derivatives: $\nabla\phi(R_c) = 0$ and $\nabla^2\phi(R_c) = 0$ [7].

Typically values of $2.5\sigma$ and $3.2\sigma$ are used for $R_c$ [7]. When calculating potentials in a simulation it is expedient to imagine that the system is contained in a box that has dimensions of at least $2R_c$ in every direction [8].

### 3.2.3 The Force Due to the LJ Potential

The force between two particles due to the LJ potential $\phi$ is represented by [1]

$$F = -\nabla\phi. \tag{3.4}$$

It can be written in terms of positions $\overline{r}_i$ and $\overline{r}_j$ of two atoms:

$$\overline{F}_{j\text{ on }i} = -\overline{F}_{i\text{ on }j} = 24\left(\overline{r}_i - \overline{r}_j\right)\left[2\left(\frac{1}{r}\right)^{-14} - \left(\frac{1}{r}\right)^{-8}\right], \tag{3.5}$$

where $r = \left|\overline{r}_i - \overline{r}_j\right|$.

### 3.2.4 Finding LJ Parameters

#### 3.2.4.1 Introduction

It is conventional to use $\varepsilon = 1$ and $\sigma = 1$ in computer simulations [2]. For real materials, however, $\varepsilon$ and $\sigma$ must be determined experimentally [9]. $\varepsilon$ is the depth of the potential well where the energy is less than $k_B T$ [10] and influences the freezing temperature of the system, and $\sigma$ influences the solid-state structure of the system [11]. Materials have standard $\varepsilon$ and $\sigma$ values, such as for water $\varepsilon = 0.6501696$ kJ/mol and $\sigma = 0.3165555$ nm [1]. The former is more often referred to as $\varepsilon / k_B$, which is expressed in units of temperature [10].

Table 3.1: The $\varepsilon$ and $\sigma$ values for selected elements [10, 12].

| **Material** | **H** | **F** | **Cl** | **Br** | **N** | **O** |
|---|---|---|---|---|---|---|
| $\varepsilon / k_B$ **(K)** | 8.6 | 52.8 | 173.5 | 257.2 | 37.3 | 44.6 |
| $\sigma$ **(nm)** | 0.281 | 0.283 | 0.335 | 0.354 | 0.331 | 0.295 |

Note in Table 3.1 that as one moves down the periodic table, $\varepsilon / k_B$ increases as the polarizability of the elements increase, while $\sigma$ decreases as one moves across the periodic table because as the nucleus charge increases, the electron shell becomes smaller. For atoms found in polymers $\sigma = \pm\ 0.3\,\text{nm}$ [10].

### 3.2.4.2 Calculating LJ parameters

The calculation of $\varepsilon$ and $\sigma$ obtained from the experimental sublimation energy and the lattice spacing at 0 K is obtained through a procedure followed by Brown in 1964 [13]. This evaluation takes into account both cubic and quartic contributions to the lattice free energy. The inclusion of the anharmonic contributions yield a better correlation between calculated and measured values.

The Lennard-Jones potential parameters are found by the simultaneous solution of equations for the lattice sublimation energy:

$$-L_0 = \Phi_0 + \Lambda E_0 + \Lambda^2 \left\{ F_3\left(0\right) + F_4\left(0\right) \right\}, \tag{3.6}$$

with $-L_0$ the lattice sublimation energy, $\Phi_0$ the static lattice energy, $\Lambda E_0$ the zero-point energy. The last terms represent the cubic and quartic contributions to the lattice sublimation energy. The total resulting contributions to lattice pressure is set equal to atmospheric pressure, such that:

$$-\rho_{\text{atm}} = \left( \frac{d\Phi_0}{dV} \right)_T + \Lambda \left( \frac{dE_0}{dV} \right)_T + \Lambda^2 \left\{ \frac{dF_3\left(0\right)}{dV} + \frac{dF_4\left(0\right)}{dV} \right\}. \tag{3.7}$$

Here

$$\Lambda = \frac{\hbar}{\sigma\left(M\varepsilon\right)^{\frac{1}{2}}},$$

$\hbar$ is Planck's constant and $M$ is the mass of a lattice atom. The static lattice energy is found from the expression

$$\Phi_0 = \frac{3m\varepsilon}{m-6}\left\{\frac{1}{m}\left(\frac{S_m^{000}}{z^m}\right)-\frac{1}{6}\left(\frac{S_6^{000}}{z^6}\right)\right\}, \tag{3.8}$$

where $S_m^{000}$ and $S_6^{000}$ are lattice sums and $z$ is the reduced intermolecular separation, with $z = b/\sigma$, where $b$ is half the edge length of the face-centred cubic (FCC) cell of the lattice. $m$ is an integer such that $0 \le m < K$, where $K$ is the total number of wave vector sampling points in the first Brillouin zone. The lattice sums discussed were taken for first, second and all nearest neighbour contributions. The lattice zero-point energy can be expressed as

$$\Lambda E_0 = 1.0223\left(\frac{4m}{m-6}\right)^{\frac{1}{2}}\frac{\varepsilon\Lambda}{z}\left\{(m-1)\frac{S_{m+2}^{000}}{z^m}+\frac{5S_8^{000}}{z^6}\right\}^{\frac{1}{2}}. \tag{3.9}$$

The experimental data needed is the values for $M$ , $b$ and $-L_0$ .

Now that both the $\Phi_0$ and $\Lambda E_0$ terms have been defined, only the anharmonic terms $\Lambda^2 F_3(0)$ and $\Lambda^2 F_4(0)$ remain to be determined. The reduced separation $x$ is defined as $x = z\sqrt{2} = b\sqrt{2}/\sigma.$ By using a first neighbour model, the reduced potential is defined in terms of $\phi(r)$ by

$$\phi(r) = \frac{6m\varepsilon}{m-6}\Psi(x), \tag{3.10}$$

where the potential $\Psi(x)$ is defined by

$$\Psi(x) = \frac{1}{mx^m} - \frac{1}{6x^6}. \tag{3.11}$$

The potential derivatives are defined by

$$\phi'(r) = \frac{1}{r}\frac{d\phi(r)}{dr}, \tag{3.12}$$

where, in general:

$$\phi^n(r) = \left(\frac{1}{r}\frac{d}{dr}\right)^n \phi(r). \tag{3.13}$$

Thus, the $n^{\text{th}}$ reduced derivative can be found from:

$$\phi^n(r) = \frac{6m\varepsilon}{m-6}(-1)^n \frac{\Psi_n(x)}{\sigma^{2n}}. \tag{3.14}$$

This can also be written as:

$$\Psi_n(x) = \frac{(m+2)(m+4)...(m+2n-2)}{x^{m+2n}} - \frac{8...(4+2n)}{x^{6+2n}}. \tag{3.15}$$

Reduced potential derivatives like these allow cubic and quartic terms to be written in a simpler form. Using first nearest neighbour treatment

$$\Lambda^2 F_3(0) = -\frac{9}{2}\varepsilon\Lambda^2 \left\{ x^2 S_1 \left(\frac{\Psi_3}{\Psi_2}\right)^2 - 4S_2\left(\frac{\Psi_3}{\Psi_2}\right) + \frac{4S_3}{x^2}\right\}, \tag{3.16}$$

and

$$\Lambda^2 F_4(0) = \frac{27}{4}\varepsilon\Lambda^2 \left\{ x^2 S_4 \left(\frac{\Psi_4}{\Psi_2}\right) - 2S_5\left(\frac{\Psi_3}{\Psi_2}\right) + \frac{4S_6}{x^2}\right\}. \tag{3.17}$$

Terms $S_1 - S_6$ are functions of volume since they depend on $\rho$, where:

$$\rho = -\frac{1}{x^2}\left(\frac{\Psi_1}{\Psi_2}\right),$$

which involves the slope of the intermolecular potential at the equilibrium lattice separation. The cubic and quartic terms in equation 3.7 involve volume derivatives of $\Lambda^2 F_3(0)$ and $\Lambda^2 F_4(0)$ and volume derivatives of the lattice sums obtained by differentiating polynomials found from cubic polynomials fit to anharmonic lattice sums. The sublimation equation and the pressure equation are solved iteratively on a computer using values of lattice spacing $b$ and sublimation energy $-L_0$ obtained experimentally.

## 3.3 Sutton-Chen Potential

Once the Lennard-Jones potential has been used to resolve computational errors and difficulties, it is replaced by the Sutton-Chen potential, which can model material properties in FCC metals.

### 3.3.1 Introduction

The Sutton-Chen potential uses particle densities in FCC metals and can determine pairwise distributions. The Sutton-Chen (SC) empirical many-body potential is a long-range Finnis-Sinclair potential, using inverse powers for both attractive and repulsive terms [14]. The inverse powers give rise to a van der Waals interaction at long range, which is an important interaction in noble metals. The same term that describes the van der Waals interaction at long range also describes the short range unsaturated covalent bond. The SC potential is able to simulate FCC metals with better accuracy and is ideal for calculating the vacancy formation energy.

### 3.3.2 The Long Range Finnis-Sinclair Potential

The Sutton–Chen potential ($U_i$) has an embedded many-body term that uses particle densities in FCC metals to determine the interactions between atoms [14]. The total energy of the simulated crystal ($U$) is calculated using

$$U = \sum_i U_i = \varepsilon \sum_i \left[ \tfrac{1}{2} \sum_{i \neq j} V\left(r_{ij}\right) - c\sqrt{\rho_i} \right], \tag{3.18}$$

with $\varepsilon$ an energy parameter and $c$ a dimensionless scaling parameter. $V$ is the pairwise repulsive potential and $\rho_i$ is a density-like bonding term [15];

$$V\left(r_{ij}\right) = \left( \frac{a}{r_{ij}} \right)^n , \tag{3.19}$$

$$\rho_i = \sum_{i \neq j} \phi\left(r_{ij}\right) = \sum_{i \neq j} \left( \frac{a}{r_{ij}} \right)^m , \tag{3.20}$$

where $r_{ij}$ is the distance between atoms $i$ and $j$, $a$ is the lattice constant, $n$ a positive integer that determines the repulsive potential and $m$ is a positive integer that determines the range

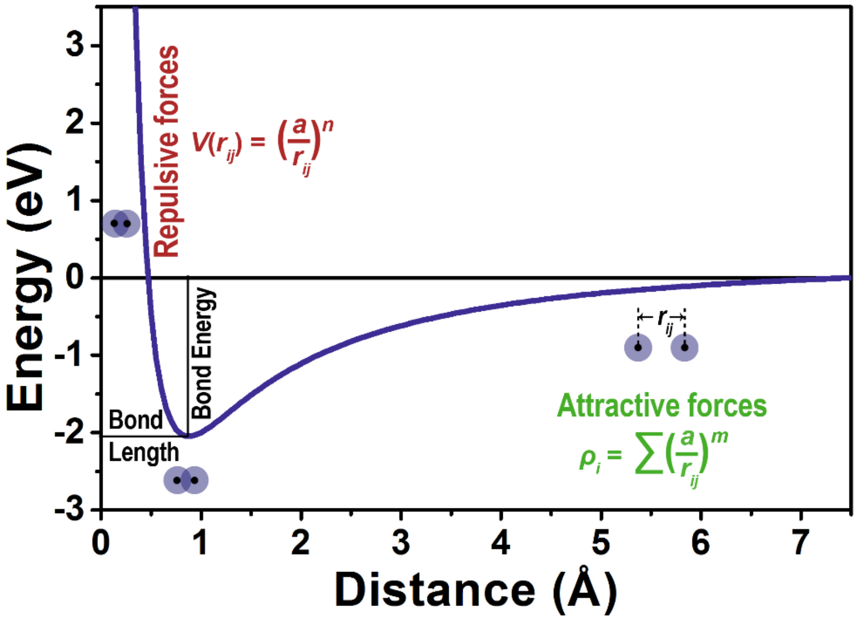


Figure 3.2. The Sutton-Chen potential has two terms, one modelling the Pauli repulsive forces, as shown in red, the other the weak Van der Waals attractive forces in green. The strength of the attractive forces and repulsive forces depend on the distance between nuclei of the atoms, and where the attractive force is a maximum, the atoms bond together.

Table 3.2 The Sutton–Chen parameters for Al, Ni, Cu, Pd, Ag, and Pt [14].

| **Element** | *ε* **(eV)** | *a* **(Å)** | *c* | *n* | *m* |
|---|---|---|---|---|---|
| **Al** | 3.3147 $10^{-2}$ | 4.05 | 16.399 | 7 | 6 |
| **Ni** | 1.5707 $10^{-2}$ | 3.52 | 39.432 | 9 | 6 |
| **Cu** | 1.2382 $10^{-2}$ | 3.61 | 39.432 | 9 | 6 |
| **Pd** | 4.1790 $10^{-3}$ | 3.89 | 108.27 | 12 | 7 |
| **Ag** | 2.5415 $10^{-3}$ | 4.09 | 144.41 | 12 | 6 |
| **Pt** | 1.9833 $10^{-2}$ | 3.92 | 34.408 | 10 | 8 |

of the attractive potential such that $n > m$. A representation of the potential is shown in Figure 3.2.

In order to determine the Sutton-Chen potential for a particular material, the constants $n$ , $m$ and $a$ need to be known [16]. The parameters used were published by Sutton and Chen [14], and are listed in Table 3.2. The strongly repulsive pairwise potential term in the short-range acts on atoms that approach each other too closely and are repelled. The weakly attractive density-like term simulates the interactions of metals with an electron cloud by calculating the contributions of surrounding atoms to the interactive potential. This density-like attractive term is long range.

### 3.3.3 Using the SC Potential to Find the Cohesive Energy

Using the SC potential, the potential energy from atomic interactions can be evaluated. This potential energy evaluated for perfect crystals ($U_{perfect}^{total}$) can be used to calculate the cohesion energy as follows [17]:

$$E_{coh} = (U_{r=\infty}^{total} - U_{perfect}^{total}) / n, \tag{3.21}$$

where $U_{perfect}^{total}$ is the total potential energy of the perfect crystal and $U_{r=\infty}^{total}$ is the total potential energy of the system where the $n$ atoms are infinitely far removed from each other, in ground state and at 0 K. When using the SC potential to evaluate system energies, it is important to note that the potential energies are entirely due to atom interactions, and that the total system energy with all atoms infinitely far removed is 0 eV. Thus equation 3.21 can be simplified to

$$E_{coh} = U_{perfect}^{total} / n. \tag{3.22}$$

### 3.3.4 The Force Due to the SC Potential

The total force can be calculated from the SC potential in the following way:

$$\bar{f}_i = -\bar{\nabla}_i U. \tag{3.23}$$

This can be reduced to:

$$\bar{f}_i = \sum_{i \neq j} f_{ij} \frac{\bar{r}_{ij}}{r_{ij}}, \tag{3.24}$$

where $f_{ij}$ is the force between atoms $i$ and $j$, and can be expressed as:

$$f_{ij} = -\varepsilon \left[ n \left( \frac{a}{r_{ij}} \right)^n - \frac{cm}{2} \left( \frac{1}{\sqrt{\rho_i}} + \frac{1}{\sqrt{\rho_j}} \right) \left( \frac{a}{r_{ij}} \right)^m \right] \left( \frac{1}{r_{ij}} \right). \tag{3.25}$$

Here $\varepsilon$ has a dimension of energy, $c$ is dimensionless, $a$ is the FCC lattice constant, and $n$ and $m$ are positive integers so that $n > m$ [16, 18, 19].

### 3.3.5 Parameters for Mixed Potentials

Feraoun et al. have developed a new many body potential inspired from the SC potential, with a new approach to modelling the cross interaction for alloys [20]. The potential was obtained by expressing the Hamiltonian (total energy in terms of potential and kinetic energies) in a Finnis-Sinclair potential form [21]:

$$\mathcal{H} = \tfrac{1}{2}\left[\sum_{i\neq j}\sum \hat{p}_i\hat{p}_j V^{AA}\left(r_{ij}\right) + \left(1-\hat{p}_i\right)\left(1-\hat{p}_j\right)V^{BB}\left(r_{ij}\right) + \left[\hat{p}_i\left(1-\hat{p}_j\right)+\hat{p}_j\left(1-\hat{p}_i\right)\right]V^{AB}\left(r_{ij}\right)\right]$$
$$-d^{AA}\sum_i \hat{p}_i\left[\sum_{i\neq j}\hat{p}_j\phi^{AA}\left(r_{ij}\right)+\left(1-\hat{p}_j\right)\phi^{AB}\left(r_{ij}\right)\right]^{1/2}$$

$$-d^{BB}\sum_i (1-\hat{p}_i)\left[\sum_{i\neq j}\left(1-\hat{p}_j\right)\phi^{BB}\left(r_{ij}\right)+\hat{p}_j\phi^{AB}\left(r_{ij}\right)\right]^{1/2}. \qquad (3.26)$$

Here $\hat{p}_i$ is a site occupancy operator and is defined as follows:

$$\hat{p}_i = \begin{cases} 1, \text{ if site } i \text{ is occupied by an } A \text{ atom,} \\ 0, \text{ if site } i \text{ is occupied by a } B \text{ atom.} \end{cases} \qquad (3.27)$$

Functions $V^{AA}$, $V^{BB}$, $V^{AB}$, $\phi^{AA}$, $\phi^{BB}$ and $\phi^{AB}$ are defined as follows:

$$V^{AA}(r) = \varepsilon^{AA}\left[\frac{a^{AA}}{r}\right]^{n^{AA}}, V^{BB}(r) = \varepsilon^{BB}\left[\frac{a^{BB}}{r}\right]^{n^{BB}}, V^{AB}(r) = \varepsilon^{AB}\left[\frac{a^{AB}}{r}\right]^{n^{AB}}, \qquad (3.28)$$

$$\phi^{AA}(r) = \left[\frac{a^{AA}}{r}\right]^{m^{AA}}, \quad \phi^{BB}(r) = \left[\frac{a^{BB}}{r}\right]^{m^{BB}}, \quad \phi^{AB}(r) = \left[\frac{a^{AB}}{r}\right]^{m^{AB}}. \qquad (3.29)$$

The constants $d^{AA}$ and $d^{BB}$ are defined as

$$d^{AA} = \varepsilon^{AA}c^{AA} \text{ and } d^{BB} = \varepsilon^{BB}c^{BB}. \qquad (3.30)$$

In equations 3.28 to 3.30 the constants $a^{AA}$, $c^{AA}$, $\varepsilon^{AA}$, $m^{AA}$ and $n^{AA}$ denote the corresponding constants for the pure metal $A$, and similarly the constants $a^{BB}$, $c^{BB}$, $\varepsilon^{BB}$, $m^{BB}$ and $n^{BB}$ denote the corresponding constants for the pure metal $B$. These values can all be obtained from the parameters for pure metals as defined by Sutton and Chen [14]. The only parameters that remain to be determined are the constants $a^{AB}$, $\varepsilon^{AB}$, $m^{AB}$ and $n^{AB}$. This may be done if the following is assumed:

$$V^{AB} = (V^{AA}V^{BB})^{1/2} \tag{3.31}$$

and

$$\phi^{AB} = (\phi^{AA}\phi^{BB})^{1/2}. \tag{3.32}$$

This assumption leads to the following expressions that allow the unknown parameters to be calculated from the parameters of the pure metals:

$$\begin{aligned} m^{AB} &= \tfrac{1}{2}(m^{AA} + m^{BB}), \\ n^{AB} &= \tfrac{1}{2}(n^{AA} + n^{BB}), \\ a^{AB} &= (a^{AA}a^{BB})^{1/2}, \\ \varepsilon^{AB} &= (\varepsilon^{AA}\varepsilon^{BB})^{1/2}. \end{aligned} \tag{3.33}$$

These mixed potentials were employed by Jin et al. [22] to determine the melting behaviour of cubo-octahedron $\text{Al}^{4033}$ nanoclusters with Pb cores. The potentials were modified slightly to describe the immiscibility of the alloy system. With the provision that equation 3.33 is entirely empirical, $m^{AB}$ and $n^{AB}$ could be adjusted to reflect the immiscible properties of the alloy, by making $n^{AB}$ slightly larger than the mean of $n^{AA}$ and $n^{BB}$:

$$n^{AB} = \tfrac{1}{2}\left(n^{AA} + n^{BB}\right) + \delta, \tag{3.34}$$

where $\delta$ is chosen for the alloy. The expression used to determine the total energy for the alloy was:

$$U_i^{AB} = \tfrac{1}{2}\varepsilon^{AB}\sum_{i \neq j} V^{AB}\left(r_{ij}\right) + \varepsilon^{AA}c^{AA}\left(\rho_i\right)^{1/2}, \tag{3.35}$$

or

$$U_i^{BA} = \tfrac{1}{2}\varepsilon^{BA}\sum_{i \neq j} V^{BA}\left(r_{ij}\right) + \varepsilon^{BB}c^{BB}\left(\rho_i\right)^{1/2}, \tag{3.36}$$

where

$$V^{AB}(r)=V^{BA}(r)=\left(\frac{a^{AB}}{r}\right)^{n^{AB}} \quad \text{and} \quad \rho_i=\sum_{i\neq j}\left(\frac{a^{AB}}{r_{ij}}\right)^{m^{AB}} . \tag{3.37}$$

Rodriguez-Lopez et al. used the mixed potentials to establish other thermodynamic behaviours of a system where the melting points were previously determined [23]. No adjustments were made to equation 3.33.

### 3.3.6 Forces for Mixed Potentials

From equation 3.24 where $f_{ij}$ is the force between atoms $i$ and $j$, it can be expressed as [24]:

$$\begin{aligned}\bar{f}_{ij} &= -\varepsilon\left[\sum_{ij}\left(1-\delta_{ij}\right)\bar{\nabla}_i\left(\frac{a}{r_{ij}}\right)^{n} - c\sum_{i}\bar{\nabla}_i\sqrt{\sum_{j}\left(1-\delta_{ij}\right)\left(\frac{a}{r_{ij}}\right)^{m}}\right] \\ &= -\varepsilon\sum_{j\neq i}\left[n\left(\frac{a}{r_{ij}}\right)^{n} - \frac{cm}{2}\left(\frac{1}{\sqrt{\rho_i}}+\frac{1}{\sqrt{\rho_j}}\right)\left(\frac{a}{r_{ij}}\right)^{m}\right]\left(\frac{\bar{r}_{ij}}{r_{ij}^{2}}\right).\end{aligned} \tag{3.38}$$

However, in equation 3.35 and 3.36 it can be seen that $\varepsilon$ is not a common factor in binary alloys. Thus, calculating the force in binary alloys would be changed to:

$$f_{ij}^{AB} = -\sum_{i\neq j}\left[\varepsilon^{AB}n^{AB}\left(\frac{a^{AB}}{r_{ij}}\right)^{n^{AB}} - \frac{\varepsilon^{AA}c^{AA}m^{AB}}{2}\left(\frac{1}{\sqrt{\rho_i}}+\frac{1}{\sqrt{\rho_j}}\right)\left(\frac{a^{AB}}{r_{ij}}\right)^{m^{AB}}\right]\left(\frac{1}{r_{ij}}\right). \tag{3.39}$$

This is the formula currently used to calculate the forces in the simulation used in this study.

However, in a PhD study conducted by S. Özdemir Kart [25] the force is calculated as follows:

$$f_{ij}^{AB} = -\varepsilon\left[\sum_{i\neq j}\frac{\partial V(r_{ij})}{\partial r_{ij}} - \frac{c^{AA}}{2}\frac{\sum_{i\neq j}\frac{\partial\phi(r_{ij})}{\partial r_{ij}}}{\rho_i} + \frac{c^{AA}}{2\rho_j}\sum_{i\neq j}\frac{\partial\phi(r_{ij})}{\partial r_{ij}}\right]\left(\frac{1}{r_{ij}}\right), \tag{3.40}$$

which can be simplified for comparison to:

$$f_{ij}^{AB} = -\varepsilon^{AB}\left[ n^{AB}\sum_{i\neq j}\left(\frac{a^{AB}}{r_{ij}}\right)^{n^{AB}} - \frac{c^{AA}m^{AB}}{2}\left(\frac{1}{\sqrt{\rho_i}} - \frac{1}{\sqrt{\rho_j}}\right)\sum_{i\neq j}\left(\frac{a^{AB}}{r_{ij}}\right)^{m^{AB}}\right]\left(\frac{1}{r_{ij}}\right). \quad (3.41)$$

Another simple direct derivation from equation 3.35 will yield:

$$\begin{aligned} f_i^{AB} &= \tfrac{1}{2}\varepsilon^{AB}\sum_{i\neq j}\frac{\partial}{\partial r_{ij}}V^{AB}\left(r_{ij}\right) + \varepsilon^{AA}c^{AA}\frac{\partial}{\partial r_{ij}}\left(\rho_i^{1/2}\right) \\ &= \tfrac{1}{2}\left[\varepsilon^{AB}n^{AB}\sum_{i\neq j}\left(\frac{a}{r_{ij}}\right)^{n^{AB}} + \varepsilon^{AA}c^{AA}m^{AB}\left(\frac{1}{\sqrt{\rho_i}}\right)\sum_{i\neq j}\left(\frac{a}{r_{ij}}\right)^{m^{AB}}\right]\left(\frac{1}{r_{ij}}\right). \end{aligned} \quad (3.42)$$

The SC force calculation is used thusly by the MBN Explorer, multipurpose computer code suitable for use in multiscale modelling [26].

## 3.3.7 Adjustments for Conservation of Energy and Momentum

Equations 3.39, 3.40 and 3.42 have been independently derived from equation 3.35 [20, 21, 25], often with small differences, which were used by the MBN Explorer, multipurpose computer code suitable for use in multiscale modelling [26], and to determine the melting behaviour of Al nanoclusters with Pb cores, with small modifications to describe the immiscibility of the alloy system [27]. It provides a formula with which to calculate the force of atom $j$ on atom $i$. This study uses equation 3.39.

However, the terms for two alloy atoms are to be calculated with $n-term_{ij} \ll m-term_{ij}$ where:

$$\begin{aligned} n-term_{ij} &= \left[\varepsilon^{AB}n^{AB}\left(\frac{a^{AB}}{r_{ij}}\right)^{n^{AB}}\right] \\ m-term_{ij} &= \left[\frac{\varepsilon^{AA}c^{AA}m^{AB}}{2}\left(\frac{1}{\sqrt{\rho_i}} + \frac{1}{\sqrt{\rho_j}}\right)\left(\frac{a^{AB}}{r_{ij}}\right)^{m^{AB}}\right]. \end{aligned}$$

The n-term is the same for either element, the m-term varies widely due to $\varepsilon^{AA}c^{AA}$ which depends on the element. Thus, when calculating the force constant, $m-term_{ij} \neq m-term_{ji}$ , and as it is much larger than the $n-term_{ij}$, $f_{ij} \neq f_{ji}$ , and the difference is significant. An

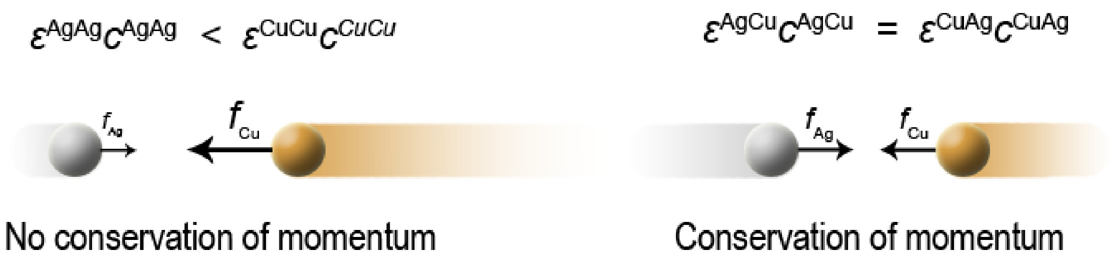

$\varepsilon^{AgAg}c^{AgAg} < \varepsilon^{CuCu}c^{CuCu}$
$f_{Ag}$
$f_{Cu}$
No conservation of momentum
$\varepsilon^{AgCu}c^{AgCu} = \varepsilon^{CuAg}c^{CuAg}$
$f_{Ag}$
$f_{Cu}$
Conservation of momentum

Figure 3.3 The forces calculated using the mixed potential and the original Sutton-Chen parameters from Table 3.2 are unequal and do not conserve momentum, and requires a new $c$ value for mixed potentials.

empirical calculation of the partial forces $f_{ij}$ and $f_{ji}$ for a two-body system of Cu and Ag yield partial forces of 0.82 eV for the Cu atom and 0.42 eV for the Ag atom at a separation of 2.3 Å, which is far from the same. It is clear the critical difference in these partial forces comes from the second term in the force calculation, due to the large differences in $c^{AA}$ and $c^{BB}$. A visual representation of the discrepancy is presented in Figure 3.3.

This is because the $c$ values, though a dimensionless parameter, represent the lattice stabilities and the $c^{AA}$ and $c^{BB}$ values represent the states of the pure metals, but not necessarily that of the alloy. If these lattice parameters differ too greatly, neither can accurately reflect the state of the lattice. Therefore, the model fails. For this purpose, a new Sutton Chen $c$ parameter must be calculated for the alloy.

Returning to the original derivation of the Sutton-Chen [14], the dimensionless parameter $c$ was fixed to a constant for a crystal at equilibrium and related to the lattice sum

$$S_n^f = \left( \sum_j \left( \frac{a^f}{r_j} \right)^n \right), \tag{3.43}$$

with the sum taken over all inter-atomic separations $r_{ij}$ from a random atom $i$. Setting $a^f$ equal to the lattice parameters allows $c$ to be written as

$$c = \frac{nS_n^f}{m\sqrt{S_n^f}}. \tag{3.44}$$

As each atom itself makes up 50% of its interactions with other atoms, half of the interspecies interaction is provided from the element of the local atom, and half from that of the nearest neighbour.

Using this new definition for $a$ in equation 3.43 and substituting into the equation for $c$ allows us to then calculate a new lattice sum parameter

$$c^{AB} = \left(c^{AA} c^{BB}\right)^{1/2}. \qquad (3.45)$$

This result was also compared to a manually calculated $c$ using the lattice sum equation and yielded similar results. Using this new parameter to calculate the partial energies between atoms of different elements gives an equal partial force being exerted by each atom, which conserves energy and momentum.

## 3.4 Genetic Algorithms

Barron et al. [28] used a Genetic Algorithm to find the low Lennard-Jones energy configurations for atomic clusters in the range $13 < n < 147$. Configurations $C$ were created with points $p$ denoting the positions of lattice atoms. Using the LJ potential, the energy of the system was calculated and the configuration with a minimum energy was found using a Genetic Algorithm.

The Genetic Algorithm involves creating a parent "population" of 9 configurations and "mating" or mutating them to create successive generations. From these generations, the configuration with the lowest LJ energy is chosen as the LJ configuration.

Once the parent generation has been created, the configurations are relaxed. Atoms with too close proximity are randomly relocated to the outer shell of the configuration to prevent overflow as a result of a too high LJ energy. Bumps are also removed from the configuration by eliminating atoms with an unusually low number of nearest neighbours and replacing them on the outer shell. Once the configuration is relaxed, bumps are removed and the LJ energy has been minimised, the parent configurations are then used to create the next generation. Six parent configurations are randomly chosen, though the probability of being selected as parent is increased with decreasing LJ energy.

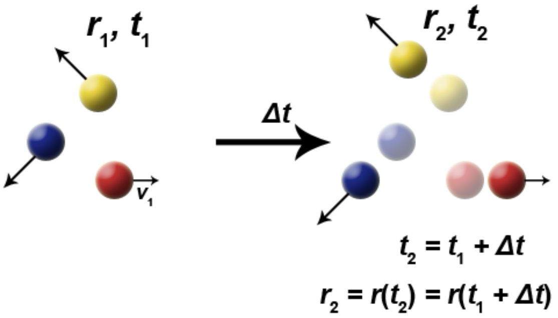


Figure 3.4. Using time integration algorithms, a new position after lapse of time $\Delta t$ can be calculated.

# 3.5 Time Integration Algorithms

Rather than using Genetic Algorithms, Time Integration Algorithms can be used. Time algorithms are necessary to integrate the equations of motion in order to follow the trajectory of the interacting particles in the system [29]. There are two commonly used algorithms, namely the Verlet algorithm and the predictor-corrector algorithm. These methods are based on finite difference methods where the time is divided into small discrete steps of $\Delta t$. Knowing the position and its time derivatives at a certain time $t$, the new position and its corresponding time derivatives can be calculated at a time $t+\Delta t$ using these algorithms, as represented in Figure 3.4. Errors that may occur during time integration are truncation errors and round off errors. Truncation errors are intrinsic to the specific algorithm and relate to the use of finite difference methods, such as Taylor expansions where the equation is truncated after a certain number of terms. The Verlet algorithm has a truncation error proportional to $\Delta t^4$ for each integration time step. Round off errors relate to the implementation of the algorithm, such as the finite number of digits used in computer arithmetics. Round off errors can be kept to a minimum by using 64-bit precision. Both round off errors and truncation errors can also be minimised by using smaller time steps $\Delta t$.

## 3.5.1 The Verlet Algorithm

### 3.5.1.1 The Taylor expansion

The Verlet algorithm is two third order Taylor expansions [30, 31] for the position $\bar{r}(t)$ at the times $t+\Delta t$ and $t-\Delta t$:

$$\bar{r}(t+\Delta t)=\bar{r}(t)+\bar{v}(t)\Delta t+\tfrac{1}{2}\bar{a}(t)\Delta t^2+\tfrac{1}{6}\bar{b}(t)\Delta t^3+O\left(\Delta t^4\right), \tag{3.46}$$

$$\bar{r}(t-\Delta t)=\bar{r}(t)-\bar{v}(t)\Delta t+\tfrac{1}{2}\bar{a}(t)\Delta t^2-\tfrac{1}{6}\bar{b}(t)\Delta t^3+O\left(\Delta t^4\right). \tag{3.47}$$

Where $\bar{v}$ is the velocity, $\bar{a}$ the acceleration and $\bar{b}$ the third derivative of $\bar{r}$ . $O\left(\Delta t^4\right)$ is the fourth term of the Taylor expansion, which is later omitted, making the local error of the Verlet scheme of the order of $O\left(\Delta t^4\right)$. The two expressions are added together to obtain

$$\bar{r}(t+\Delta t)=2\bar{r}(t)-\bar{r}(t-\Delta t)+\bar{a}(t)\Delta t^2+O\left(\Delta t^4\right). \tag{3.48}$$

Here

$$\bar{a}(t)=-{}^{1}\!/\!{}_{m}\nabla V\left(\bar{r}(t)\right). \tag{3.49}$$

The velocity which is needed to calculate the kinetic energy can be found from:

$$\bar{v}(t)=\frac{\bar{r}(t+\Delta t)-\bar{r}(t-\Delta t)}{2\Delta t}, \tag{3.50}$$

which has a global error proportional to $\Delta t^2$. Alternatively, the velocity can be obtained through the Velocity Verlet Scheme.

#### 3.5.1.2 The Velocity Verlet Scheme

The alternative to using equation 3.50 is to use the Velocity Verlet:

$$\bar{v}(t+\Delta t)=\bar{v}(t+\Delta t/2)+\tfrac{1}{2}\bar{a}(t+\Delta t)\Delta t, \tag{3.51}$$

which can be obtained from

$$\bar{r}(t+\Delta t)=\bar{r}(t)+\bar{v}(t)\Delta t+\tfrac{1}{2}a(t)\Delta t^2, \tag{3.52}$$

$$\bar{v}(t+\Delta t/2)=\bar{v}(t)+\tfrac{1}{2}\bar{a}(t)\Delta t, \tag{3.53}$$

and

$$\bar{a}\left(t+\Delta t\right) = {}^{-1}\!/_{m} \nabla V\left(\bar{r}\left(t+\Delta t\right)\right). \tag{3.54}$$

Equation 3.54 is found from calculating the acceleration and substituting equation 3.52, where after the Velocity Verlet is easily obtained by substituting in equations 3.53 and 3.54.

An alternative expression for the Velocity Verlet is

$$\bar{v}\left(t+\Delta t\right) = \bar{v}\left(t\right) + \tfrac{1}{2}\left(\bar{a}\left(t\right) + \bar{a}\left(t+\Delta t\right)\right)\Delta t, \tag{3.55}$$

which is obtained from substituting equations 3.49 and 3.54 in equation 3.55. The truncation error involved in calculating the velocity through the Velocity Verlet is of the order of $\Delta t^4$ as opposed to the error of order $\Delta t^2$ of equation 3.50.

## 3.5.2 Predictor-Corrector Algorithm

An alternative to the Verlet Velocity algorithm is the Predictor-Corrector algorithm, which involves three steps [32]:

1. **Predictor:** From the known $\bar{r}$ , $\dot{\bar{r}}$ and $\ddot{\bar{r}}$ values at time $t$ the same values are predicted for a time $t+\Delta t$ by a Taylor expansion.
2. **Force evaluation:** The force is calculated from the gradient of the potential at the new time $t+\Delta t$ and the acceleration obtained from the calculated force is compared to the predicted value for $\bar{a}\left(t+\Delta t\right)$. The difference in these values is the *error signal*.
3. **Corrector:** The error signal is used to correct the values for $\bar{r}$ , $\dot{\bar{r}}$ and $\ddot{\bar{r}}$ at time $t+\Delta t$ and the error signal is proportional to the corrections made. The proportionality constant is determined to ensure the maximum stability of the algorithm and is a 'magic number'.

# 3.6 Temperature Control

The temperature in the system is related to the kinetic energy of the atoms in the crystal. The kinetic energy is obtained from the average velocities of the atoms. When the crystal is created, the velocities of the crystal are set using random velocities generated from a normal distribution. Figure 3.5 shows a visual representation of the possible spread of velocities of the atoms around

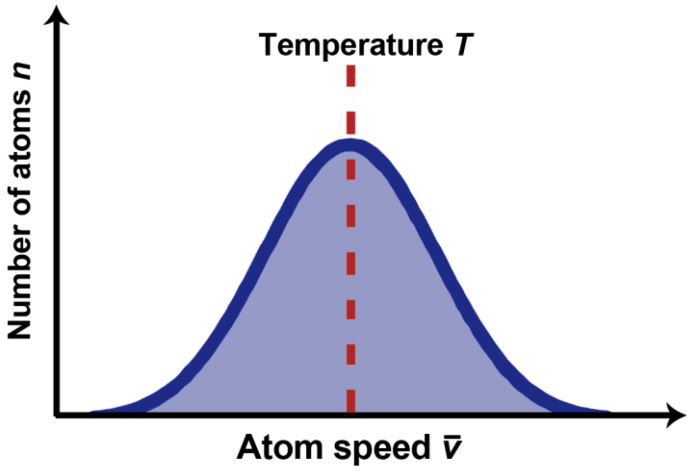


Figure 3.5 The temperature is found from the average kinetic energy derived from the distribution of atom velocities. Alternatively, a spread of atom velocities can be obtained from a normal distribution around an average velocity obtained from the desired temperature.

an average velocity calculated from temperature "*T*". The directions of these vectors are isotropically spread using unit vectors calculated from randomly generated numbers.

## 3.6.1 Kinetic Energy and Temperature

The kinetic energy is given as the sum of the kinetic energies of all the atoms in the system [33]:

$$KE = \sum_{i=1}^{N} \frac{|p_i|^2}{2m_i}. \tag{3.56}$$

The kinetic energy is related to the temperature as follows:

$$KE = \frac{k_B T}{2}\left(3N - N_C\right). \tag{3.57}$$

$N$ is the total number of atoms in the system, $N_C = 3$ is the number of constraints on the system, $T$ is the system temperature and $k_B$ is Boltzmann's constant. Thus, using the sum of atom velocities as represented in Figure 3.5, the temperature can be obtained from the average kinetic energy. This gives the temperature:

$$T = \frac{2KE}{k_B\left(3N - 3\right)}. \tag{3.58}$$

### 3.6.2 Velocity Scaling

As the velocities of the atoms in the system are updated with the time-integration algorithm, it can affect the temperature. This temperature needs to be adjusted to account for the change. The change in temperature can be calculated as [33]:

$$\Delta T = \frac{1}{2}\sum_{i=1}^{N} 2\frac{m_i\left(\lambda \upsilon_i\right)^2}{N_{df} k_B} - \frac{1}{2}\sum_{i=1}^{N} 2\frac{m_i \upsilon_i^2}{N_{df} k_B}. \tag{3.59}$$

$N_{df}$ is the number of internal degrees of freedom. Equation 3.59 can also be written as:

$$\Delta T = \left(\lambda^2 - 1\right) T\left(t\right), \tag{3.60}$$

where $\lambda$ is a factor that is multiplied to the velocities for velocity scaling, with $T\left(t\right)$ the temperature at time $t$. This gives the velocity scaling factor as:

$$\lambda = \sqrt{T_0 / T\left(t\right)}, \tag{3.61}$$

with $T_0$ the desired temperature of the system.

### 3.6.3 Berendsen Thermostat

The Berendsen Thermostat adjusts the temperature of the system with a velocity scaling factor, by coupling the system to a heat bath with a fixed temperature $T_0$ through a feedback cycle. How closely the system is coupled to the heat bath determines the scaling factor. The rate of change in temperature is proportional to [33]:

$$\frac{dT\left(t\right)}{dt} = \frac{1}{\tau}\left(T_0 - T\left(t\right)\right), \tag{3.62}$$

where $\tau$ is a coupling parameter that determines how closely the heat bath and the real system are linked. The change in temperature for each time step is then:

$$\Delta T = \frac{\delta t}{\tau}\left(T_0 - T\left(t\right)\right), \tag{3.63}$$

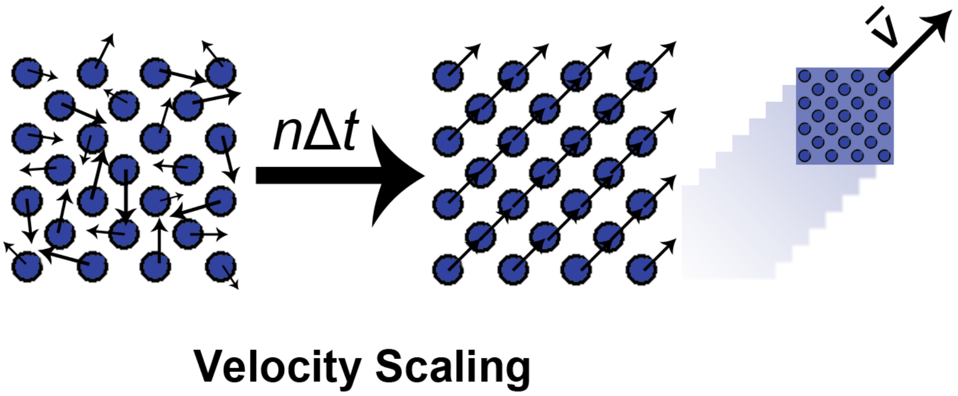


Figure 3.6. This diagram shows a representation of a velocity scaled crystal after a number of time steps, where all the atom velocities have been converted into particle translation and the particle has turned into a “flying ice-cube”.

and the velocity scaling factor is thus:

$$\lambda^2 = 1 + \frac{\delta t}{\tau}\left\{\frac{T_0}{T\left(t-\frac{\delta t}{2}\right)} - 1\right\}. \tag{3.64}$$

The coupling parameter $\tau$ is used empirically to adjust the strength of the coupling. At the limit where $\tau$ is infinitely large, the coupling is inactive. Where $\tau$ is chosen too small, temperature fluctuations have an unrealistically scaled temperature fluctuation amplitude. A typical value used in MD simulations is $\tau \approx 0.1ps$. This however does not generate a canonical ensemble.

## 3.6.4 Dissipative Particle Dynamics

A common problem of velocity scaling is that all the kinetic energy of the atoms is converted into a translated velocity of the centre of mass, whereas the internal motion of the particle relative to each other are reduced as shown in Figure 3.6. Dissipative Particle Dynamics (DPD) can be used to add random noise and prevent the amplification of relative motion of the simulated system by adding random noise to the velocity, as shown in Figure 3.7.

DPD is often used in polymers, where there are mesoscopic particles, that is, soft spheres connected by a spring [34, 35]. DPD adds pairwise dissipative random forces *f* to the force term of the equations of motion in terms of the positions $\bar{q}_i$ and momenta $\bar{p}_i$, which are:

$$\frac{d\bar{q}_i}{dt} = \frac{\bar{p}_i}{m_i}, \frac{d\bar{p}_i}{dt} = f_i(t) = \sum_{i\neq j}\left(F_{ij}^C + F_{ij}^D + F_{ij}^R\right). \tag{3.65}$$

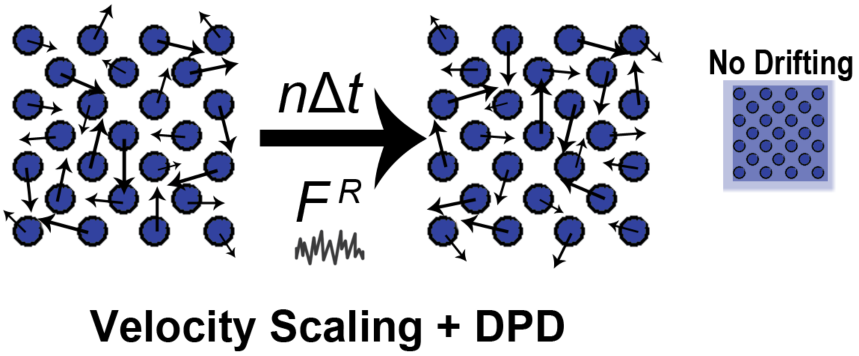


Figure 3.7. This diagram shows a velocity scaled crystal which includes random noise from DPD, preserving internal random motion, which prevents the particle from drifting.

$F^C$ are the conservative forces, $F^D$ are the dissipative forces and $F^R$ are the random forces. In the case of a Molecular Dynamics simulation, only the random forces are of concern. These random friction forces are applied to pairs of particles to conserve total linear momentum. The pairs can be chosen at random, according to a distance-weighted probability, or one nearest neighbour per atom.

The noise vector can be three dimensional and applied isotopically to the force vector. The noise vector is applied to the pairs of particles up to the cut-off distance, beyond which noise forces are 0. The vector is also linearly scaled according to inter-atom distances. The friction can be applied parallel to the interatomic separation, perpendicular to it, or in the direction of the velocity itself.

First the velocity noise factor $g$ is determined;

$$g = \sqrt{f\left(2-f\right)k_B T_{ref} / \mu}. \tag{3.66}$$

Here $\mu$ is the reduced mass of the particle pair:

$$\mu = \frac{m_i m_j}{m_i + m_j}. \tag{3.67}$$

The relative velocity vector is given as:

$$v = \lambda_B \left(v_i - v_j\right). \tag{3.68}$$

$\lambda_B$ is the Berendsen scaling factor.

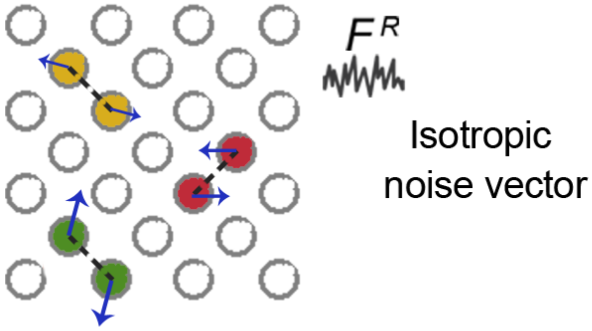


Figure 3.8. The first method of selecting vector directions is to select random vectors.

The choice is then whether to apply the noise isotropically, in which case the constructed vector would be:

$$\Delta v = -fv + g\xi, \tag{3.69}$$

where $\xi$ consists of 3 randomly chosen numbers from a normal distribution:

$$\xi = \left(\xi_1, \xi_2, \xi_3\right). \tag{3.70}$$

Figure 3.8 shows a representation of randomly chosen atom pairs with isotropically applied noise where the vectors are chosen randomly.

If the noise is to be applied in parallel, then the unit vector $e_1$ is used:

$$e_1 = \frac{r_{ij}}{\left|r_{ij}\right|}, \tag{3.71}$$

where $r_{ij} = r_j - r_i$ in the interparticle direction. Then:

$$\Delta v = \left(-fv_{par} + g\xi\right)e_1. \tag{3.72}$$

direction.

To find the component of $v$ in the interparticle direction one uses:

$$v_{par} = ve_1. \tag{3.73}$$

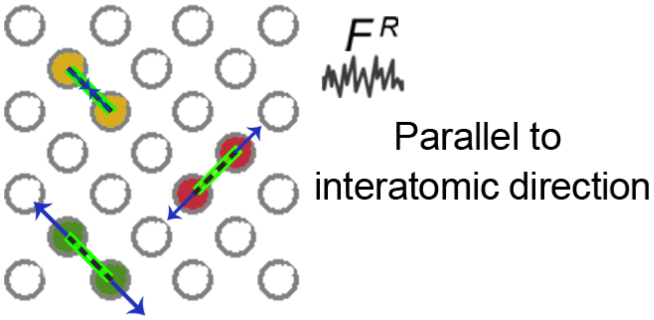


Figure 3.9. An alternative way to apply noise is in the direction parallel to the interatomic

A representation of the crystal with random noise applied in the interatomic direction can be seen in Figure 3.9.

The other option is in the direction perpendicular to the interparticle direction. In this case the perpendicular component of velocity is:

$$v_{perp} = v - (ve_1)e_1, \tag{3.74}$$

which has a unit vector of:

$$e_2 = \frac{v_{perp}}{|v_{perp}|}, \tag{3.75}$$

which can be used to calculate the unit vector $e_3$:

$$e_3 = e_1 \times e_2. \tag{3.76}$$

Then the perpendicular vector can be constructed as:

$$\Delta v = -fv_{perp} + g\left(\xi_2 e_2 + \xi_3 e_3\right). \tag{3.77}$$

Figure 3.10 shows random noise applied in the direction perpendicular to the interatomic direction.

The vectors are applied over two particles as follows:

$$\begin{aligned} v_i^R &= \lambda_B v_i + \frac{\mu}{m_i}\Delta v, \\ v_j^R &= \lambda_B v_j + \frac{\mu}{m_i}\Delta v. \end{aligned} \tag{3.78}$$

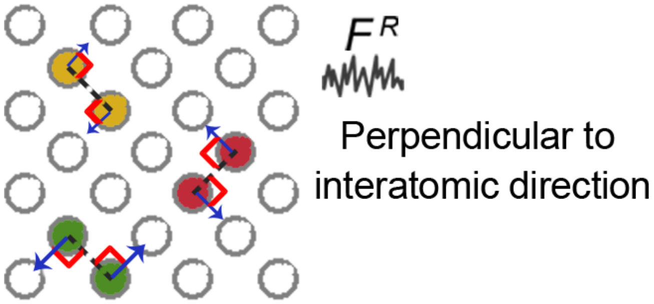


Figure 3.10. The random forces can be applied in the direction perpendicular to the interatomic direction.

Thus momentum $m_i v_i + m_j v_j$ is conserved.

## 3.7 Conclusion

After a thorough analysis of the various algorithms used to simulate materials, algorithms were chosen to be used in this study. The energies and forces were calculated with the Sutton-Chen potential, positions, velocities and accelerations were given by the Velocity Verlet integrator, and temperature control used the Berendsen thermostat. Other algorithms such as Dissipative Particle Dynamics were later used as well.

Having done a thorough study of all the algorithms involved in Molecular Dynamics simulations, the next step is implementing the code.

## 3.8 References

[1] "The Lennard--Jones potential," accessed February 6, 2011, http://polymer.bu.edu/Wasser/robert/work/node8.html.

[2] F. Ercolessi (October 9, 1997). "The Lennard-Jones potential," accessed February 6, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node15.html.

[3] "Absolute Astronomy: Lennard-Jones potential," accessed February 6, 2011, http://www.absoluteastronomy.com/topics/Lennard-Jones_potential.

[4] M. R. Wilson, *Journal of Chemical Physics* **107** (1997): 8654.

[5] University of Cambridge (2007). "Lennard Jones Potential – Solution," *NRICH*, accessed February 7, 2011, http://nrich.maths.org/6509&part=solution.

[6] J. H. Lienhard (1988). "No. 2593 John Lennard Jones," *Engines of Our Inginuity*, accessed February 7, 2011, http://uh.edu/engines/epi2593.htm.

[7] F. Ercolessi (October 9, 1997). “Potential truncation and long-range corrections,” accessed February 6, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node16.html.

[8] F. Ercolessi (October 9, 1997). *The minimum image criterion*, accessed February 6, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node18.html.

[9] University of Cambridge (2007). “Lennard Jones Potential – Problem,” *NRICH*, accessed February 6, 2011, http://nrich.maths.org/6509.

[10] W. L. Mattice (July 2, 1999). “Lennard-Jones potential,” accessed February 6, 2011, http://gozips.uakron.edu/~mattice/ps674/lj.html.

[11] D. Wolff (Sep 2, 1998). “Lennard-Jones Potential,” accessed February 6, 2011, http://www.physics.orst.edu/~rubin/CPUG/CPlab/MoleDynam/lj.html.

[12] P. Baranello, “Using the Gibbs Ensemble Monte Carlo Method for Diatomic Lennard-Jones Systems,” accessed February 7, 2011, http://kea.princeton.edu/ppe/old/pbaranel/paper.htm.

[13] J. S. Brown, *Proceedings of the Physical Society* **89** (1966): 987.

[14] A. P. Sutton and J. Chen, *Philosophical Magazine Letters* **61** (1990): 139.

[15] J. P. K. Doye and D. J. Wales, “Global Minima for Transition Metal Clusters Described by Sutton-Chen Potentials,” arXiv:cond-mat/9711038v1 (Nov 5, 1997).

[16] “Implementation of Sutton-Chen Potential for Molecular Dynamics,” accessed February 6, 2011, http://www.genetical.com/dc/ScientificResearch/Rowland/MolecDynamED/SuttonChen/Potential.html.

[17] R. Fournier, *Journal of Chemical Physics* **115** (2001): 2165.

[18] W. Smith (December 5, 2003). “Metal Potentials,” accessed February 6, 2011, http://wanglab.bu.edu/DLPOLY2/node56.html.

[19] S. Y. Liem and K.-Y. Chan, *Surface Science* **328** (1995): 119.

[20] H. Feraoun, C. Esling, L. Dembinski, T. Grosdidier, C. Coddet and H. Aourag, *Superlattices Microstruct.* **31** (2002): 287.

[21] H. Rafii-Tabar and A. P. Sutton, *Philosophical Magazine Letters* **63** (1991): 217.

[22] Z. H. Jin, H. W. Sheng, and K. Lu, *Physical Review B: Condensed Matter* **60** (1999): 141.

[23] J. L. Rodríguez-López, J. M. Montejano-Carrizales, M. José-Yacamán, *Applied Surface Science* **219** (2003): 56.

[24] D. Förster, “Modelling of the structure and dynamics of nanoparticles on metal substrates,” (MSc thesis, University of Lyon, 2012), http://www.ens-lyon.fr/DSM/SDMsite/M2/stages_M2/Forster2012.pdf.

[25] S. Özdemir Kart, “Physical Properties of Pd, Ni Metals and Their Binary Alloys,” (PhD thesis, The Middle East Technical University, 2004), https://etd.lib.metu.edu.tr/upload/12604914/index.pdf.

[26] I. A. Solov'yov, A. V. Yakubovich, P. V. Nikolaev, I. Volkovets, and A. V. Solov'yov, *Journal of Computational Chemistry* **33** (2012): 2412.

[27] Z. H. Jin, H. W. Sheng, and K. Lu, *Physical Review B: Condensed Matter* **60** (1999): 141.

[28] C. Barron, S. Gomez., D. Romero, A. Saavedra, *Applied Mathematics Letters* **12** (1999): 85.

[29] F. Ercolessi (October 9, 1997). “Time integration algorithm,” accessed February 7, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node20.html.

[30] F. Ercolessi (October 9, 1997). “The Verlet algorithm,” accessed February 7, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node21.html#verletalg.

[31] H. Jónsson, “Lecture Notes: Classical Dynamics,” (University of Iceland, 2003). Accessed February 8, 2011. http://www.dsf.unica.it/~fiore/vv.pdf.

[32] F. Ercolessi (October 9, 1997). “Predictor-corrector algorithm,” accessed February 7, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node22.html.

[33] P. H. Hünenberger, *Advances in Polymer Sciece* **173** (2005): 105.

[34] Y. Zhao, “Brief introduction to the thermostats,” accessed February 16, 2014, http://www.math.ucsd.edu/%7Ey1zhao/ResearchNotes/ResearchNote007Thermostat.pdf.

[35] S. A. Moga, N Goga and A. Hadar, *Materiale Plastice* **50** (2003):196.

# Chapter 4 - The Code and Implementation

This chapter examines the process of implementing the calculations discussed in the previous section; the particulars of coding; and some of the challenges and how they were overcome. From packing the crystals, running a simple relaxation run of a packed crystal, to optimizing processing time and the more creative solutions, are fully discussed in this chapter.

## 4.1 Packing the Crystals

The first challenge was to determine a regular way to pack the atoms to produce free surfaces with the low index surface orientations used in this study.

### 4.1.1 Packing Bulk Crystals

Each Cartesian coordinate was incremented in a set pattern, allowing for variations every other row or layer to produce the characteristic surfaces shown in Figure 4.1.

The (110) surface had to switch between two identical patterns of packing for each layer, the one slightly offset in height and position from the other. To program the offset, the "mod" function was used, using a remainder to keep track of whether to offset a layer or row, or not. Combining the function with the counter for the particular row or layer allowed the precise switching between one set of coordinates and another at the correct junction.

```
shift_layer = j Mod 2

set_position(int_selected_atom, x) = i * root_two_a + shift_layer * (root_two_a / 2)
set_position(int_selected_atom, y) = j * lattice_const / 2
set_position(int_selected_atom, z) = k * (root_two_a) + shift_layer * (root_two_a / 2)
```

Here "`i`" is the atom number in the row, "`j`" is the number of rows and "`k`" the number of layers.

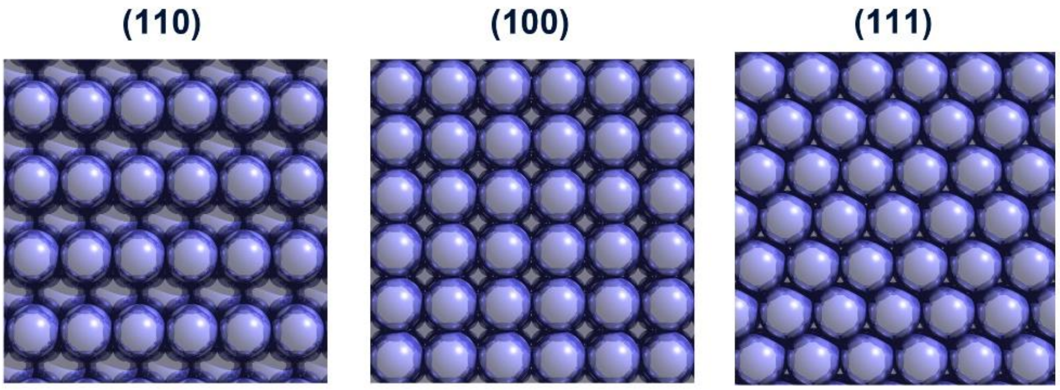
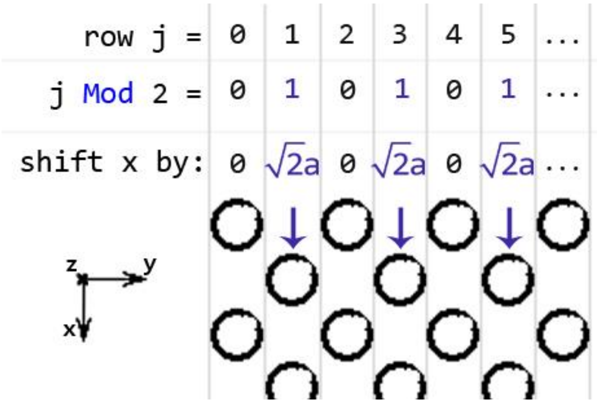


Figure 4.1. The free surfaces of the three bulk crystal packings; the (110) free surface with crests and troughs; the (100) square-packed surface; and the closely packed (111) surface.



| row j = | 0 | 1 | 2 | 3 | 4 | 5 | ... |
|---|---|---|---|---|---|---|---|
| j Mod 2 = | 0 | 1 | 0 | 1 | 0 | 1 | ... |
| shift x by: | 0 | $\sqrt{2}a$ | 0 | $\sqrt{2}a$ | 0 | $\sqrt{2}a$ | ... |

Figure 4.2. This graphic demonstrates the principle of using “Mod” to shift atoms at regular intervals in the lattice.

Using the “j Mod 2” function allows the shift of every second row, in the x- and z-direction, as can be seen in Figure 4.2, which is found in code lines 3 and 5. To pack the (100) is simple as each row is identical, though every second layer is offset in the x- and y-directions:

```
shift_layer = k Mod 2

set_position(int_selected_atom, x) = i * root_two_a + shift_layer * (root_two_a / 2)
set_position(int_selected_atom, y) = j * root_two_a + shift_layer * (root_two_a / 2)
set_position(int_selected_atom, z) = (k) * (lattice_const / 2)
```

Finally, the most complex to pack is the crystal with a (111) free surface, where both rows and layers are shifted:

```
shift_layer = k Mod 3

shift_row = j Mod 2

set_position(int_selected_atom, x) = i * root_two_a + shift_row * (root_two_a / 2)
    + shift_layer * (root_two_a / 2)
set_position(int_selected_atom, y) = j * three_root_two_a
    + shift_layer * root_three_root_two_a
set_position(int_selected_atom, z) = k * root_three_a
```

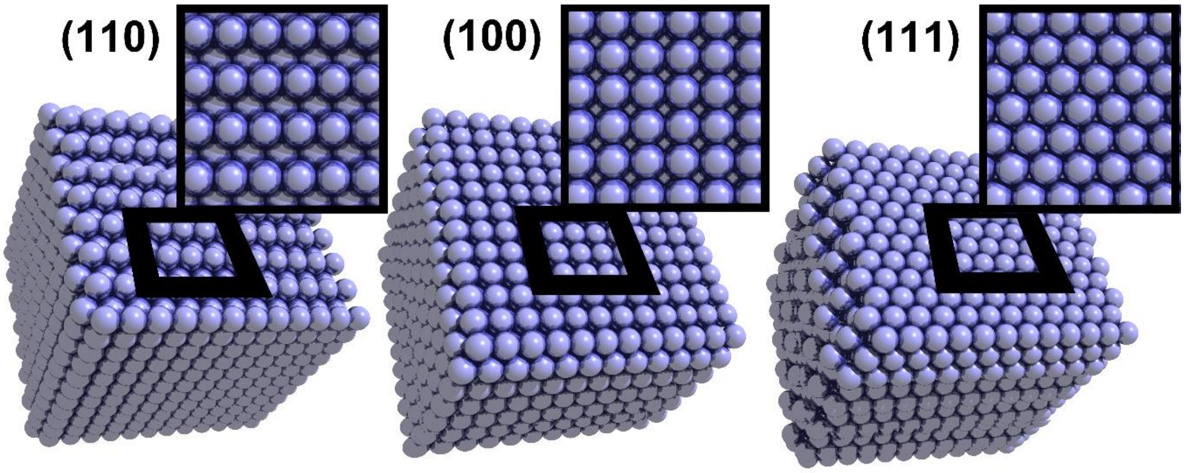


Figure 4.3. A representation of the surface orientations of the bulk crystals used to calculate the vacancy formation energy, migration energy, segregation energy and diffusion activation energy in bulk (volume of the crystal away from surface defects).

The final product is a large crystal, larger in width than the cut-off radius, shown in Figure 4.3. The size means that in the centre of the crystal, surface effects no longer influence atom dynamics. The exact shape of the crystal also needs to fit like a puzzle piece with a copy of itself for periodic boundaries which is discussed in section 4.3.3. As a Molecular Dynamics simulation works on a larger scale than some of the more intensive models like Density Functional Theory (DFT), it is able to simulate thousands of atoms up to 5 nm in width with little difficulty.

## 4.1.2 Packing Nanocubes

Having unpacked the structure of the bulk crystals, creating custom nanocrystals is the next challenge. The nanocrystals simulated in this study resemble nanocubes manufactured by Wang et al. who have reported synthesis of Cu nanoparticles with six (100) facets in perfect cubes [1], and similar Ag nanocubes manufactured by Sun et al. [2].

Packing the cubes themselves involves adjusting the (100) crystal packing to take on perfect (100) faces on all sides. The cubes produced in the labs sometimes assumed the shape of a rhombicuboctahedron, meaning the cubes were truncated where the edges and corners have had layers shaved away. To mimic this, {110} and {111} planes were calculated and used as a boundary condition of perfect cubes; if atoms fell on the wrong side of this plane, they were removed, as shown in Figure 4.4.

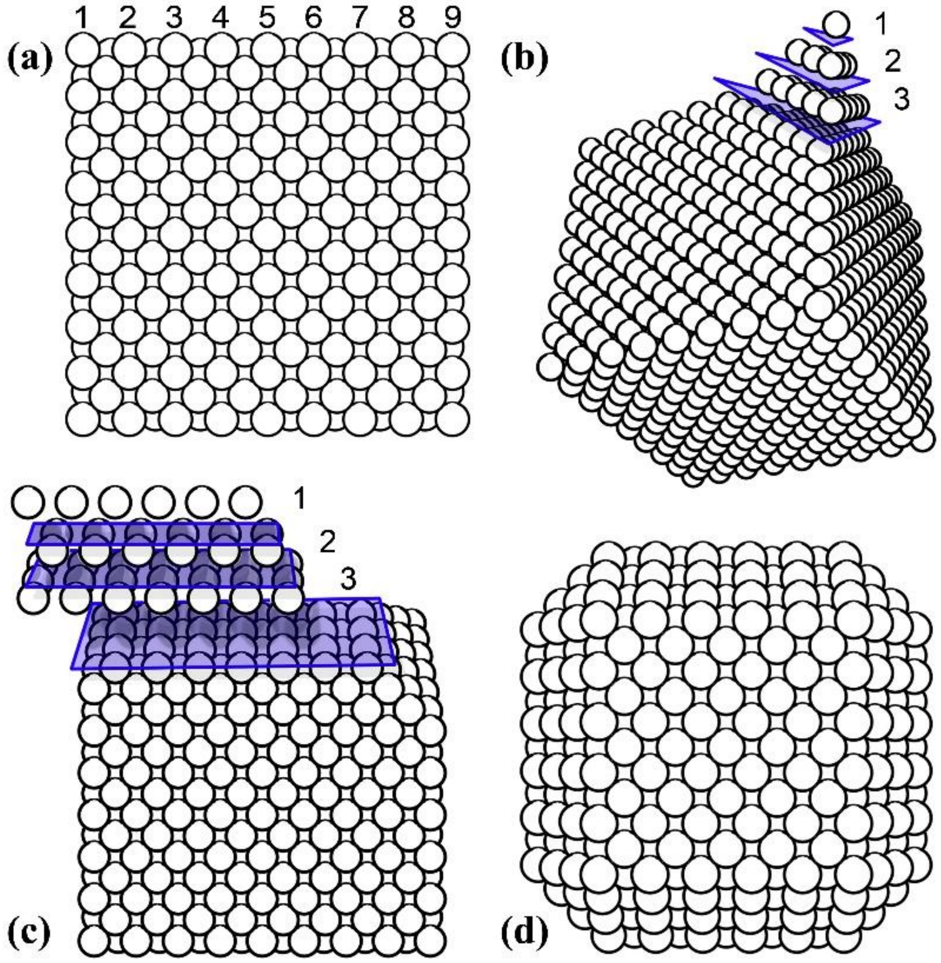


Figure 4.4 (a-d) shows the process of packing an imperfect Cu nanocube packed 9 atoms per row, with (100) faces, (110) edges and (111) corners. First the perfect cube with (100)-faces was packed in (a), then 3 layers of atoms were removed along the (111) plane to reveal (111) corners (b), and finally another 3 layers of atoms were cut away along the (110) edges (c). This produced the $9\times9\times9-3$, or $9_{-3}$ particle (d).

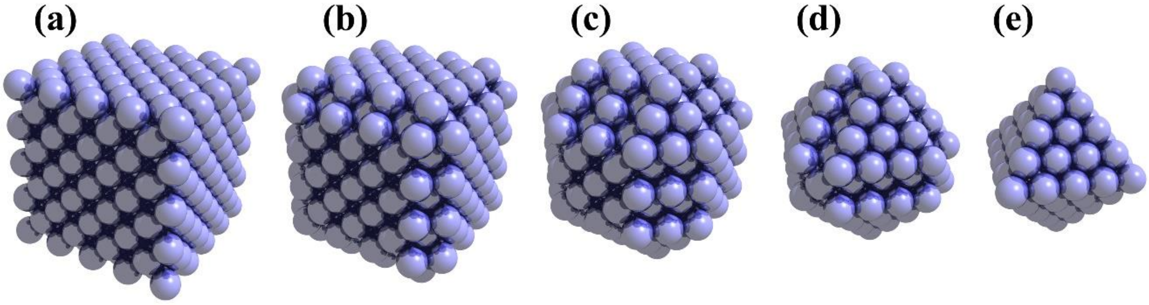


Figure 4.5 shows a perfect Cu nanocube, $5_0$ shown in (a), with edges and corners progressively cut away to produce rhombicuboctahedrons (b) $5_{-1}$, (c) $5_{-2}$, (d) $5_{-3}$, and the octahedron $5_{-4}$ in (e).

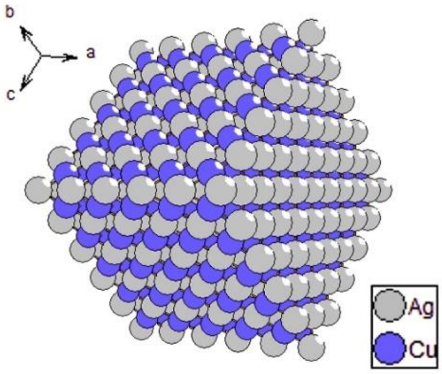


Figure 4.6. A small 6×6×6 cube packed with each atom having a maximum number of nearest neighbours of a foreign element produces a layered packing.

Figure 4.5 shows smaller examples of the various shapes of nanocubes that can be easily produced using the software, from a perfect cube, through truncated cubes, called a rhombicuboctahedron, to the final shape of two inverted pyramids attached at their bases: an octahedron.

The particular shapes of these nanocubes provided an opportunity to study the interactions of relationships between low-index surfaces, their boundary interactions, and particle size, along with the other factors that could be studied in the bulk crystals.

## 4.1.3 Assigning the Elements

As this study ultimately aims at examining segregation in bimetallic nanoparticles, packing crystals with more than one element was essential.

### 4.1.3.1 Homogenous concentration

A homogenous packing of crystals would be an ideal starting point from which to measure segregation, as any change in the consistent concentration of the different metals would be easier to pick up. However, an attempt to make sure that each atom has the maximum number of nearest neighbours of a foreign element in contact with them led to a banded, or layered, structure, as seen in Figure 4.6. Ultimately it was decided to try other configurations to measure segregation in this study.

#### 4.1.3.2 Random distribution

Another method of mixing two different metals was to use random assignment of each element. All atoms would be set to the first element, and, depending on the concentration, a certain number of atoms would then be changed to the second element, chosen by a random number generator. This method was used for a long relaxation run, where atoms are moved to positions which minimize the crystal potential energy, at the very end of this study in Chapter 9.

#### 4.1.3.3 Assigning the Central Atom as a Foreign Element

The final method of assigning a second element to the crystal was also the main method used for much of this study. By calculating the average position of all atoms in the x-, y-, and z-directions, the central atom of the crystal can be easily determined. By setting the element of this atom to a foreign element, the atom can be used for calculations that are independent of surface interactions.

#### 4.1.3.4 Assigning a Custom Atom as a Foreign Element

Finally, in some calculations such as determining the migration energy along a diffusion path, the entire crystal is set to the first element, and the selected atom is set to the second element. In the case of the migration segregation energy calculations, the elements of the atoms are switched around, so that the main selected atom is always the one with the foreign element and all the others are of the same original element.

## 4.2 Relaxing the Crystal

A large part of the simulations run in this study involved looking at the reaction of atoms to their environment in real time. The integration step subroutine is called multiple times in the code and used in different capacities, from relaxing atoms in a local location, to relaxing the crystal after a change in atom positions and elements, to main relaxation runs that examine the effects of crystal size, shape, temperature and free surfaces on the energy and atom interactions over time. For each time-step, the integration step is run, where the dynamics of each atom is calculated for the same instant.

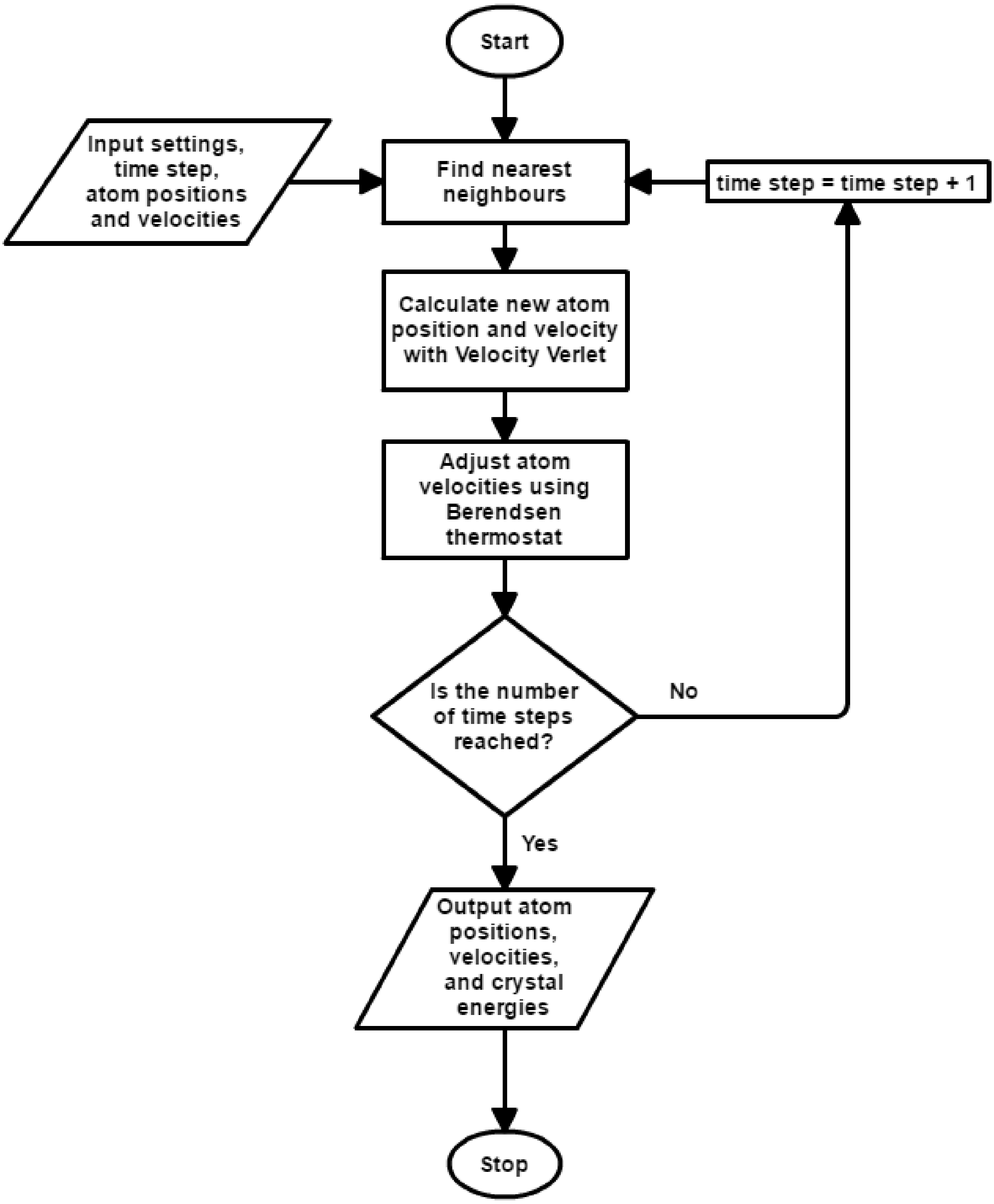

Start
Input settings, time step, atom positions and velocities
Find nearest neighbours
time step = time step + 1
Calculate new atom position and velocity with Velocity Verlet
Adjust atom velocities using Berendsen thermostat
Is the number of time steps reached?
No
Yes
Output atom positions, velocities, and crystal energies
Stop

Figure 4.7. A flow diagram representing the main crystal relaxation process.

### 4.2.1 The Integration Step

Each small time-step is only a picosecond, and thus the longest run of 500 000 steps is a mere 0.5 milliseconds of time. The process is summarized in Figure 4.7. The settings include a large number of variables, including the Sutton-Chen parameters, number of atoms, time step, required temperatures, the thermostat constant which determines heat damping, and so

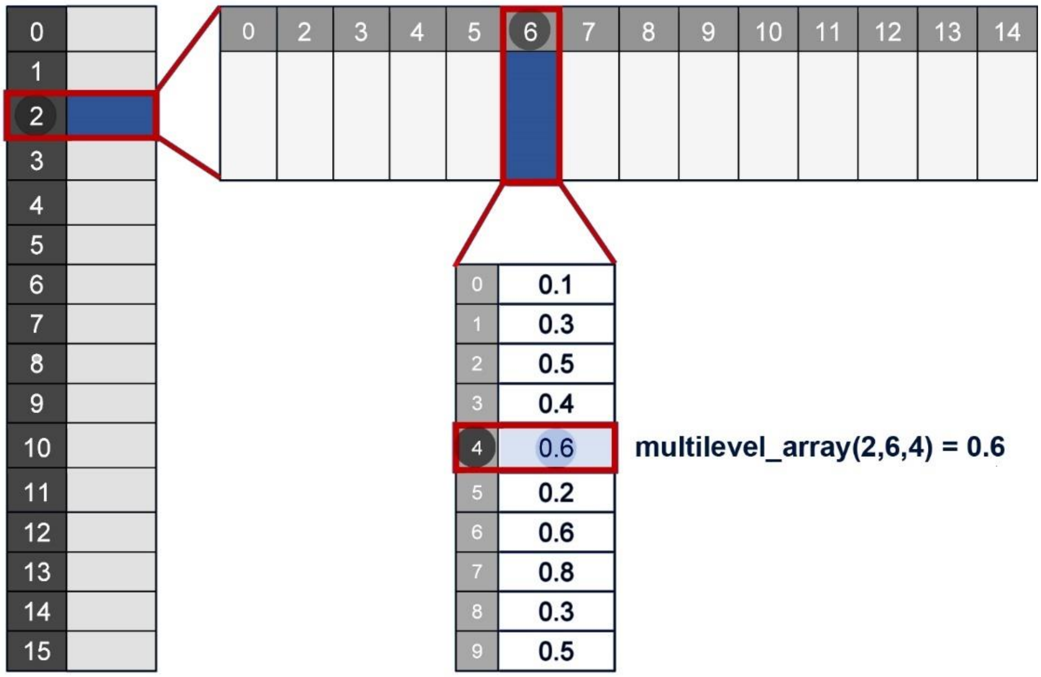


Figure 4.8. A representation of a multilevel array. Arrays can have many levels, but can compound the amount of memory needed to store them quickly with each added dimension.

on. Along with these settings, each atom’s position and initial velocity vector is also read into the program.

The main program uses a simple loop which moves the atoms according to the Velocity Verlet, adjusts the temperature, and then checks whether the run has been completed. The diagram does gloss over some of the processes, such as checks for saving regular image files, a sleep function which allows the program to pause, and a function which writes the progress to console, but it shows the important steps.

### 4.2.2 Using Multilevel Arrays

The program makes extensive use of multilevel arrays. The largest of these is the array which stores the nearest neighbour information, which is discussed in the next section.

Figure 4.8 shows a representation of a multilevel array, with the dimensions 16×15×10 (recall that the first entry is always 0). Though it appears small at first glance, the example array can store 2 400 entries. Although it is an easy way to organize related data, this is also a shortcoming of this type of array, as it ties up large quantities of memory, not all of which is always used.

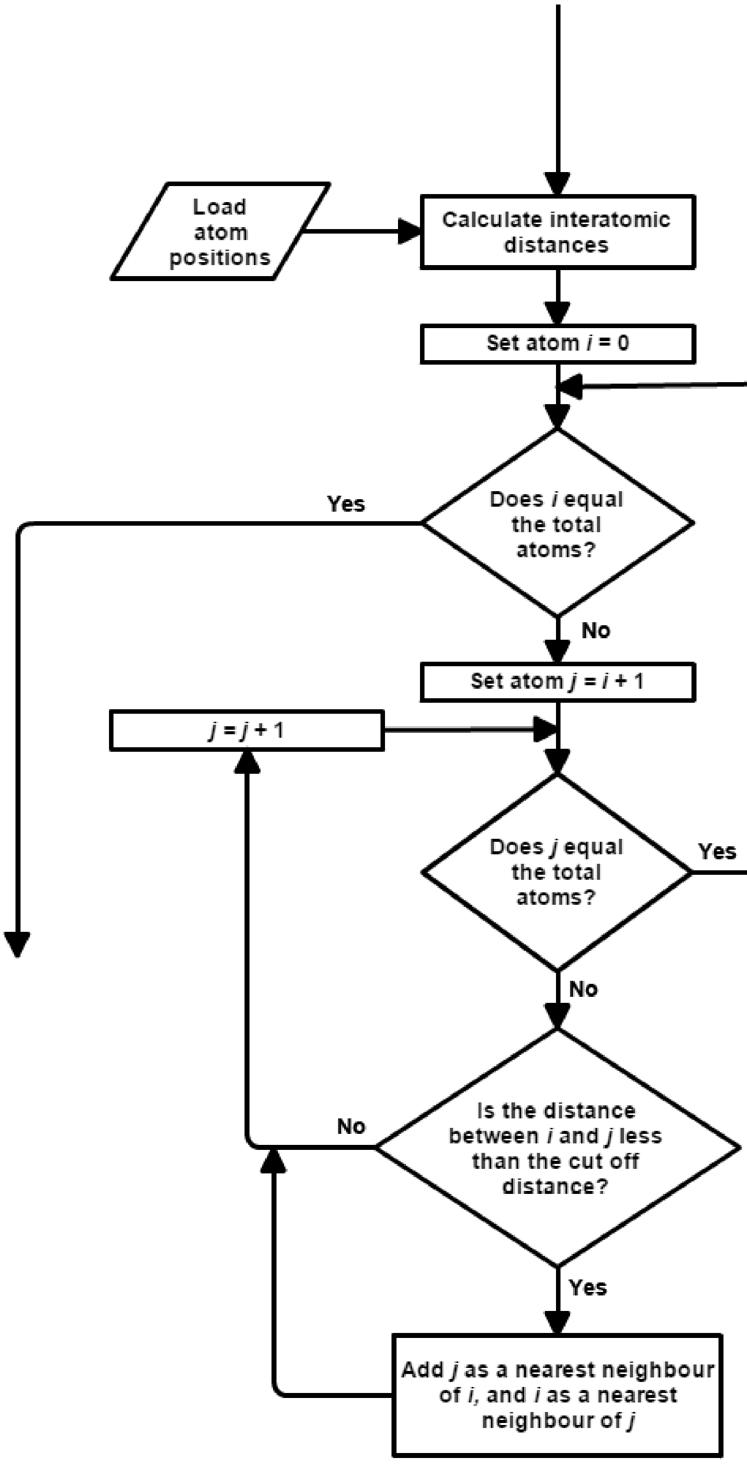

Load atom positions
Calculate interatomic distances
Set atom *i* = 0
*i* = *i* +1
Yes
Does *i* equal the total atoms?
No
Set atom *j* = *i* + 1
*j* = *j* + 1
Does *j* equal the total atoms?
Yes
No
No
Is the distance between *i* and *j* less than the cut off distance?
Yes
Add *j* as a nearest neighbour of *i*, and *i* as a nearest neighbour of *j*

Figure 4.9. This flowchart shows the process of isolating which nearest neighbours are within the cut off range of an atom. The subroutine is called from within the program, and the program continues using the list constructed during the sub.

### 4.2.3 Selecting Nearest Neighbours

The array which stores the nearest neighbour information is a multilevel array. It needs multiple levels, as each atom has its own list of neighbours, and each atom-neighbour interaction has a few factors that need to be saved. These include reference numbers to point at their positions in the array so that the correct index is used to call up a particular neighbour. Additionally, this array was used to save whether the interactions were between atoms of the same element, element 1 or 2, or different elements.

The flowchart presented in Figure 4.9 shows the process by which this array is populated. It requires nested loops that cycle through each atom, then every other atom it interacts with, and checks whether they are close enough to each other to interact. (The second loop starts at $j = i + 1$, which is a time-saving method discussed in section 4.3.4.)

## 4.2.4 Implementing the Bimetallic Potential

From the very beginning the program was written using equation 3.39. Where parameters like $c^{AB} \to c^{AA}$ , equation 3.39 becomes equation 3.25. Thus, the following code is equation 3.39, using variables that are pre-determined by the atom whose partial force is being calculated, and the nearest neighbour whose contribution to the force is being considered. The element of the selected atom, called “D” in the code, and the nearest neighbour, “nearest_neighbour”, will determine which Sutton-Chen parameters are used, namely, parameters for pure-metal interactions, or mixed parameters.

```
partial_force_constant = (g_dbl_SC_constants(e, sc_element) / _
         (g_dbl_interatomic_distance(D, nn) * g_dbl_interatomic_distance(D, nn))) _
         * ((g_dbl_SC_constants(n, sc_element) * g_dbl_nn_SC_terms(D, nn, 1)) _
         + ((((g_dbl_SC_constants(m, sc_element) * g_dbl_SC_constants(c, sc_element))_
         / 2)) * (g_dbl_nn_SC_terms(D, nn, 0))) _
         * ((1 / g_dbl_nn_SC_terms(D, 0, 0)) + (1 _
         / g_dbl_nn_SC_terms(nearest_neighbour, 0, 0))))
```

The parameters for the pure metals and mixed interactions are calculated when the settings are created, and passed with other settings when the program is loaded. These are stored in a simple 2-level array. During the integration step, the type of interaction is stored in the nearest-neighbour information array, and called up to determine which parameters to load. This check is repeated for every interaction.

## 4.2.5 Loading and Saving

The loading and saving modules were written in C# by L. A. L. Wessels [3], and integrated into the Visual Basic code using:

```
Imports FileUtils
```

With simple commands, individual variables could be read from the settings files, saved to them, and updated. Atom positions and velocities were saved in a resume file with the *.txt.gz extension, a zipped file which saves a percentage of hard drive space when writing the files. The file utilities could also save image files – snapshots of the atoms' positions and instantaneous system potential energy, kinetic energy, and temperature – to compile into a ISO (Image Standard Optical) file, from which temperature and energy graphs, and even videos can be extracted using another (program-) plugin from the same author.

## 4.3 Saving Calculation Time

Although Molecular Dynamics calculations are less time-intensive than more intensive methods like DFT, and are measured in days or weeks, rather than months, it is still beneficial to streamline calculations, cut down on calculation time and save as much memory space as possible.

### 4.3.1 Streamlining calculations

Although it is often easier to follow the equations in the code by doing them step by step, this creates extra operations, and when these are done thousands of times inside nested loops (themselves an undesirable programming trait) they add up. Below is an example of code which could be streamlined:

```
For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)

            a = g_dbl_SC_constants(a, sc_element)

            distance = g_dbl_interatomic_distance(D, nn)

            n = g_dbl_SC_constants(n, sc_element)

            m = g_dbl_SC_constants(m, sc_element)

            mterm = (a / distance) ^ m

            nterm = (a / distance) ^ n

Next nn
```

In this case the code is easier to read and debug should errors arise. However, especially in a loop such as the "For"-loop in the above example, the extra steps needed to define each variable can take up precious micro-seconds, which, when compounded over thousands of iterations, can slow down a calculation by a significant percentage.

```
For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)

            mterm = (g_dbl_SC_constants(a, sc_element) / _
                    g_dbl_interatomic_distance(D, nn)) ^ _
                    g_dbl_SC_constants(m, sc_element)
            nterm = (g_dbl_SC_constants(a, sc_element) / _
                    g_dbl_interatomic_distance(D, nn)) ^ _
                    g_dbl_SC_constants(n, sc_element)
Next nn
```

Above is a sample of the same code where the values were injected straight into the equations. Though the equation becomes long and difficult to read, it saves computation time and is much preferred. Wherever possible, it is more efficient to eliminate intermediate steps that serve to clarify, in favour of simplicity in the code.

### 4.3.2 The Cut-Off Radius

With reference to Figure 3.2 from the previous chapter, the interaction potential that determines the magnitude of the force of one atom acting upon another falls off to 0 the further apart the atoms are from each other. At a certain point the calculation cost of taking into account these miniscule influences becomes too expensive when compared to the effect they have on the overall forces in the system. As such, a cut-off radius is determined beyond which all interactions are assumed to be 0, as mentioned in section 3.2.2. Some typical cut-off distances are $2.5\sigma$ and $3.2\sigma$, distances that include successively more layers of nearest neighbours, where $\sigma$ is the distance between two directly adjacent atoms when the potential is zero. These larger cut-off distances are consequently costlier in processing time and memory. The current study used $3.2\sigma$ wherever possible.

### 4.3.3 Periodic Boundaries

A large part of this study involved comparing calculations from the simulations with experimental values. This means using large crystals that simulate bulk conditions, which can then be compared to reference values from experimental measurements. Of course, such large crystals can contain tens of thousands of atoms and consume weeks of processing time and gigabytes of internal memory. Another method for saving some time and memory is to simulate a smaller central crystal, and replicate it in its surrounds, mapping its reactions back onto itself. These copies are made in the x- and y-directions, leaving the z-surface free. In effect, the left-most atoms will see the right-most atoms as their left-hand neighbours, and so on (refer to Figure 4.10).

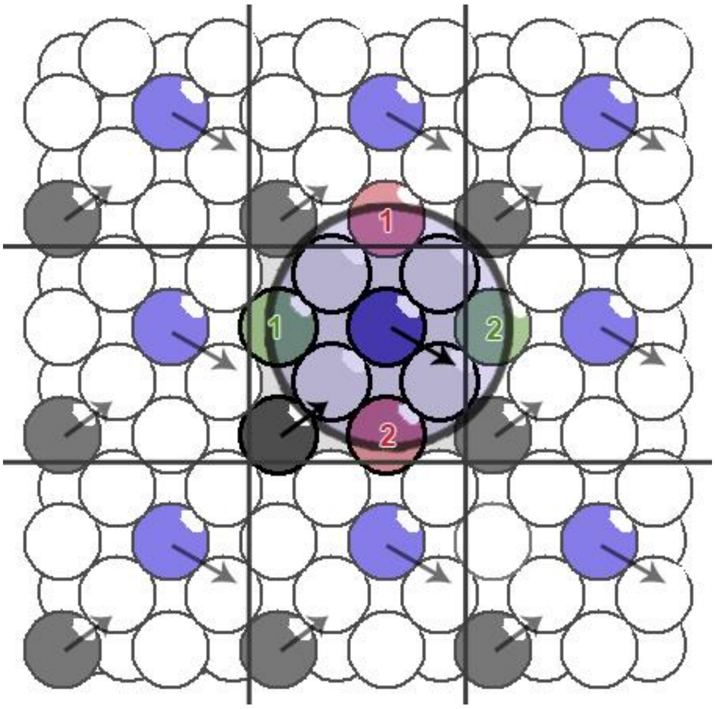
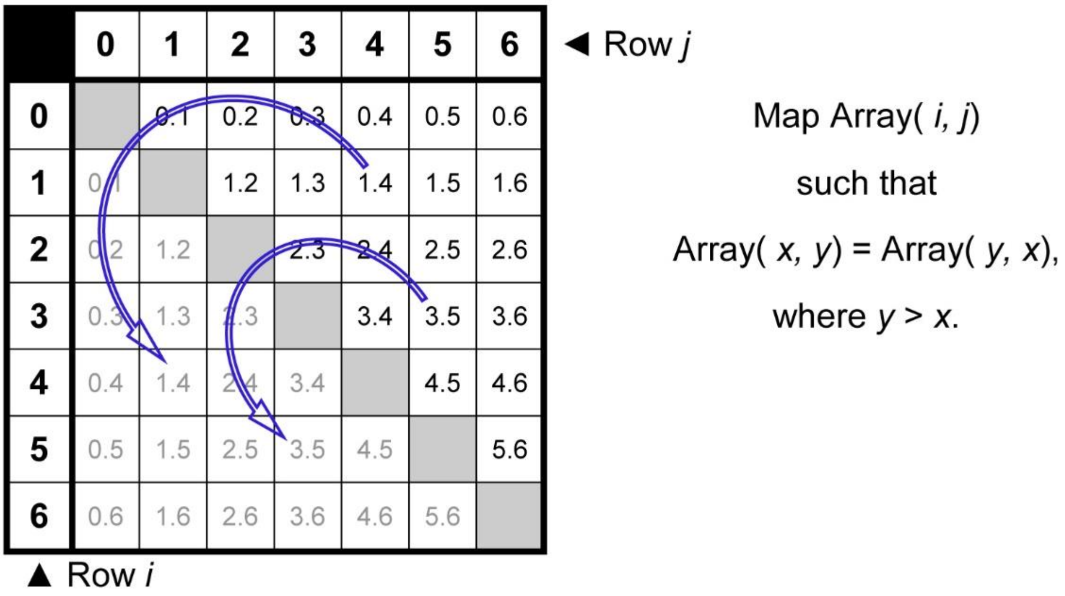


Figure 4.10 shows how periodic boundaries would work, mirroring the position of every atom 8 times in its surroundings. Atoms close to the boundaries may interact with atoms on the other side of the original crystal because of this mirroring. It is important to choose the box-length large enough so that, unlike in this example picture, an atom will not have the same atom as nearest neighbour twice, or even interact with itself.

As a result, every atom will have endless neighbours in every direction except the z-direction, allowing the computation of large crystal dynamics without the processing cost. Of course, the box lengths of the original crystal need to be larger than the cut-off distance. The above example fails in that regard in that some of the atoms are counted twice as nearest neighbours, which can lead to artificial effects that are artefacts of the boundaries, such as wave-like undulations on the surface, and not true reactions of a bulk crystal.

### 4.3.4 Mirroring Array Values

Mirroring array values is another simple way to save calculation time. As mentioned before in section 4.2.3, it is an invaluable tool to save time when calculating interactions of thousands of atoms. Figure 4.11 shows an example where the interactions between 7 atoms, 0 to 6, need to be calculated; each atom with every other atom except with itself. Using a simple rule to divide the array into halves, the first atom *x*, and the second atom *y* are chosen such that *y* > *x*. As the forces and influences upon each other are equal, the interaction for (*x*, *y*) can be calculated once, and the result copied to (*y*, *x*).



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 0 | | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 |
| 1 | 0.1 | | 1.2 | 1.3 | 1.4 | 1.5 | 1.6 |
| 2 | 0.2 | 1.2 | | 2.3 | 2.4 | 2.5 | 2.6 |
| 3 | 0.3 | 1.3 | 2.3 | | 3.4 | 3.5 | 3.6 |
| 4 | 0.4 | 1.4 | 2.4 | 3.4 | | 4.5 | 4.6 |
| 5 | 0.5 | 1.5 | 2.5 | 3.5 | 4.5 | | 5.6 |
| 6 | 0.6 | 1.6 | 2.6 | 3.6 | 4.6 | 5.6 | |

Map Array( *i, j*)

such that

Array( *x, y*) = Array( *y, x*),

where *y* > *x*.

Figure 4.11. Mirroring identical values saves time. As the force of atom *x* on atom *y* is the same as the force of atom *y* on atom *x*, it only needs to be calculated once. The force of atom *x* on itself does not need to be calculated.

This scenario still works even if every interaction is not taken into account because of the reciprocity. For example, the cut-off radius means that the interaction between every atom and every other atom does not need to be calculated, but only the interaction list with a list of nearest neighbours. Copying the value from (*i*, *j*) to (*j*, *i*) becomes a little complicated, because stepping through array values no longer involves incrementing in integer values. Instead, the numbers of the nearest neighbour, its position in the nearest neighbour list, and the position of the original atom on the nearest neighbour's nearest neighbour list need to kept into account! It is thus important to save pointers in the nearest neighbour information array, so that, as the list of neighbours is stepped through, the related array entries can be identified and accessed. Identifying which interactions can be ignored entirely will be discussed in the next section.

## 4.3.5 Using Selected Atoms

In some calculations, like finding the migration energy on a diffusion path, the changes in energy occur in a localised area which includes first- and second-nearest neighbours, so the partial energies of most of the atoms do not need to be recalculated. In this case, a list of atoms close to the affected area is set up, such as shown in Figure 4.12, where it involves

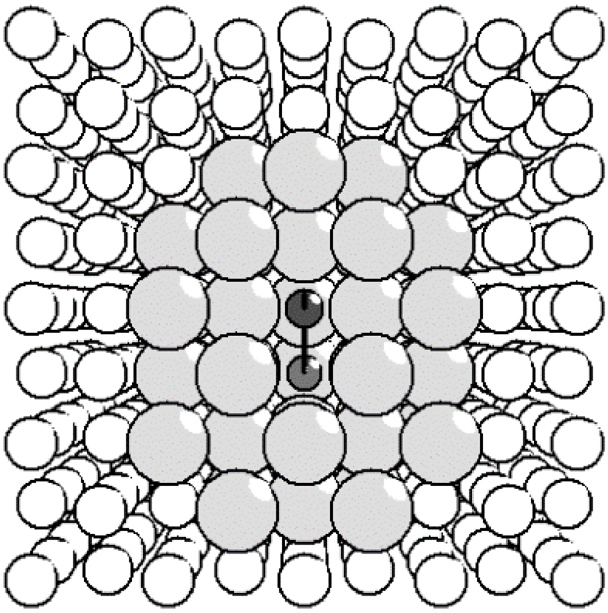


Figure 4.12. When measuring the change in energy along a diffusion path in the crystal, choosing to calculate the changes in energies of only the nearest neighbours (shown here as larger light grey atoms) of the start point and end point (in darker grey) allows for a much shorter calculation time.

the nearest neighbours of the start point and end point of the migration atom. The cut-off distance used was $3.2\sigma$, as before.

## 4.4 Finding the Energy and Force Using the Sutton Chen Potential

Originally, the force exerted by every atom on every other atom would need to be found. Using the cut-off radius, that list can be pared down to a much smaller list of interactions, and if the calculation is localized, all but a select few atoms can be ignored. However, the partial forces for each atom need to be calculated at least once, and this is done using the Sutton-Chen potential.

The potential is quite involved and takes several steps to calculate. Each step is a separate subroutine in a module for the Sutton-Chen calculation. The calculation has two major terms, the $n-term$ and the $m-term$. The $m-term$ also contains the density-like term $\rho_i$, of equation 3.18, which is a sum itself. The calculation starts with calculating the terms

$$\left(\frac{a}{r_{ij}}\right)^{n} \text{ and } \left(\frac{a}{r_{ij}}\right)^{m},$$

where the parameters used depend on whether they are the same element or two different elements. Once these two terms have been calculated, the second can be summed for all nearest neighbours to produce $\rho_i$. These terms can then be used in the partial energy and calculation for each individual atom-atom interaction, which is summed for all nearest neighbours to determine that atom’s energy in the crystal. These terms are also used to find the force of one atom on another, and summing those forces over nearest neighbours produce the total force acting on the atom.

# 4.5 Calculating Velocity and Acceleration

Calculating the velocity and acceleration is straightforward. The equations are followed as set out in section 3.5.1.2, starting with calculation of the new position from equation 3.52, then the new acceleration, and the velocity at half a time step from equations 3.53 and 3.54, and finally the new velocity using equation 3.51, such as the example shown below:

```
g_dbl_atom_velocity(i, 0) = g_dbl_atom_velocity(i, 0) + (1 / 2) * _
                            g_dbl_acceleration(i, 0) * dt
```

There is a correction factor multiplied to the force to ensure the units of the resulting energy are in electron volt. For the complete code of the module, refer to Appendix A.

# 4.6 Temperature Control

## 4.6.1 The Berendsen Thermostat

Temperature control, too, is straightforward.

After the new velocities have been calculated, the kinetic energy is found, and from that, the temperature.

```
kinetic_energy = kinetic_energy + (conversion_factor) * (mass(element(i)) * _
                 velocity_squared(i)) / 2
```

```
Dim temperature As Double

    temperature = (2 / 3) * (kinetic_energy / k_boltzmann)     '=(1.60218E-19)

Return temperature
```

The program uses the Berendsen thermostat, which is really just velocity scaling with a factor which is calculated using equation 3.64.

```
Berendsen_scaling_factor = 1 + (dt / Berendsen_constant) * _
                 ((desired_temperature / system_temperature) - 1)
```

The scaling factor is applied by multiplying it with each component of the velocity for each atom. However, the scaling only reduces or increases the magnitude of the velocities, and does nothing to change their direction.

### 4.6.2 The Flying Ice-Cube

As mentioned in section 3.6.4, one difficulty with using velocity scaling is that there is nothing to prevent all the atoms from locking into position and flying off together into one particular direction, such as the particle illustrated in Figure 4.13 (a). The program does not take the velocity of the reference frame into account, so a frozen particle moving at high velocity still counts as a high system temperature. This phenomenon was named the “Flying Ice-Cube” by Harvey et al. [4].

To solve this problem, one solution is to calculate the particle velocity and rotation and correct for them. Another solution is to make use of Dissipative Particle Dynamics (DPD) [5, 6], which adds dissipative friction forces in such a way as to conserve momentum. The program used isotropically applied DPD noise, and made use of equations 3.66 to 3.73, and equation 3.78. The result is a particle with random velocities, such as the one pictured in Figure 4.13 (b). Use was made of DPD by applying equation 3.69, where the direction of the noise vectors were randomly chosen using the Troschuetz Random Number Generator library [7] according to equation 3.70. The atom pairs to which the noise was applied was also chosen using the random number generator.

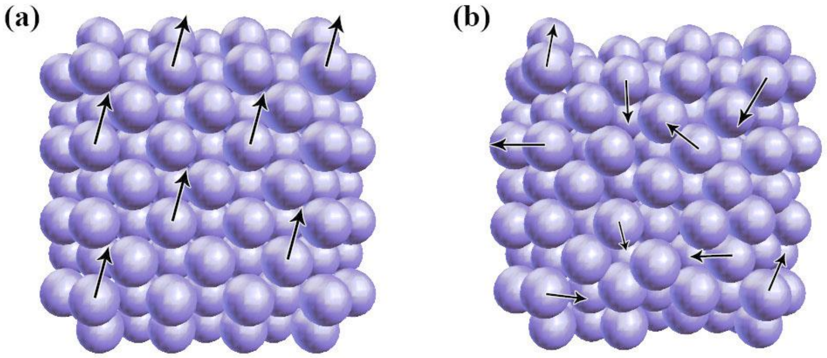


Figure 4.13 (a, b). No matter the velocities of the crystal as a whole, the frozen ice-cube is locked tightly in a 0 K packing (a) with no changes in inter-atom displacement, diving headlong through space. (b) Introducing noise into the velocity vectors of the atoms keeps them from locking into place and rushing off, instead simulating the flux and change of dynamics in a high-temperature crystal.

## 4.7 Surface Measurement

Calculating the vacancy formation energy has two parts; extracting a bulk atom from the centre of the crystal and removing it as far away as possible, and placing the extracted adatom back on the surface. Unexpectedly, the second process is far more involved and time-consuming, and involves multiple problems that require creative solutions.

Whereas extracting the atom from inside the crystal means adding considerable distance to one of its coordinates, placing it on the surface as in Figure 4.14 means mapping a large part of the surface to determine the optimum binding sites. Figure 4.15 shows an area of the surface of a crystal with a (100) free surface, with the third dimension representing the surface energy. The mapped energy is lighter where there are surface atoms, and the gaps between them appear darker. These more negative sites also indicate where an atom would preferentially bind.

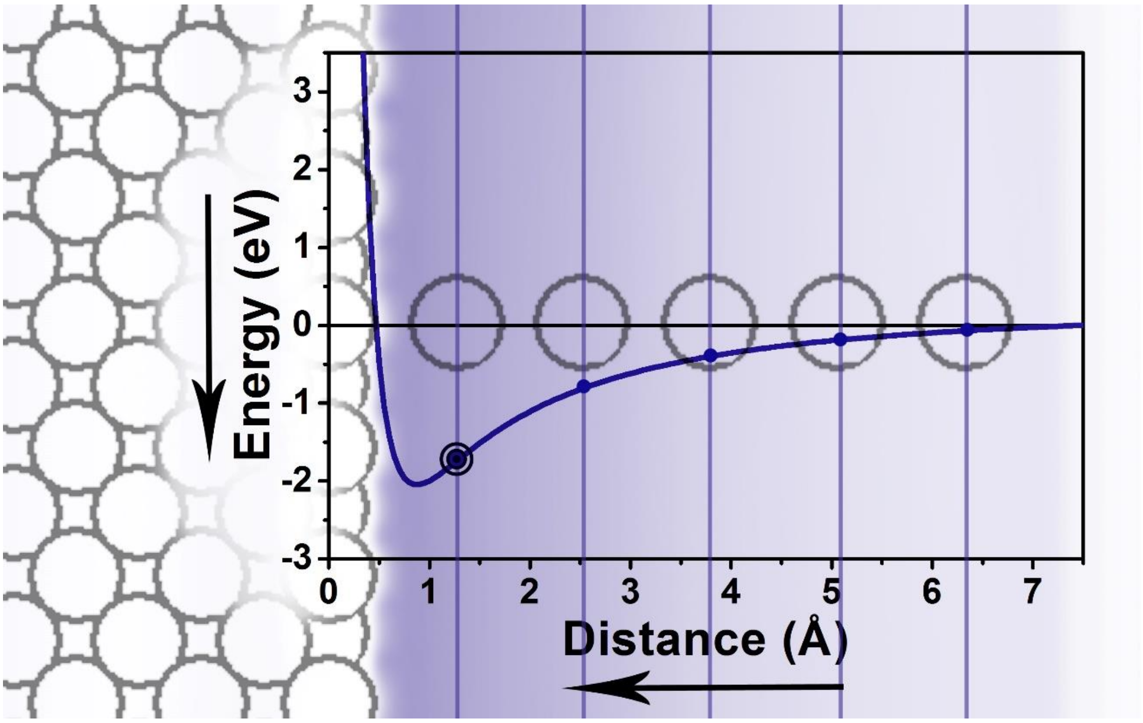


Figure 4.14 shows the potential energy curve produced by bringing a single atom closer to a surface. The atom is likely to occupy the distance from the surface indicated by the potential energy minimum. To find that minimum without stepping the atom through thousands of points requires selective sampling, discussed later.

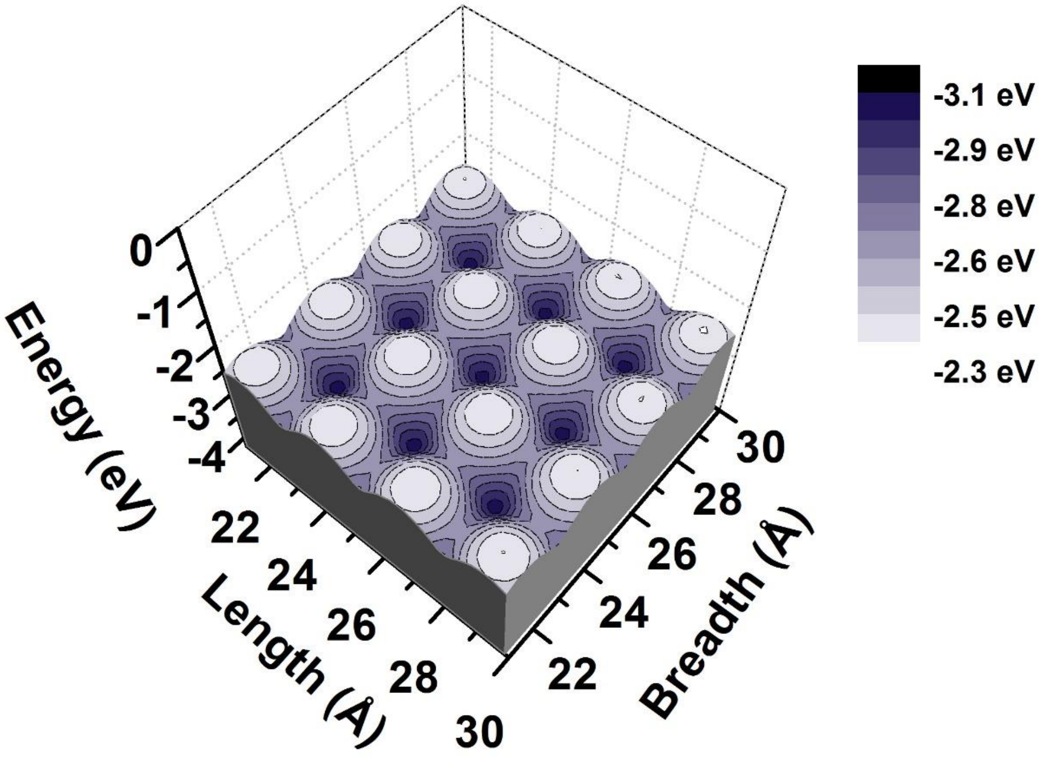


Figure 4.15. The surface energy measured on a (100) surface shows clearly the preferred binding sites in darker colours and hint at the surface atom positions in white.

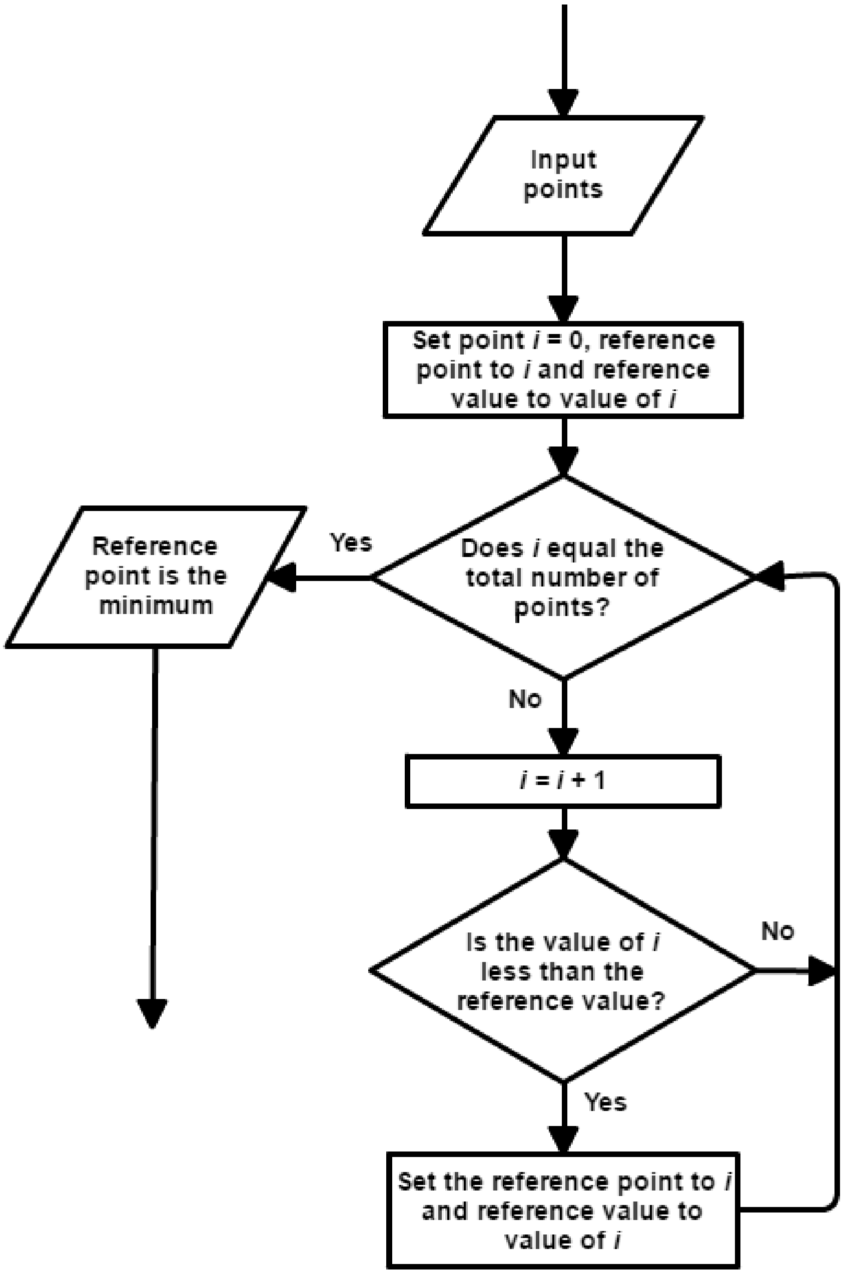

Input points
Set point *i* = 0, reference point to *i* and reference value to value of *i*
Does *i* equal the total number of points?
Yes
Reference point is the minimum
No
*i* = *i* + 1
Is the value of *i* less than the reference value?
No
Yes
Set the reference point to *i* and reference value to value of *i*

Figure 4.16. Finding the minimum of a range of points involves a loop counting through all the values and saving only the smallest value.

In order to find the optimum distance from the surface, and the optimum coordinate of width and breadth on the surface, the absolute minima, and local minima must be determined. The general programming code to find a minimum is set out in the flowchart of Figure 4.16. The process involves a reference value and index for comparison, and then cycling through all the points, comparing them to the reference point to see if their values are smaller than the reference. If they are, the reference value is replaced with the new value and index, and the search continues. For a general minimum, the search is cycled through all points and only

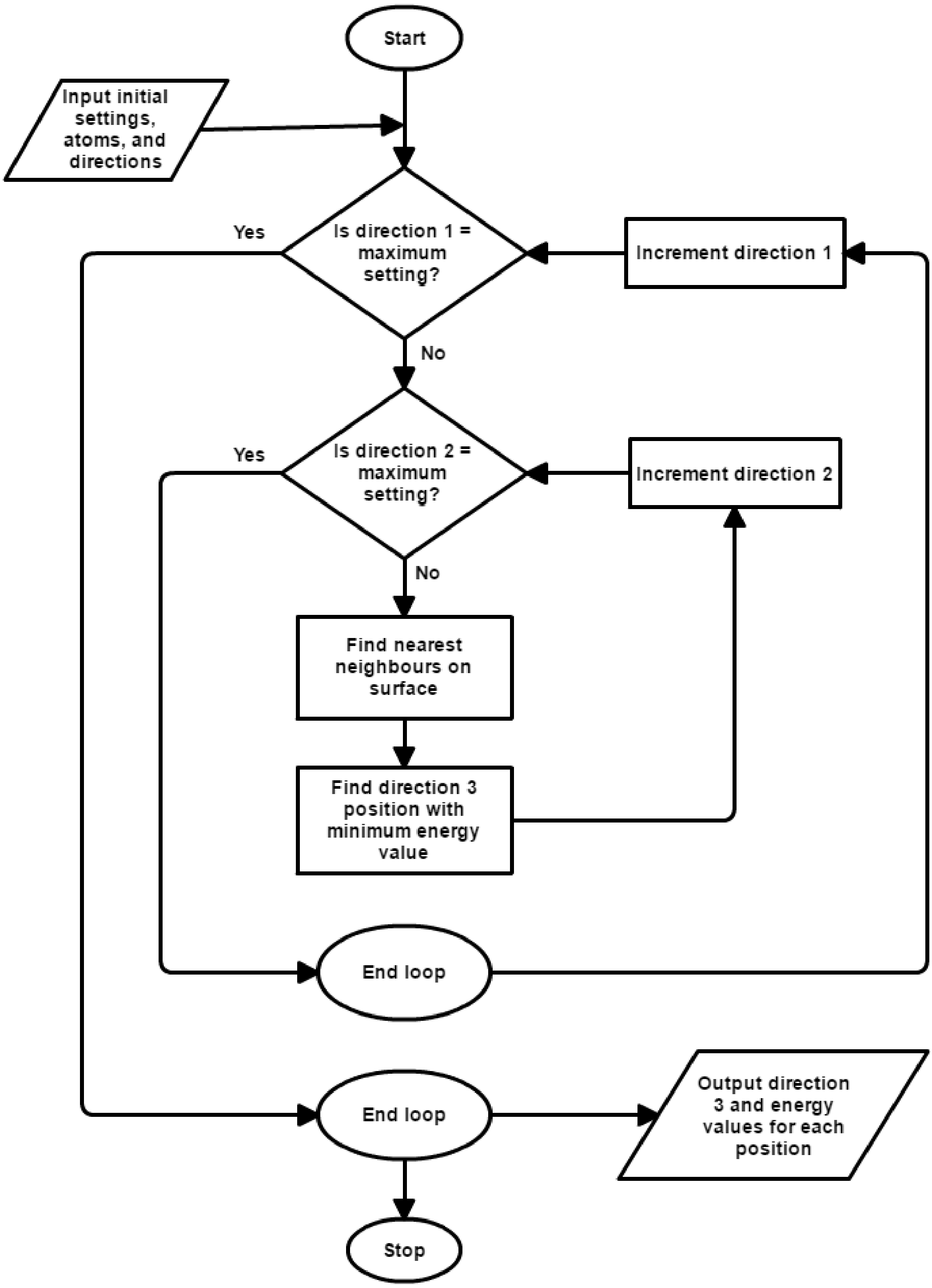

Start
Input initial settings, atoms, and directions
Is direction 1 = maximum setting?
Yes
No
Increment direction 1
Is direction 2 = maximum setting?
Yes
No
Increment direction 2
Find nearest neighbours on surface
Find direction 3 position with minimum energy value
End loop
End loop
Output direction 3 and energy values for each position
Stop

Figure 4.17. To find the minimum binding energy on the surface requires a nested loop stepping through each point of 2 directions while finding a minimum in the third direction.

the extreme minimum is returned. For local minima, a search range must be defined, either cycling through indexes, or using other data such as surface position in a certain radius, to search through. The search range is kept small, and as soon as the next search values are all larger than the reference value, a local minimum is assumed and the search loop is escaped.

Having determined how to find a minimum, this needs to be implemented in the program to classify the surface over an area of the crystal. A simplified version of the program is detailed in the flowchart in Figure 4.17. Note that the chart in Figure 4.17 used “direction 1, 2, 3” instead of “x, y, z”, a measure included to allow 3D investigation of any of the three planes. The x-, y-, and z-coordinates can be substituted for any of the three directions, or even flipped with a negative sign to measure the opposite side of the crystal.

Finding the minimum energy above the surface is repeated in nested loops that step through every point across the surface area in 2 dimensions. However, the optimum height is a little more difficult to find without incrementing through hundreds of steps from sufficiently far away from the crystal not to become caught inside an atom or interstitial position in the crystal. To cut down on time and processing, while quickly isolating the optimum binding distance from the crystal, the Multi-Section Method was used.

## 4.7.1 The Multi-Section Method

A common method to search for a minimum is to use two points, bisect it with a third, then choose the two points with the smallest values and repeat the process; narrowing down the exact position of the minimum more quickly than incrementing through it extremely slowly.

If the position is incremented, then as the atom is brought closer to the surface along a trajectory perpendicular to the surface, the potential energy decreases as the long-range, weak attractive force pulls the adatom, until it approaches the surface too closely and is repelled by the short-range strong repulsive force.

The Multi-Section Method (MSM) represented in Figure 4.18 takes into account that the minimum could lie outside the range of the initial chosen two points, and so extends the range by adding a further two points on either side of the range. It is critical that none of the points fall within the crystal to start, as this can confuse the readings.

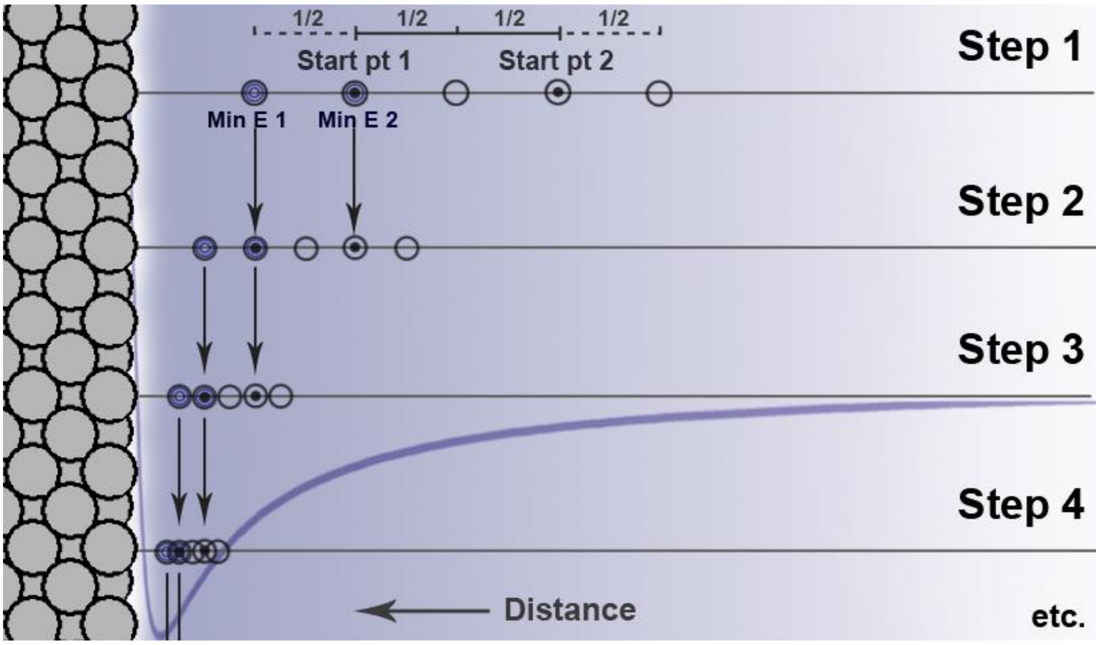


Figure 4.18. A graphical representation of the MSM. Step one shows two starting points, and three neighbouring points chosen, one bisecting and two extending the range. The two points with the minimum energy values are chosen, and the step as repeated with the minima as starting points.

The whole process starts with locating the position of the surface by cycling through crystal atom positions. Once an approximate position is determined, a safety margin is added to the height so the MSM will not be caught in interstitial positions inside the crystal. Next, the adatom, which will scan the surface, is brought closer to the surface until the adatom is in close proximity to another crystal atom, and all atoms within the cut-off radius are added to the nearest neighbour list of the adatom. Now the initial two points of the MSM are used, two points are extrapolated on either side and one point is interpolated in between. Of the five points, the two smallest energies are chosen as the new starting points, and the process is repeated. The MSM is started further away, instead of from the position where nearest neighbours are chosen, as the extrapolated points could land in an interstitial point inside the crystal and give a false minimum. As the methods closes in on the surface minimum, the distances between the search points decreases and there is a smaller chance of an interstitial point being chosen.

## 4.7.2 3D Surface Measurements

For nanocrystals, there are corners and edges where some minima could be missed if only measured from one direction, so a corner is chosen, and measured from three directions.

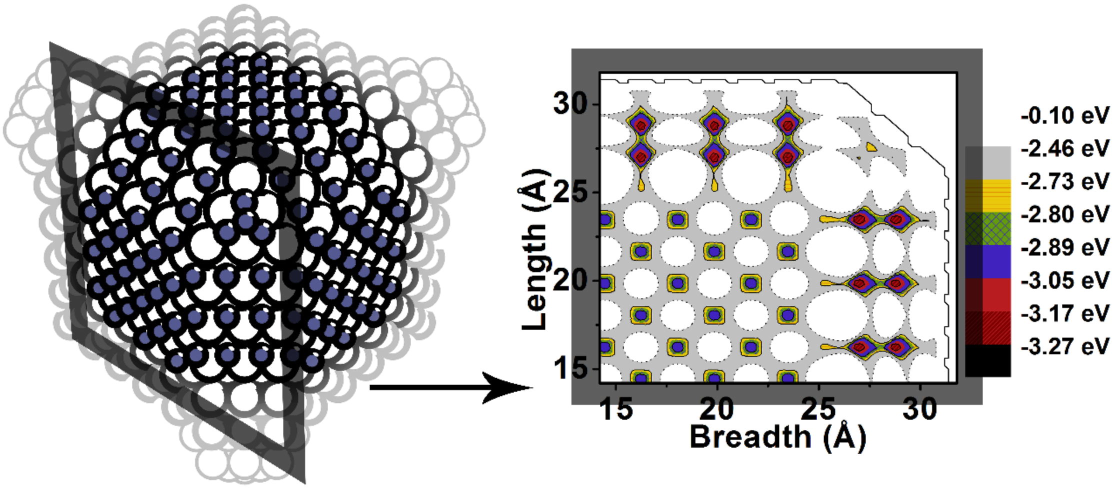


Figure 4.19 shows on the left a 3D view of the corner's potential binding sites on the surface (grey dots) which correspond to local minima on the contour plot of the surface energy on the right.

## 4.7.3 Finding Local Surface Energy Minima

Once the energies across the surface have been measured, the plot is often much like that in Figure 4.19 on the right-hand-side. To translate it back to the left-hand-side of the figure, which has exact positions, where an adatom would preferentially bind on the surface, have been pin-pointed, a further sweep of the surface data must be done to determine local minima. In this case, an arbitrary number of points are defined in a grid across the surface of the contour energy plot, and allowed to search nearby energies by choosing coordinates within a small radius of the starting point. As soon as the search reveals a local minimum, the search at that point on the grid is stopped. The minima are classified by their energies, and can be drawn on the surface as an imaginary adatom to show their position, as in the left-hand-side figure of Figure 4.19.

### 4.7.3.1 Finding Difficult Minima

Figure 4.20 shows a saddle point, where the search for a local minimum gets caught between two minima. Sometimes a slope in one direction is steeper, directing searches toward one minimum and away from another. Other times a minimum is hidden behind outcroppings, something that occurs more frequently during higher temperature runs. Some simple solutions are to run the search using a tight grid with many search points across the surface; to increase the search area slightly once a local minimum is found, to make sure it is not a saddle point; and to search the same point from multiple directions. This is also why 3D measurements are useful.

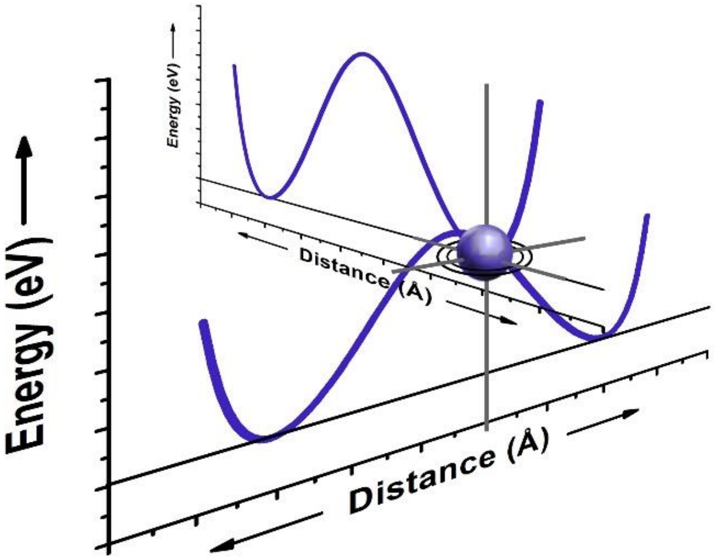


Figure 4.20. Some minima can be saddle points, and prevent a search from finding nearby minima unless the range is extended.

#### 4.7.3.2 Eliminating Double Results

One challenge with 3D measurements is that sometimes the same search yields different results as the true minimum is out of sight from one direction and yields a false minimum instead. It is thus necessary to run the results through a screen which searches through minima in close proximity to each other, but having similar, or different binding energies. Some minima will be listed twice, where one of the values can be safely ignored, while others cannot be ignored. In high temperature runs with surface fluctuations there is a danger of eliminating a true minimum because it has an unusual energy and position, thus manual monitoring along with the automatic screening is also important. Most of the time two points in extreme proximity are a false minimum and true minimum double, and the lowest energy can be kept as the true value.

#### 4.7.3.3 Eliminating Boundary Results

Figure 4.21 presents another challenge. Some minima may be cut off by the measurement range itself, and have atypical energy values that can distort the results. As a general rule, all minima that lie directly on a measurement boundary are ignored.

#### 4.7.3.4 Differentiating Between Different Surface Orientations

The final step in classifying the surface data is differentiating between different surface

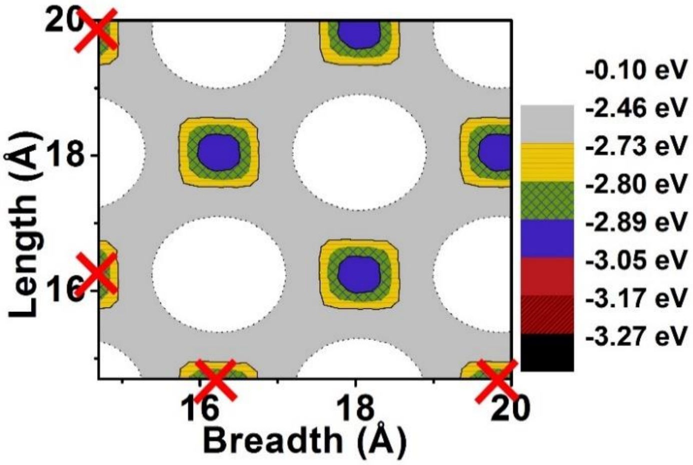


Figure 4.21. Minima found on the extreme edges of the measured ranges run the risk of being atypical values, and should be eliminated.

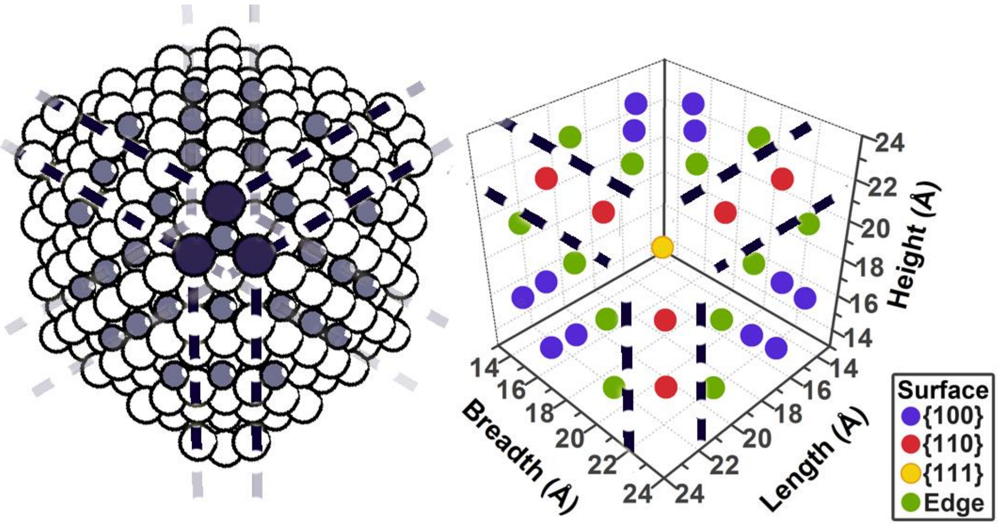


Figure 4.22. Using the (111) point to orient the cube in space, lines can be drawn that delineate the different surfaces along their edges, allowing the classification of surface types and energies.

orientations. On the free surfaces of the bulk crystals this step is not necessary, but on the multi-faceted surfaces of the nanocubes, it is important to differentiate the many different binding points.

In order to find the edges and distinguish the different surfaces, three points on the corners of the (111) surface were chosen. Cutting the cube along the cardinal planes through those three points hems in the {110} surfaces and the {100} faces. The edges of these planes fall on edge-type energies which have atypical binding values. Figure 4.22 shows a representation of this division of surfaces and how the orientations are assigned.

# 4.8 Migration and Segregation Measurements

The segregation energy and migration energy measurements are similar in that they both occur along a diffusion path, plotted along atom coordinates from a chosen surface atom into the bulk to the centre-most atom. Whereas for the migration energy measurements, the next atom in line is removed and the first atom is stepped incrementally to the second position, for segregation energy, the atom positions remain the same but the foreign atom element is exchanged from one atom along the path to another. For both processes the change in total crystal energy is measured, and used to calculate either migration energy or segregation energy.

## 4.8.1 Taking Measurements

### 4.8.1.1 Choosing the diffusion path

Figure 4.23 shows the program flowchart for measuring migration energy. Initially the program structure closely resembles that of the surface measurements in Figure 4.17, in that it has a nested loop, inside which the important processing takes place. In this case though, instead of incrementing through surface coordinates, the program cycles through atoms along a diffusion path, into the bulk of the crystal, in the outer loop. For each atom along that path, the program then cycles through all of the path atom's nearest neighbours in the inner loop. Once a path atom has been chosen, and it's nearest neighbour selected, that nearest neighbour is placed 1 m away from the crystal to create a vacancy, and the chosen atom is drawn into that vacancy at incremental steps, while the crystal energy is measured and saved.

For segregation energy calculations, the processing inside the inner loop is a simple switching of the atom's elements. The first atom along the path has its element changed to that of a foreign element and the crystal energy is measured; then the element is changed back and the next atom along the diffusion path has its element changed to that of an impurity and the energy is saved, and so on. The system energy calculation, where the impurity is at the centre of the crystal, is saved for later comparison.

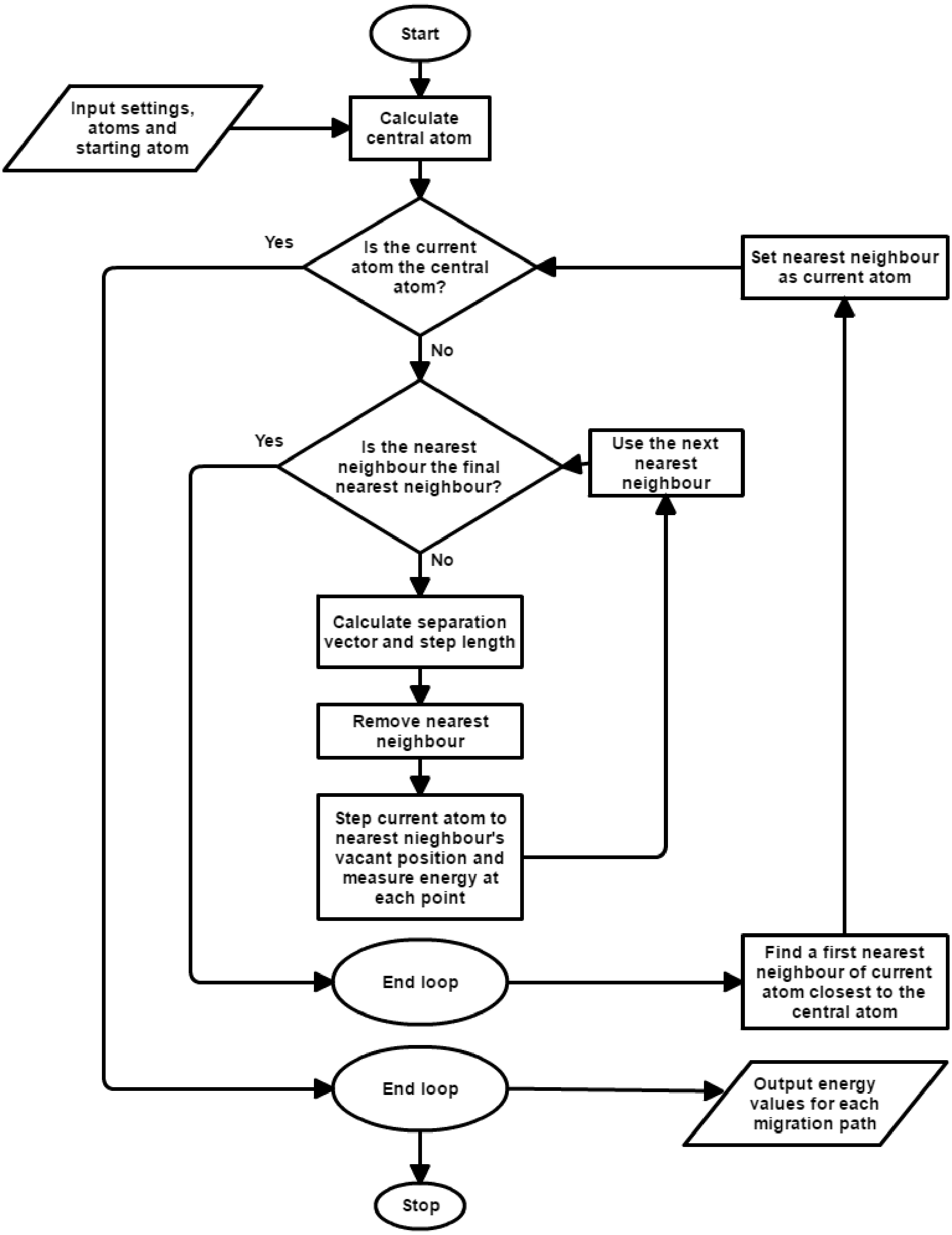

Start
Input settings, atoms and starting atom
Calculate central atom
Is the current atom the central atom?
Yes
No
Set nearest neighbour as current atom
Is the nearest neighbour the final nearest neighbour?
Yes
No
Use the next nearest neighbour
Calculate separation vector and step length
Remove nearest neighbour
Step current atom to nearest nieghbour's vacant position and measure energy at each point
End loop
Find a first nearest neighbour of current atom closest to the central atom
End loop
Output energy values for each migration path
Stop

Figure 4.23. Similar to measuring surface binding energies, measuring the migration energy involves nested loops, though not in directions, but in counting through atoms and their nearest neighbours. The measurement for each path also involves a loop incrementing along the migration path.

Figure 4.24 (a) and (b) shows the process in more detail. Only the nearest neighbours in the plane facing the reader are considered to reduce clutter and make the diagram simpler to understand.

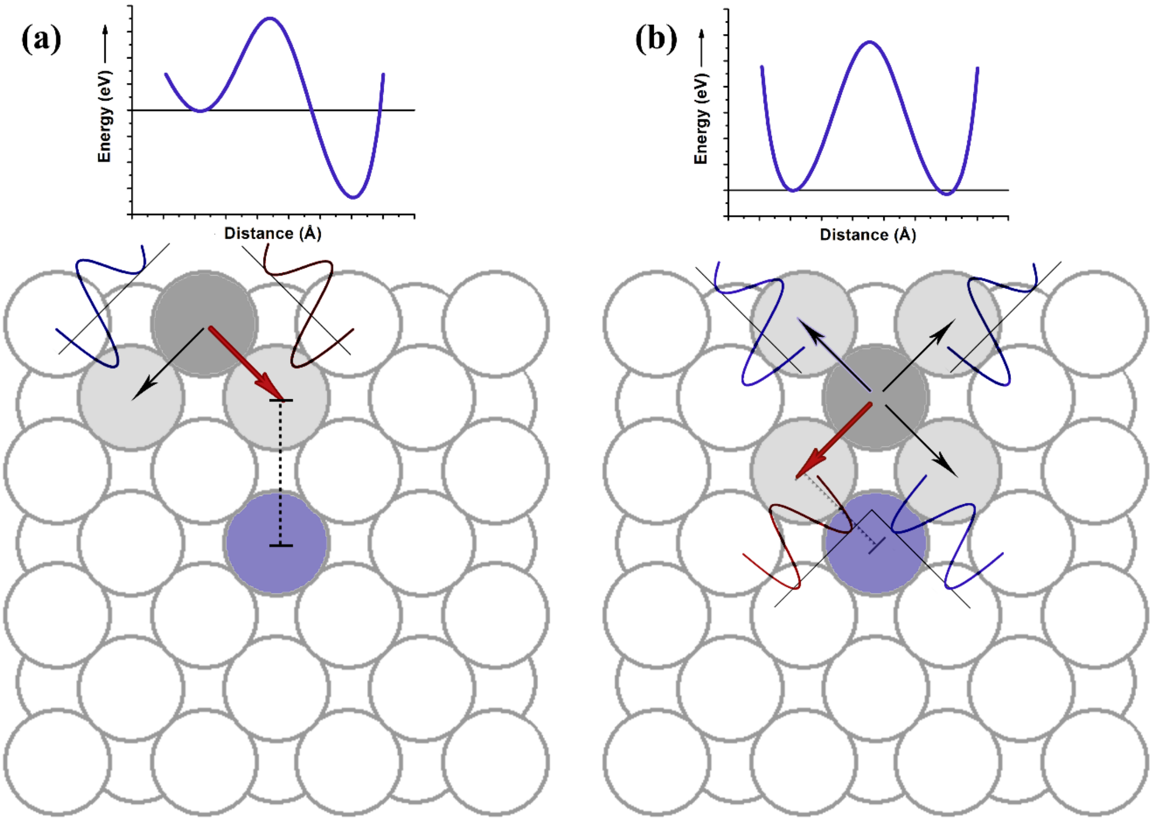


Figure 4.24 shows a visual representation of the process described in the flowchart. Beginning at the surface in (a), the atom is migrated along paths to the position previously occupied by its displaced nearest neighbours. The nearest neighbour closest to the centre of the crystal is chosen to become the migrating atom in (b). All migration energy profiles are saved, but those along the migration path to the centre are stitched together to give a migration energy profile along the diffusion path from surface to bulk.

Figure 4.24 (a) shows the first step of a migration energy calculation, starting with the pre-selected atom on the surface. In the inner loop of the program, each nearest neighbour is removed from the crystal and the surface atom is stepped into the vacancy left behind. Once all the nearest neighbours have had their paths measured, the nearest neighbour closest to the centre is identified; its path is highlighted in red.

This neighbour is chosen as the new primary atom, and the whole process is repeated in Figure 4.24 (b) with a new primary atom stepping to its nearest neighbours. Its neighbour closest to the centre is identified and the whole process is repeated. This laborious process is done with every nearest neighbour so that in the full dataset, the different profiles can be studied, including steps within the same layer, or in the surface layer. This helps to calibrate the results as atoms in the same layer should have similar energies in bulk.

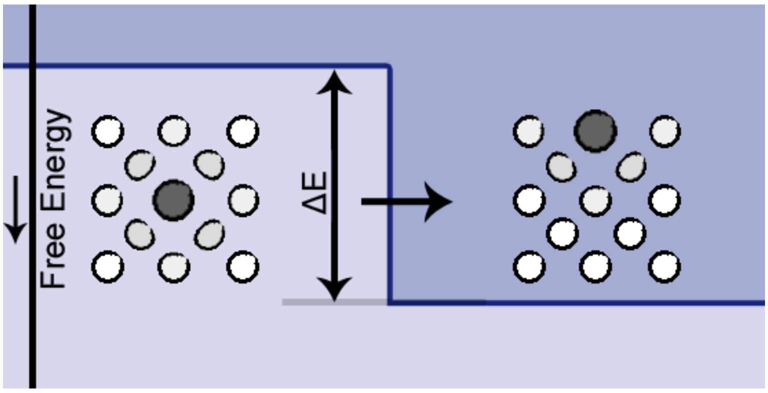


Figure 4.25 shows how the relative position of a foreign atom in the crystal can affect the overall crystal energy. That difference in crystal energy causes a drive to segregate and is equivalent to the segregation energy.

## 4.8.2 Creating an Energy Profile

Once all the migration energy profiles along multiple paths have been measured, the paths highlighted in red in Figure 4.24 are isolated using the atomic numbers of each primary migration atom and the nearest neighbour which becomes a vacancy. These are stitched together end to end, though which end is which also depends on whether the migration path is following the migration of the atom, or that of the vacancy.

For the segregation energy, the only thing to stitch together is the series of system energy measurements done as the element of each successive atom was changed to a foreign element.

### Extracting the Energy Difference

For segregation energy, the final step is to compare the range of energies with the total crystal energy where the foreign element is buried deep inside the centre. The energy difference is an indication of whether the crystal has an overall lower energy with the foreign element in bulk, on the surface, or somewhere in between.

For the migration energy, the energy differences are more complex as the process involves creating a vacancy. Thus, the reference energy used to zero the measurements is the crystal state where one atom is in a binding position within the centre of the crystal and another is removed to create a vacancy. This energy is deducted from the entire profile, and shows whether a vacancy closer to the surface creates a more or less stable crystal state, and whether the atom moving between two positions increases or decreases crystal energy compared to this default state.

## 4.9 Conclusion

The program and its difficulties are varied and involved, providing a challenge in measuring known and unknown quantities in new ways. However, in the next chapters the data generated by the simulation algorithms will be explored to discover if the program can replicate experimental results found in literature. The effects of new configurations, such as change in crystal size or system temperature, will also be investigated.

## 4.10 References

[1] Y. H. Wang, P. L. Chen and M. H. Liu, *Nanotechnology* **17** (2006): 6000.

[2] Y. Sun and Y. Xia, *Science* **298** (2002): 2176.

[3] L. A. L. Wessels, *Masters Project* (University of the Free State, 2012), C# code.

[4] S. C. Harvey, R. K.-Z. Tan and T. E. Cheatham III, *Journal of Computational Chemistry* **19** (1998): 726.

[5] S. A. Moga, N. Goga and A. Hadar, *Materiale Plastice* **50** (2003): 196.

[6] Y. Zhao, “Brief introduction to the thermostats,” accessed February 16, 2014, http://www.math.ucsd.edu/%7Ey1zhao/ResearchNotes/ResearchNote007Thermostat.pdf.

[7] S. Troschuetz, (May 29, 2007). *Troschuetz Random Number Generator,* .NET Framework Class Library, C# Code.

# Chapter 5 - Confirmation of the Model Using Bulk Crystals

## 5.1 Introduction

The modelling software was developed with the Sutton-Chen potential, Velocity Verlet scheme, and Berendsen thermostat discussed in Chapter 3. Using the periodic boundaries described in section 4.3.3, bulk materials could be simulated. Experimental studies are often conducted on large single crystals. Thus, validating the voracity of the model is best done by comparing calculations to bulk values.

The cohesion energy is defined as the energy required to be added to a crystal to create atoms at infinite separation in their ground state at 0 K [1]. Cohesion energies are well established and form a good standard for comparison.

The bulk vacancy formation energies ($E_v$) were calculated for Al, Ni, Cu, Pd, Ag, and Pt single crystals with a molecular dynamics simulation that made use of the Sutton-Chen many-body potential. The monovacancy formation energies for single crystals with the surface orientations of (111), (100) and (110) were calculated at temperatures ranging from 0 K to 1000 K or to below the melting temperatures of some of the elements. As the calculation only investigates a single monovacancy-adatom pair, all energies given in electron Volt (eV) are per atom, even if not explicitly stated. In this chapter, temperature and surface orientation dependence of the vacancy formation energy in bulk FCC crystals are investigated.

## 5.2 Method and Calculations

The six types of FCC metals were simulated as crystals with exposed surface orientations of (100), (110) and (111) at different temperatures, such as those seen in Figure 4.3 from section 4.1.1. Molecular Dynamics simulations were run for each metal for a range of temperatures from 0 K to 1000 K in 50 K steps where the simulated crystal was not melted. For the purposes of this study investigation was limited to where premelting was not observed in surface layers. Although disordering such as surface roughening was observed at high temperatures in some metals as a result of anharmonic vibrations, these results did fall inside the scope of the study, provided the crystal surface did not premelt. The results would fall outside the scope of this study if the crystal surface layers became amorphous and a crystal packing structure could no longer be identified.

Perfect crystals were simulated for each metal with 1584 atoms arranged in 12×12, 11 layers deep, to minimise internal effects of the surface. The crystal structures were allowed to relax for 20 000 steps. Snapshots of the crystal state were taken at time intervals of 1 000 steps during which the crystal energy was monitored. To establish the validity of the current calculation software, the cohesion energy was calculated for each of the three crystals.

The cohesion energy in the crystal is the energy needed to scatter all the atoms in the crystal infinitely far apart, and is essentially a measure of how stable the crystal is. Using equation 3.22 from section 3.3.3, this can simply be calculated by dividing the total crystal energy ($U_{perfect}^{total}$) by the number of atoms in the crystal $n$:

$$E_{coh} = U_{perfect}^{total} / n.$$

After confirming the comparative calculating capabilities of the model using cohesion energies, the vacancy formation energy was calculated for Schottky defects. In this study the particular case of surface-adatom/vacancy pair formation was studied. A vacancy was created by extracting an atom from the middle of the crystal, removing it to a distance of 1 m from the crystal to approximate infinity where interactions between the adatom and the crystal become negligible. The vacancy in the middle of the crystal was created far enough from free surfaces so that the vacancy was not affected by the crystal's surface. Atoms surrounding the vacancy were not allowed to relax. The extracted atom, which became the adatom, was then added to the surface. The surface-adatom energy was calculated for the system where the adatom occupied the ideal bonding site on the crystal surface. Using likely bonding sites, the average surface binding energy for each crystal was determined for each of the various temperatures.  From the energy measurements described above, the vacancy formation energy was calculated as described in section 2.2.3.

The perfect crystal-, atom extraction- and surface-adatom binding energies were measured for several snapshots of the crystal state over time. At higher temperatures, the crystal state tends to vary due to the anharmonic movements of the atoms in the crystal. By averaging the different crystal states calculated over time, a better average of the energy states for each temperature was obtained.

## 5.3 The Cohesion Energy in Bulk FCC Crystals at 0 K

Perfect single crystals of FCC Al, Ni, Cu, Pd, Ag, and Pt were simulated at 0 K using the SC potential to calculate the total crystal potential energy. Using equation 3.22, the cohesion energy for each element was evaluated and compared to literature values (Table 5.1). All the values calculated for cohesion energy compare well with established literature values reported by Kittel (2005) [1]. This shows that the SC potential accurately models the atomic binding energies in the crystals simulated.

Table 5.1 Cohesion energy as calculated for Al, Ni, Cu, Pd, Ag, and Pt and compared to values from literature [1].

| Element | Calculated (eV/atom) | Literature (eV/atom) |
|---|---|---|
| **Al** | 3.23 | 3.39 |
| **Ni** | 4.24 | 4.44 |
| **Cu** | 3.34 | 3.49 |
| **Pd** | 3.81 | 3.89 |
| **Ag** | 2.82 | 2.95 |
| **Pt** | 5.75 | 5.84 |

## 5.4 The Vacancy Formation Energy in Bulk FCC Crystals

After calculating and comparing cohesion binding energies, the vacancy formation energy was calculated using equation 2.14:

$$E_v = E_{extr} - E_{surf}.$$

The vacancy formation energy depends on two energies: the strength of the bond of an atom in the bulk, and the strength of the bond of an atom on the surface.

## 5.4.1 Surface Orientation Dependence at 0 K

The influence of the surface orientation on $E_v$ was investigated in the following section.

### 5.4.1.1 The Extraction Energy

The vacancy formation energy was calculated for crystals with the different surface packing structures (111), (100) and (110) shown in Figure 4.3. An atom was extracted from the perfect bulk crystals to obtain the extraction energy $E_{extr}^{(\mathrm{Bulk})}$ using equation 2.15:

$$E_{extr} = U_{perfect}^{total} - U_{vacancy}^{total}.$$

The extraction energy was determined from the difference between the perfect crystal and the crystal containing a vacancy where the extracted atom was removed far enough away for zero interaction with the crystal. At intervals of 1 000 steps during the simulation, the crystal state was saved, producing a "freeze-frame" that captured the atom positions, velocities and energies.

The crystal energy was determined for each freeze-frame. The energy was then determined for the same crystal freeze-frame after a vacancy had been created, and finally the surface of that crystal state was characterized.

Although the extraction energy was calculated for crystals packed with a (111), (100) and (110) surface orientation, there is no difference in the energy values for the different orientations, i.e. $E_{extr}^{(111)} = E_{extr}^{(100)} = E_{extr}^{(110)}$. For example, the extraction energy calculated for a single Cu atom was 4.32 eV, and 3.90 eV for Al, irrespective of surface orientation. This is as expected, as it shows that the simulated crystal was big enough, so that the vacancy in the middle of the crystal was not affected by the different surface orientations.

### 5.4.1.2 The Surface Binding Energy

Each crystal freeze-frame, or snapshot, taken in order to calculate the extraction energy also had its surface characterized using equation 2.16:

$$E_{surf} = U_{surf}^{total} - U_{vacancy}^{total}.$$

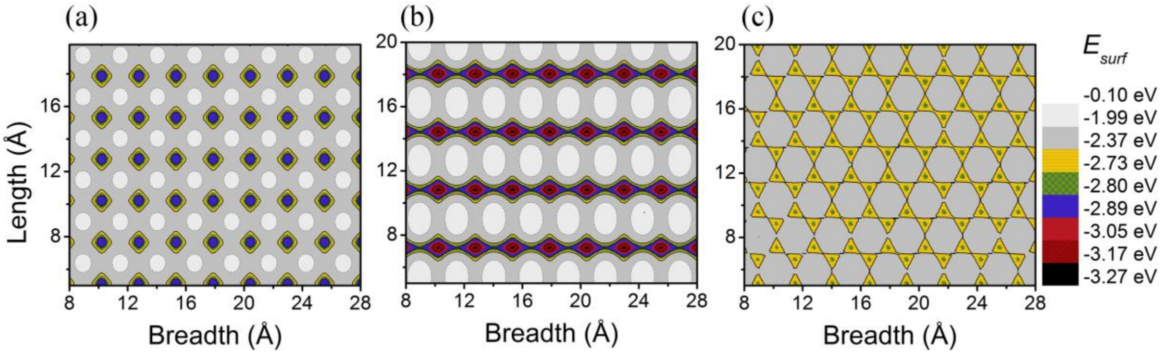


Figure 5.1 (a-c). The contour map shows the surface-adatom binding energy on, from left to right, the Cu(100) surface in (a), the Cu(110) surface in (b) and the Cu(111) surface in (c). Preferential bonding sites are coloured in the contour plot and show where the adatom is most strongly attracted to the surface. The (110) contour plot shows sites with the strongest bonding sites in red and (111) the weakest in yellow.

Figure 5.1 shows contour plots of the Cu(100) surface in (a), the Cu(110) surface in (b) and the Cu(111) surface in (c). Each colour on the plot corresponds to a different magnitude of surface-adatom binding energy.

Different colours show preferential binding sites as the adatom energy reaches the greatest minimum at these points. Averaged over these preferred binding sites, Cu(110) at 0 K has an adatom surface binding energy of 3.24 eV, whereas Cu(100) has an adatom energy of 3.00 eV and Cu(111) has the lowest adatom energy at 2.81 eV.

These adatom energies are due to the packing structure and surface orientation. The (110) surface has ridges and deep troughs, resulting in an uneven surface. Adatoms can bind deeply in troughs and as a result of the deep binding position, adatoms can form bonds with more atoms near the surface which results in a larger surface-adatom binding energy. Conversely the (111) surface is the most closely packed surface orientation, offering shallow binding sites where the adatom can only weakly bind to a few surface atoms it comes into contact with, so that the surface-adatom binding energy is significantly lower. The relationship that was observed between the adatom energies is $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$.

### 5.4.1.3 The Vacancy Formation Energy

The vacancy formation energy was evaluated for Al, Ni, Cu, Pd, Ag, and Pt for each of the freeze-frames of the crystal at intervals in the run-time and then averaged over time to describe more accurately the various crystal states. The results for the vacancy formation energies calculated at 0 K are detailed in Table 5.2.

The calculated values are compared to various values from literature, both theoretical, and experimentally obtained by positron annihilation spectroscopy. The surface orientation dependence of the vacancy formation energy is clearly shown and shows a relationship of $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$. This corresponds to the results obtained by Terblans [2, 3]. The implication of this relationship is that the activation energy needed to create a vacancy-adatom pair in the bulk in a (110) surface orientation FCC crystal is less than the activation energy needed for the same process in a crystal with a (111) surface orientation. The vacancy formation energies compared in Table 5.2 show good correlation between the literature values and the average vacancy formation energies. Values from literature often correspond best with the vacancy formation energies calculated for the (100) surfaces.

Table 5.2 The results for each metal are detailed below. Results for these metals from previous studies are also detailed; bulk values with surface orientation dependent values are noted below the results calculated in this study, in the relevant surface orientation column. Bulk values without a surface orientation component are noted in a separate column. All values are per atom.

| **Element** | **110<br>(eV)** | **100<br>(eV)** | **111<br>(eV)** | **Lit.<br>(eV)** |
|---|---|---|---|---|
| **Al** | 0.59±0.05<br>0.37 [3] | 0.64±0.02<br>0.50 [3] | 0.78±0.01<br>0.66 [3] | 0.69±0.03 [4] 0.68 [5]<br>0.56, 0.60, 0.73 [6]<br>0.66, 0.68 (experimental) [7] |
| **Ni** | 1.38±0.02 | 1.68±0.01 | 1.91±0.01 | 1.67, 1.78 [8]<br>1.43-1.94 [9]<br>1.55-1.88 (experimental) [9] |
| **Cu** | 1.08±0.02<br>1.07 [2] | 1.33±0.01<br>1.34 [2] | 1.51±0.01<br>1.54 [2] | 1.29±0.02 [10]<br>1.31±0.05 (experimental) [11]<br>1.42 (experimental) [7] |
| **Pd** | 1.49±0.01 | 1.80±0.01 | 2.09±0.01 | 1.71, 1.70 [12]<br>1.44 [13]<br>1.85 (experimental) [7] |
| **Ag** | 1.20±0.01 | 1.47±0.01<br>1.04 [15] | 1.65±0.01<br>0.9 [15] | 1.16±0.02 [10]<br>1.06, 1.13 [14]<br>1.31 (experimental) [7] |
| **Pt** | 1.41±0.07 | 1.59±0.03 | 1.96±0.02 | 1.68 [13]<br>1.66, 1.62 [16]<br>1.35 (experimental) [7] |

## 5.4.2 Temperature Dependence in Cu Crystals

This section investigated the effects of temperature on $E_v$, including it's the effects on surface orientation dependence.

### 5.4.2.1 The Extraction Energy

To investigate the effect of temperature on these calculated energy values, the same calculations detailed above were repeated with crystals that were simulated at 0 K. The process was repeated at different temperatures, ranging from 0 K to 1000 K in 50 K steps, with the exclusion of where premelting was observed in Al from 400 K and in Cu at 900 K. The results can be seen in Figure 5.2.

The SC potential used to simulate the Al crystals showed melting of the surface at lower temperatures than the other metals, especially considering the expected melting temperature. Al is the lightest of the elements simulated in this study. Although the melting temperature of Al is found at 933.47 K, melting of the crystal was found to occur at temperatures as low as 400 K. This underestimation of the melting temperature in Al has also been observed before in thermodynamic studies of Al clusters and bulk simulations [17-21]. The Al surface was further found to premelt at temperatures below its melting temperature [22]. For the purposes of this study Al crystals were simulated for temperatures ranging up to 350 K.

Cu also displayed surface premelting. At high temperatures, a large degree of surface disordering was observed, especially for the (110) orientation, which exhibited roughening and the formation of an adlayer. Previous work in literature with an embedded atom model simulating a Cu(110) surface also showed the formation of an adlayer at 900 K through a generation of vacancies, and surface premelting at 80 K below the simulated bulk melting point [23]. Another study used a semi-empirical potential based on the tight-binding method to study the thermal behaviour of low index copper surfaces [24] and it was found in Cu(110) that above 700 K an adlayer formed due to adatom/vacancy formation, which lead to roughening and premelting of the Cu(110) surface. The Cu(100) surface showed disordering above 800 K, whereas the Cu(111) surface was observed to be the most stable with an ordered surface observed for high temperatures. Thermodynamic properties were calculated for Cu in the range between 0K up to 850 K.

As mentioned before in section 5.4.1.1, the extraction energy is independent of surface orientation. The extraction energies obtained over the range of temperatures show a slight decreasing trend over the temperature range of 0 K to 1000 K, as seen in Figure 5.2. This shows a temperature dependence in the extraction energy, with the cost in energy to extract

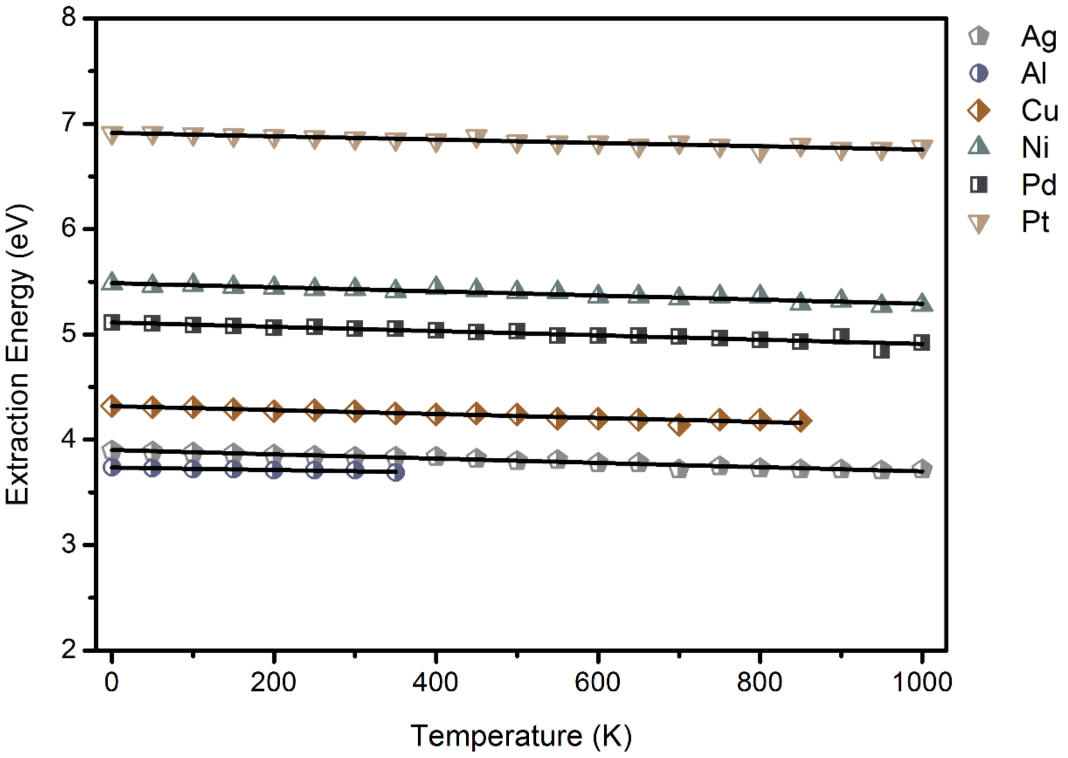


Figure 5.2. The extraction energy for the different metals as calculated at 50 K intervals. An atom is more easily extracted from the bulk of Al than of Pt.

an atom becoming lower as the temperature increases. These extraction energies were found to decrease by as little as 1% for Al, or as much as 5% for Ag. The extraction energy of Cu decreased by 3-4%, that of Ni decreased by 3.5%, Pd by 4% and Pt by 2%.

#### 5.4.2.2 The Surface Binding Energy

It is expected that the expansion of crystals at higher temperatures will affect the extraction energy as seen in Figure 5.2, but it is also important to investigate the effect of temperature on the surface. Figure 5.3 shows contour plots of the different surface orientations of Cu at different temperatures. The (110) surface which has deep troughs at 200 K has fewer strong binding sites as the temperature increases. As a result of the high surface-adatom binding energy and the unstable binding of the atoms in ridges, the surface is more unstable and becomes easily disordered. The disordering leads to the filling of the adatom binding sites in troughs and a decrease in the depth of available binding sites so that the adatom binding energy on the surface decreases. The (111) surface moves from a closely packed stable surface at 200 K to a more open surface at 800 K as a result of disordering from the high temperature. This leads to an increase in the adatom binding energy for the (111) surface.

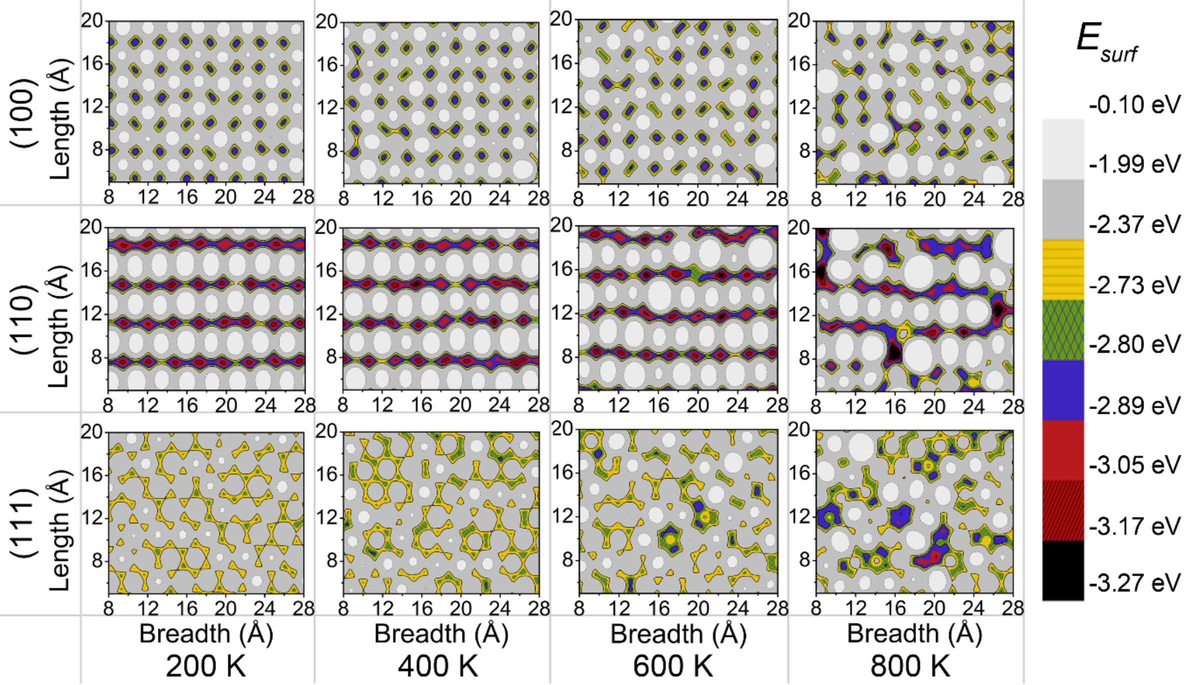


Figure 5.3. These are contour plots of the surfaces of Cu(110), (100) and (111) surface orientations at different temperatures.

### 5.4.2.3 The Vacancy Formation Energy

Using the calculated energies of Cu over temperature, Figure 5.4 sums up the temperature dependency in the FCC crystals studied, as they all show similar tendencies. Figure 5.4 shows a summary of the energies and their tendencies over increased temperature. The extraction energies shown in Figure 5.4 (a) are the same for all three orientations and decrease slightly as the crystal expands from increased temperature. The surface-adatom binding energies shown in Figure 5.4 (b) are not the same. The (110) surface-adatom binding energy is significantly higher than that of (100), whereas the (111) surface yields the lowest value. From eq. 4 this gives: $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$ which can also be seen in Figure 5.4 (c).

At a high temperature where surface disordering and anharmonic vibrations influence the surface packing, the surface-adatom energy, given as positive values, changes as seen in Figure 5.4 (b); decreasing for (110) by 3.5%, barely increasing in (100) by 0.2%, while increasing in (111) by 3%. Coupled with the 3-4% decrease in the extraction energy, this gives an indication of the ratio by which the vacancy formation energy changes over temperature. $E_v^{(111)}$ decreases by 15%, $E_v^{(100)}$ decreases by 12% and $E_v^{(110)}$ only decreases by 4%.

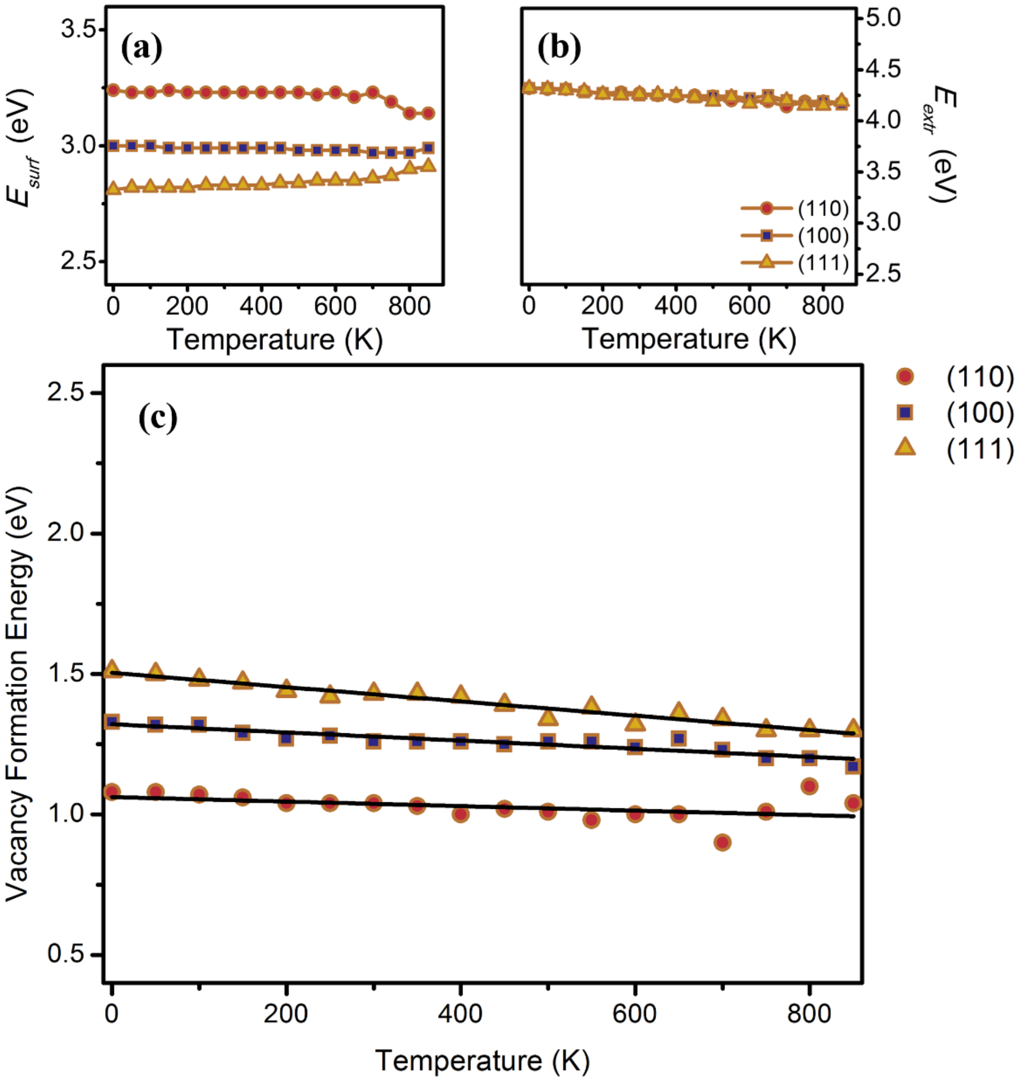


Figure 5.4 (a-c). (a) shows the surface-adatom binding energy which is different for the (110), (100) and (111) surface orientations. The extraction energy for copper (b), is the same for all three orientations.  As a result the vacancy formation energy in (c) is different for the different surface orientations.

## 5.4.3 Summary of Al, Ni, Cu, Pd, Ag, and Pt Crystal Energies

The energies for Al, Ni, Cu, Pd, Ag, and Pt crystals were also evaluated for higher temperatures. Figure 5.5 to Figure 5.7 show the vacancy formation energy for the different surface orientations for all six of the FCC metals. The lowest vacancy formation energy belongs to the metal with the lowest melting temperature. The vacancy formation energy is progressively larger for elements that have progressively higher melting points. Al, with the lowest melting temperature, where premelting is seen at temperatures as low as 400 K, also has the lowest vacancy formation energy.

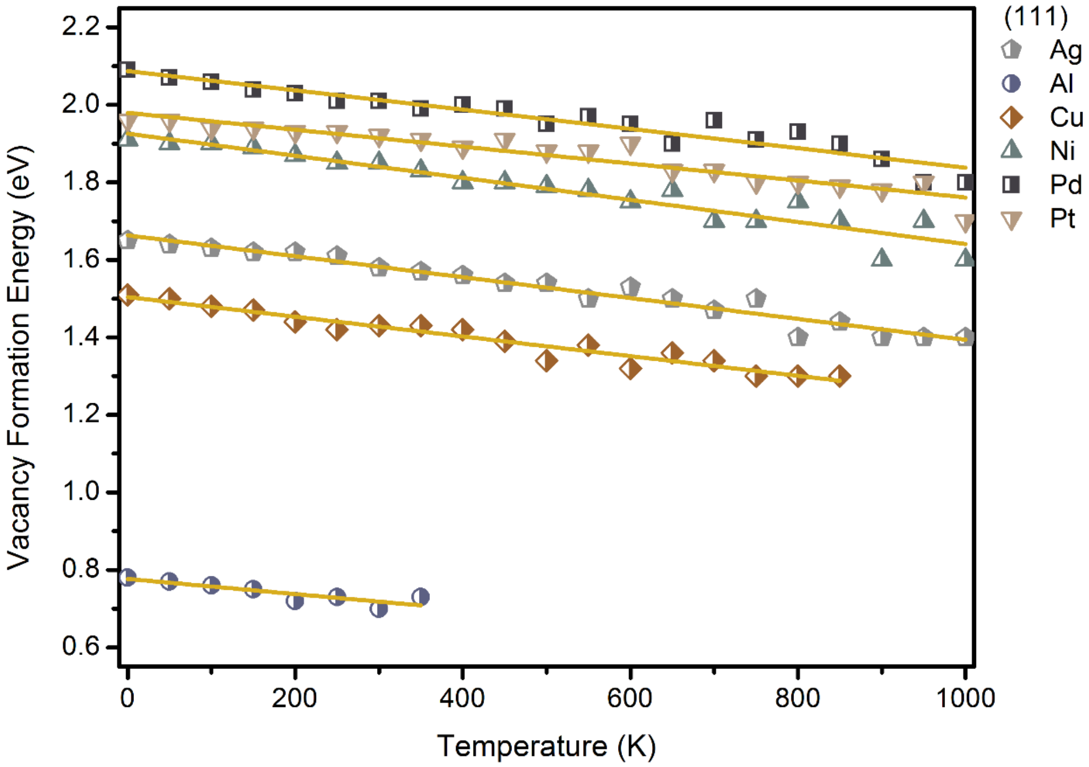


Figure 5.5. The vacancy formation energy for all six FCC metals where the perfect crystal had a (111) surface orientation. Metals such as Al, Cu and Ag with lower melting temperatures were more subject to surface disordering and premelting.

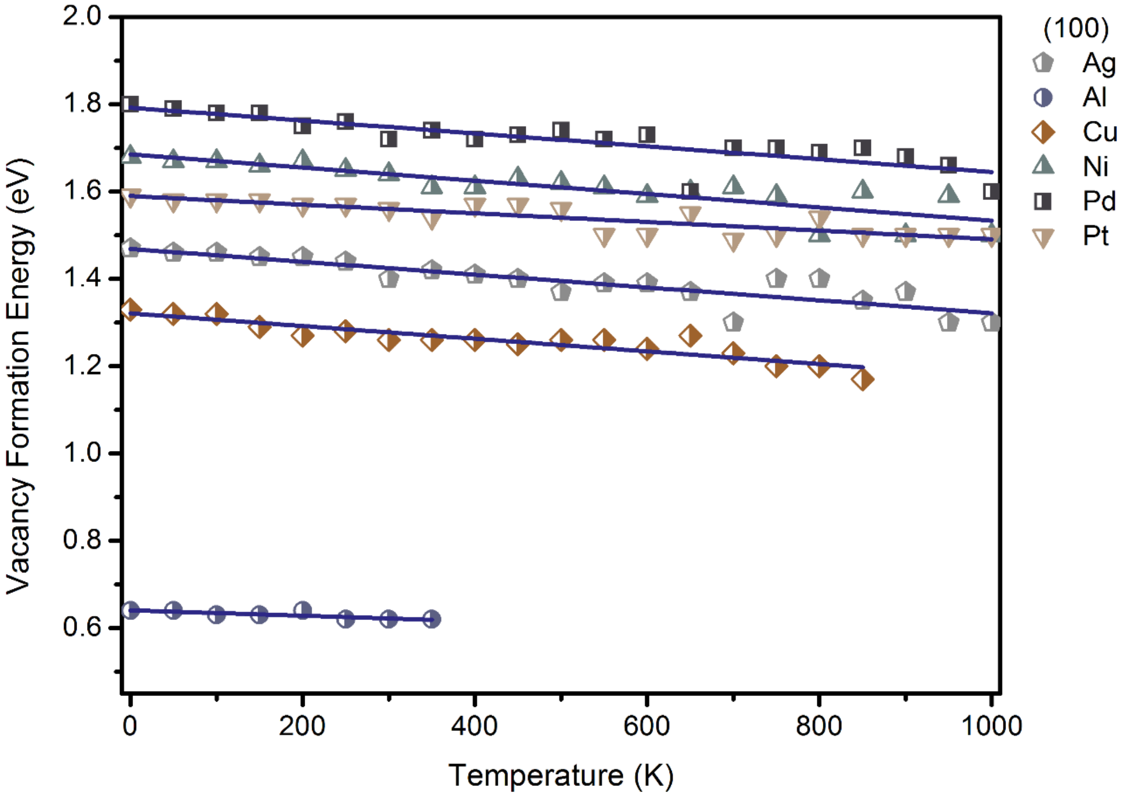


Figure 5.6. The vacancy formation energy for all six FCC metals where the perfect crystal had a (100) surface orientation.

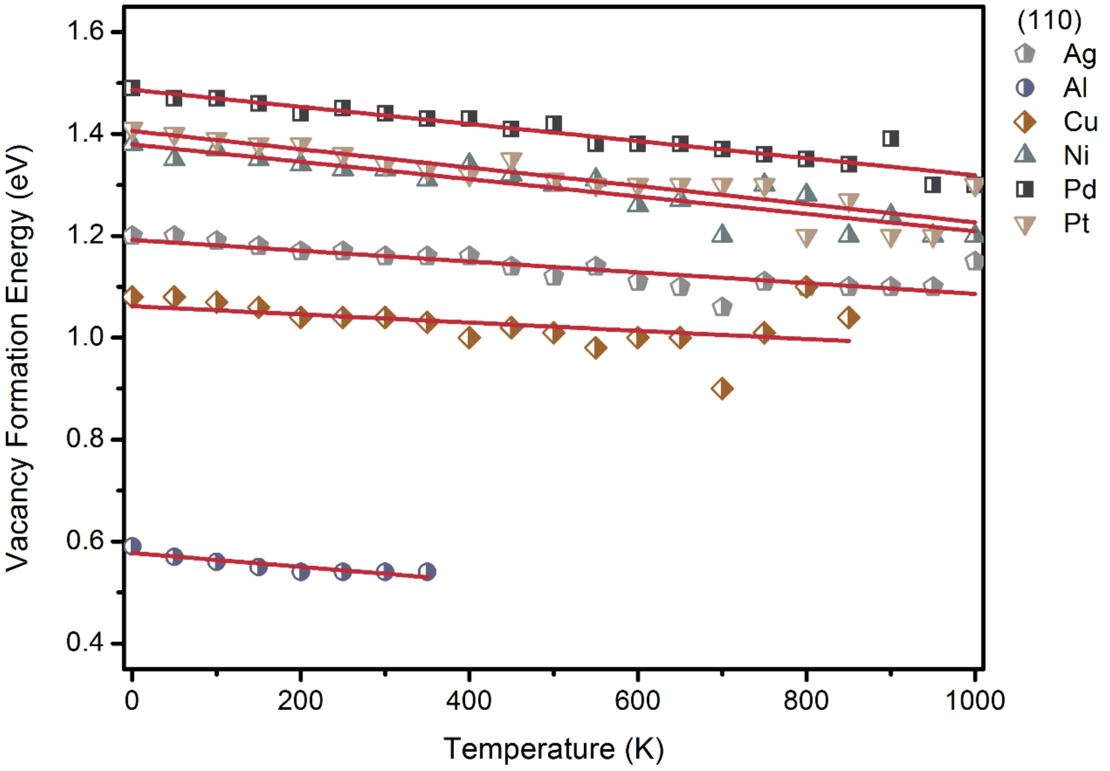


Figure 5.7. The vacancy formation energy for all six FCC metals where the perfect crystal had a (110) surface orientation.

Cu also displays premelting below 1000 K and has the next lowest vacancy formation energy. Ag and Ni have melting points above 1000 K, where Pd and Pt have the highest melting points. Comparing the vacancy formation energies between Figure 5.5, Figure 5.6 and Figure 5.7, it is clear that the vacancy formation energy is the highest for (111) surface orientations and the lowest for (110). In other words: $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$ for all the studied FCC elements. The vacancy formation energy of all three orientations shows a trend of a slight decrease as the temperature increases. As the surface disorders, it is no longer as closely packed and the atoms in the bulk also expand.

With increased temperature, the (111) surfaces shown in Figure 5.5 and (100) surfaces shown in Figure 5.6 become more disorganized. Binding positions for the adatom on the surface allow the adatom to bind with more surface atoms, resulting in an increase of the surface-adatom binding energy for (111) and (100). The result is that the vacancy formation energy becomes lower. In contrast (110) shown in Figure 5.7 displays a much less marked decrease of vacancy formation energy at increased temperatures.

Here, as the surface becomes disordered, the deep binding sites in the troughs of the (110) surface where the adatom bonds with many atoms, become shallower and thus bind with

Table 5.3 The percentage decrease in vacancy formation energy with increased temperature over the temperature range studied for the different surface orientations of the Al, Ni, Cu, Pd, Ag, and Pt crystals.

| **Element** | **110**<br>**(%)** | **100**<br>**(%)** | **111**<br>**(%)** |
|---|---|---|---|
| **Al** | 8.5 | 4 | 6 |
| **Ni** | 9 | 9 | 15 |
| **Cu** | 4 | 12 | 15 |
| **Pd** | 10 | 9 | 12 |
| **Ag** | 4.5 | 13 | 17 |
| **Pt** | 10 | 7.5 | 12 |

fewer atoms. The greater (110) surface-adatom binding energy, due to multiple surface atoms bonding to the adatom, decreases as fewer atoms on the surface are close enough to form a bond, resulting in a less marked vacancy formation energy decrease. A slight decrease is still seen as $E_{extr}$ decreases.

Table 5.3 summarises the decreases of the vacancy formation energy in the different metals for different surface orientations. Elements such as Ni, Cu, and Ag showed anharmonic vibrations at high temperatures which led to some disordering of the crystal surface at high temperatures. This is evident in the larger decrease of $E_v^{(111)}$ of 15-17% and a markedly smaller decrease in $E_v^{(110)}$ of between 4 and 9 %, with the decrease in $E_v^{(100)}$ lying in between that range. This is not observed in Al as the data points with premelting is outside the scope of this study, leaving a small arrangement of points from which to draw a trend.

Ag showed some surface disordering from anharmonic vibrations as it approached higher temperatures. Surface roughening is observed for Ag(110) as low as 950 K, but is not observed for Ag(111) for temperatures of 1000 K or lower, although surface vacancies were seen to form in Ag(111) at 1000 K. Surface disordering was also observed in Ag(100) for temperatures 950 K or higher. Literature studies on surface morphology on Ag(110) from 300 to 1100 K, using reliable, many-body interaction potentials and Molecular Dynamics simulations, show that the surface disorders around 750 K, roughens around 930 K and premelts at about 1000 K [25]. Similar studies of Ag(111) show that the surface does not disorder until 1100 K and does not premelt [26]. In a literature study by Kimura et al. a value of 1.03 eV was obtained for the monovacancy formation energy of Ag [16]. The study used the SC potential but relaxed the restraints on the parameters used. The vacancy formation energies obtained for Ag in this study of 1.20-1.65 eV are slightly larger. It may be that the original SC parameters published by Sutton and Chen [27] are not the ideal fit to describe the vacancy formation energy in Ag.

Pt and Pd did not display any surface disordering in the temperature range that was studied as the temperature range is well below the melting point of Pt or Pd and thus their surface-adatom binding energies were not influenced significantly. Pd and Pt also show a stronger decrease in $E_v^{(111)}$ of 12%, than in $E_v^{(110)}$ where it is 10%. However, the decrease for $E_v^{(100)}$ is slightly larger than the decrease of $E_v^{(110)}$ for Pd and Pt in this temperature range. It may be worth investigating the trend for Pd and Pt closer to their respective melting temperatures.

## 5.5 Summary

The Sutton Chen potential was used to simulate Al, Ni, Cu, Pd, Ag, and Pt FCC crystals with (110), (100) and (111) surface orientations at a range of temperatures. To establish the accuracy of the model, cohesion energies were calculated at 0 K for all the crystals. The calculated cohesion energies compared well with values from literature. The vacancy formation energy for Schottky defects was calculated. It was found that the surface orientation of a crystal influences the vacancy formation energy of the underlying bulk crystal, in that $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$. This is evident in all six FCC metals studied. Some premelting was seen in Al and Cu, and surface disordering was observed in Ag and Ni as well. Where the surface packing becomes disordered, the difference in $E_v^{(110)}$ is less marked as the surface-adatom binding energy decreases, whereas there is a more marked decreasing trend in $E_v^{(111)}$ where the surface-adatom binding energy increases at high temperatures. In Pd and Pt where the melting temperature is significantly higher than the temperature range studied, surface disordering is not observed. A decrease in vacancy formation energy resulting from crystal expansion at higher temperatures was observed for all the vacancy formation energies calculated.

## 5.6 References

[1] C. Kittel, *Introduction to Solid State Physics*, 8th ed. (New York: John Wiley & Sons Inc., 2005).

[2] J. J. Terblans, *Surface and Interface Analysis* **33** (2002): 767.

[3] J. J. Terblans, *Surface and Interface Analysis* **35** (2003): 548.

[4] P. Tzanetakis, J. Hillairet and G. Revel, *Physica Status Solidi (b)* **75** (1976): 433.

[5] N. T. Gladkikh and O. P. Kryshtal, *Functional Materials* **6** (1999): 823.

[6] T. Hoshino, N. Papanikolaou, R. Zeller, P. H. Dederichs, M. Asato, T. Asada and N. Stefanou, *Computation Materials Science* **14** (1999): 56.

[7] Y. Kraftmakher, *Physics Reports* **299** (1998): 79.

[8] P. A. Korzhavyi, I. A. Abrikosov, B. Johansson, A. V. Ruban and H. L. Skriver, *Physical Review B: Condensed Matter* **59** (1999): 11693.

[9] T. Mizuno, M. Asato, T. Hoshino and K. Kawakami, *Journal of Magnetism and Magnetic Materials* **226-230** (2001): 386.

[10] W. Tritshauser and J. D. McGervey, *Applied Physics* **6** (1975): 177.

[11] M. J. Fluss, L. C. Smedskjaer, R. W. Siegel, D. G. Legnini and M. K. Chason "Positron Annihilation Measurement of the Vacancy Formation Enthalpy in Copper," (paper presented in *Proceeding of the Fifth International Conference on Positron Annihilation*, Lake Yamanaka, Japan, April 8-11, 1979).

[12] T. R. Mattsson and A. E. Mattsson, *Physical Review B: Condensed Matter* **66** (2002): 214110.

[13] S. M. Foiles, M. I. Baskes and M. S. Daw, *Physical Review B: Condensed Matter* **33** (1986): 7983.

[14] H. M. Polatoglou, M. Methfessel and M. Scheffler, *Physical Review B: Condensed Matter* **48** (1993): 1877.

[15] K. G. Lynn, "Slow positron studies on single crystals of Ag (100), Ag (111) and Cu (111)," (paper presented in *Proceeding of the Fifth International Conference on Positron Annihilation*, Lake Yamanaka, Japan, April 8-11, 1979).

[16] Y. Kimura, Y. Qi, T. Çağin and W. Goddard III, (1998). "Caltech Asci Technical Report No. 003", caltechASCI/2000.003.

[17] P. Puri and V. Yang, *Journal of Physical Chemistry C* **111** (2007): 11776.

[18] S. Ozgen and E. Duruk, *Materials Letters* **58** (2004): 1071.

[19] P. Stoltze, J. K. Norskov and U. Landman, *Physical Review Letters* **61** (1988): 440.

[20] Y. Qi and P. E. Krajewski, *Acta Materialia* **55** (2007): 1555.

[21] A. Budi, D. J. Henry, J. D. Gale and I. Yarovsky, *Journal of Physics: Condensed Matter* **21** (2009): 144206.

[22] J. G. O. Ojwang, R. van Santen, G. J. Kramer, A. C. T. van Duin and W. A. Goddard III, *Journal of Chemical Physics* **129** (2008): 244506.

[23] R. N. Barnett and U. Landman, *Physical Review B: Condensed Matter* **44** (1991): 3226.

[24] F. J. Resende, V. E. Carvalho, B. V. Costa and C. M. C. de Castilho, *Brazilian Journal of Physics* **34** (2004): 414.

[25] T. S. Rahman, Z. J. Tian and J. E. Black, *Surface Science* **374** (1997): 9.

[26] A. N. Al-Rawi, A. Kara and T. S. Rahman, *Surface Science* **446** (2000): 17.

[27] A. P. Sutton and J. Chen, *Philosophical Magazine Letters* **61** (1990): 139.

# Chapter 6 - Comparative Study of Cu Nanocubes at 0 K

## 6.1 Introduction

After using bulk crystals to validate the model, nano-crystals were studied. The nano particles of FCC Cu with shapes ranging from perfect cubes through to octahedrons, resembling the nanocubes synthesized by Wang et al. [1], were modelled and characterized. Bulk properties in the volume of the crystal, surface energies, vacancy formation energy, $E_v$ and cohesion energies, $E_{coh}$, were investigated for particles simulated up to 5 nm in diameter. This chapter presents the findings of the investigation of the vacancy formation energy shape-, size- and surface orientation-dependency, using the embedded atom potential developed by Sutton and Chen [2]. All energy values are given per atom. Employing the model, properties can be examined in far more detail than is possible in the lab.

Using the surface characterization as well as measurements of the overall cohesion energy of the particles, size and shape dependence was investigated to find the most stable cube, rhombicuboctahedron or octahedral shape for each particle size. Although any particle shape can be produced theoretically, for practical purposes it is best to determine the most preferred forms.

## 6.2 Methods

All calculations were performed at 0 K to eliminate the effects of temperature on energies. The particles were created by first packing a perfect cube with six {100} facets such as the 9 atoms per row, in 9 rows in 9 layers, or $9\times9\times9$ cube shown in Figure 4.4. By cutting away three {111} layers (-3), and {110} layers, it produces the imperfect cubic particle called a rhombicuboctahedron, shown in Figure 4.4 (d), which will be denoted by $9_{-3}$. This particle is roughly 3 nm in diameter. The sizes of the crystals were varied, from $3\times3\times3$

which is roughly 0.7 nm in diameter, to $15\times15\times15$ which measures about 5 nm in diameter and which contains over 12 000 atoms. For each crystal size, the corners and edges were progressively cut away until the particles resembled octahedrons, as can be seen in Figure 4.5. For more information on packing the nano-crystal, refer to section 4.1.2.

For each crystal size, from $3_0$ through to $15_{-14}$, the crystal was relaxed and the energies were measured for 20 000 time steps of 1 picosecond each, before an atom of the crystal was removed and vacancies were created to simulate the Schottky mechanism for vacancy formation. The removed atom was extracted from various depths and placed at a distance of 1 m from the particle to simulate infinity, the energy difference giving the binding energy $E_{extr}$ of this atom in the volume of the nanoparticle. The final part of the vacancy formation energy calculation involved placing the removed atom back on the surface of the crystal to yield the surface binding energy $E_{surf}$. As in Chapter 5, these energies were used to determine $E_v$ using equation 2.14. $E_{coh}$ was also calculated using equation 3.22.

## 6.3 Depth of the Vacancy in Bulk

The effect of the depth of vacancy on the extraction energy was calculated first. All the results calculated thus far were determined by extracting the adatom at the centre of the crystal. To investigate the effect on the vacancy formation energy, the atom was progressively brought closer to the surface in each particle of each size. For each particle, the adatom was extracted progressively closer to the surface. The resulting $E_{extr}$ for the $15\times15\times15$ cube was drawn in Figure 6.1.

Except for the $3\times3\times3$ particle which is less than 1 nm in diameter, all the particles exhibit a similar trend, where the adatom extracted from within the volume of the crystal gives a constant extraction energy value as soon as it is sufficiently deep beneath the surface. This constant value approaches the value found in the bulk, presented in the previous chapter [3]. The extraction energy spikes slightly by a negligible 0.01 eV when the adatom is taken from just beneath the surface, showing the strongest binding energies are achieved at subsurface layers. This is where the vacancy atom is binding to surface atoms which have fewer, stronger bonds. The exception to this trend is the octahedron, where the subsurface spike in extraction energy is up to 0.3 eV, depending where on the surface the measurement is taken.

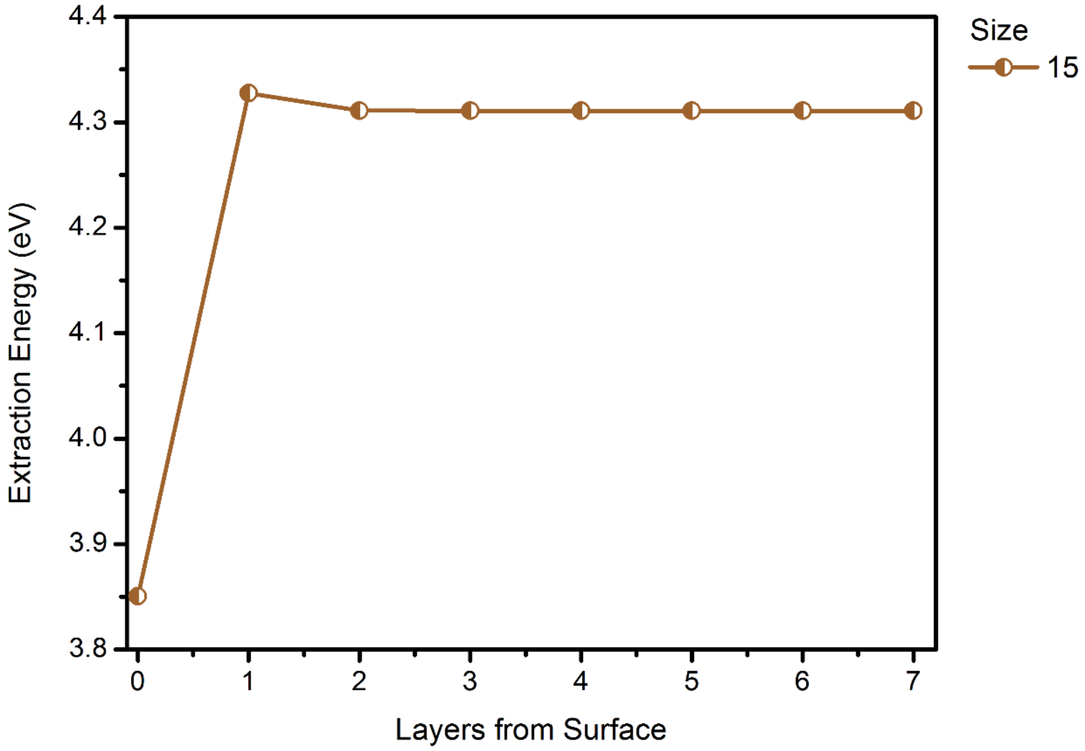


Figure 6.1. Single adatoms were extracted (one at a time) at different depths within a perfect cube of 15×15×15 with a diameter 50 Å, or 5 nm. At each depth the energy needed to extract the atom was measured. 0 Å represents the surface layer and 25 Å represents the centre of the particle. Energy values are per atom.

The adatom extracted from the surface yields the lowest activation energy. This is also dependent on which surface orientation the adatom is extracted from, as this influences the number and strength of bonds with neighbouring atoms. The adatom was always cut away from the middle of the (100) face. When all the edges and corners have been cut away, this atom is found at the tip of a pyramid, making its bonds weak and the subsequent extraction energy low. Cutting the adatom away from the {110} edges or {111} corners, this further influenced the extraction energy of the vacancy atom on the surface. However, it is quickly established that below the subsurface layer, the extraction energy is independent of the surface orientation or particle shape and instead becomes dependent on particle size. As the intermediate steps in subsurface layers do not influence the final vacancy formation energy value, every vacancy was consequently created in the centre of the nanoparticles. For this reason, the number of atoms per row, the number of rows and of layers were odd numbers, giving a convenient obvious central atom.

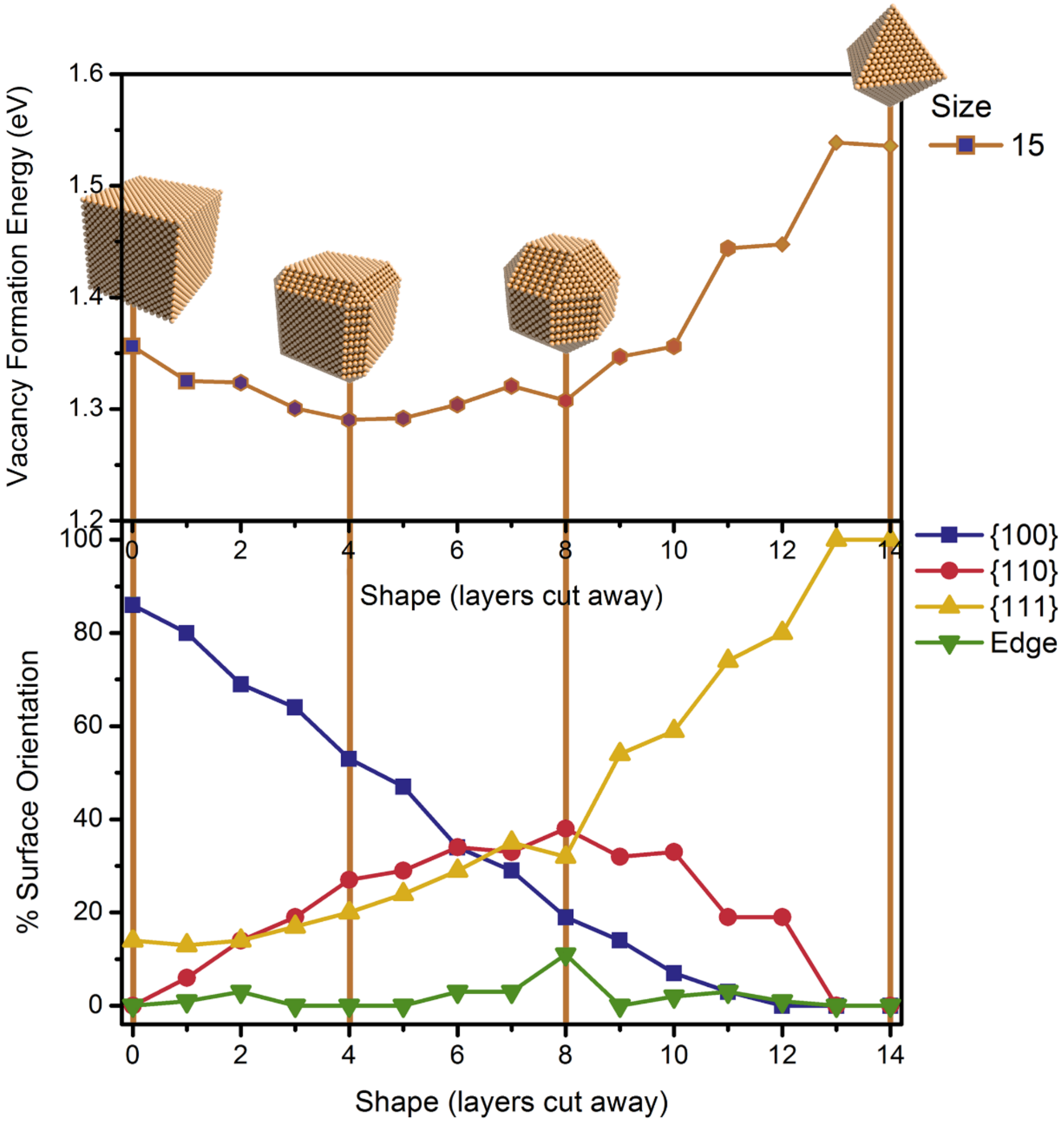


Figure 6.2. The surface of a 5 nm diameter nanoparticle was analysed as its shape was changed, with it starting as a perfect cube at 0 on the x-axis, moving through rhombicuboctahedron [4] to octahedron at 14 on far end. The x-axis shows the variable *i* in the expression $x_{-i}$ which represents the nanoparticle shape produced from shaving *i* layers of atoms from the edges and corners of a perfect $x \times x \times x$ -packed cube. The corresponding vacancy formation energy is represented above each point of surface characterization.

## 6.4 Surface Orientation and Shape Dependence

An investigation into the roles of the extraction energy $E_{extr}$ and surface-adatom binding energies $E_{surf}$ in the vacancy formation energy $E_v$ reveals that where the particle is large enough, $E_{extr}$ remains constant. This extraction energy does change when the particle is so small that the volume atoms become influenced directly by the surface defect. To investigate the effect of the surface orientation dependency on the vacancy formation energy, the overall

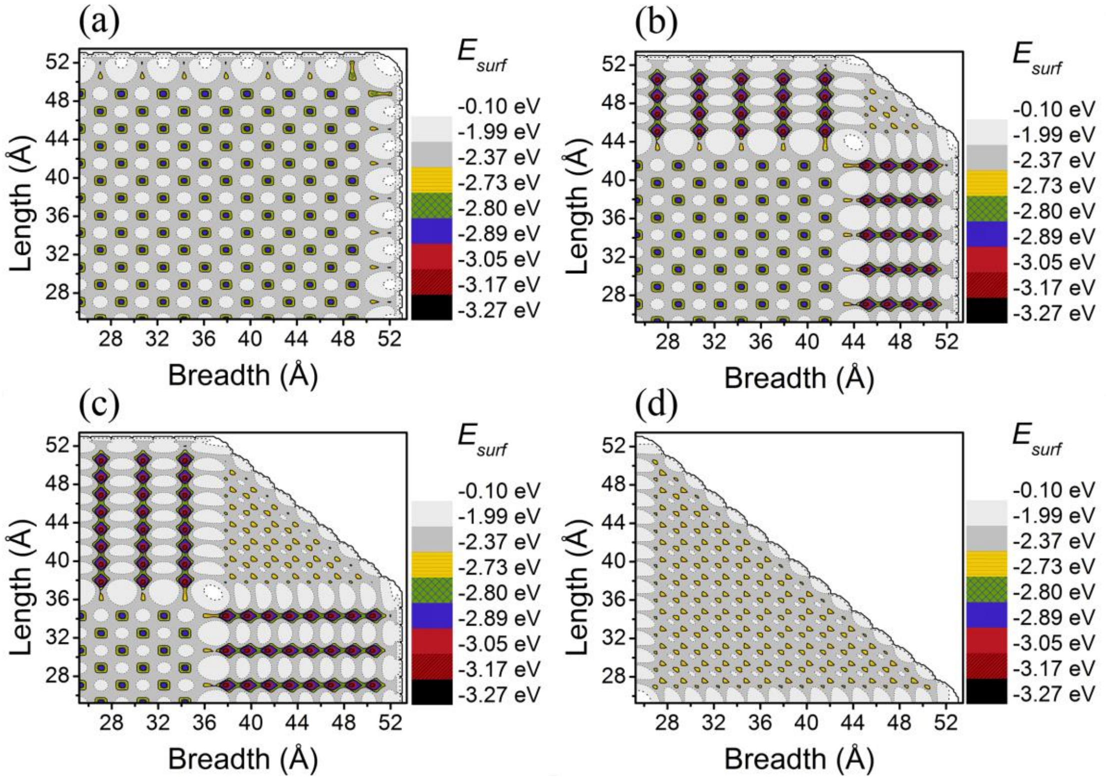


Figure 6.3 (a-d). (a) represents the contour graph of a corner of the surface energy across a Cu(100) face of the perfect cube with diameter of 5 nm, expressed as $15_0$. (b) shows a corner of the rhombicuboctahedron formed from cutting 4 layers from edges and corners to form the $15_{-4}$ particle. (c) Another rhombicuboctahedron with more {110} binding points on the edges, denoted by $15_{-8}$. (d) The octahedron formed from $15_{-14}$.

percentage of each orientation binding point on the surface was drawn as the edges were cut away. "edg" denotes binding points on the surface with unusual binding energies, typically found at the interface between surfaces of different orientations.

$9\times9\times9$ particles up to $15\times15\times15$ packed particles all display similar characteristics. The percentages of the different surface orientations of each shape of $15\times15\times15$, $15_0$ to $15_{-14}$, were plotted in Figure 6.2 along with the corresponding $E_v$. As the {100} surface is cut away, the percentage of {100} surface declines steadily. Both the occurrences of {110} and {111} increase, where the percentage of {110} edges initially rises quickly.

At the shape $15_{-4}$, whose surface is shown in Figure 6.3 (b), where 4 layers have been cut from the edges and corners, {100} and {110}, which have more reactive binding sites, have reached an equilibrium with {111} binding sites. Thus, up to $15_{-4}$ has an increase in {110} surfaces, moderately reactive {100} surfaces and fewer non-reactive {111} surfaces, to produce the lowest vacancy formation energy, meaning that the surface is overall the most reactive.

Up to the particle shape $15_{-8}$, shown in Figure 6.3 (c) where the edge {110} is a maximum percentage, as more layers are cut away, the percentage of less reactive {111} corners increases rapidly at the expense of the more reactive {110} edges, and {100} decreases as well, lowering overall reactivity. Up to a critical point. {111} percentage dips, while "edg" energies spike, which suggests that the intersections between different orientations more strongly affect each other.

At this shape the surface has the most {110} binding points and $E_v$ dips as well, as the deep binding sites on the {110} orientated surfaces are ready binding places for an adatom and good potential sinks for vacancy creation, thus requiring less vacancy formation energy. From this point on {100} continues to decrease while percentage {111} increases stepwise and {110} decreases in the same stepwise fashion. $E_v$ seen in the top graph of Figure 6.2 shows the same step-like character as observed in the relative percentages of the different orientations from the same graph, with discrete changes in energy values. The vacancy formation energy also increases as the percentage of {111} orientated surface becomes exposed, as the closely-packed {111} surface shown in Figure 6.3 (d) does not bind as strongly to adatoms.

It is necessary to plot out each individual particle and to characterize each surface in turn to determine the surface reactivity. It is also a more precise indication of the available activated sites than using Wulff constructs [4], geometric approximations of surface reactivity based on particle shape, and can be useful in predicting useful shapes with active sites for further growth of more advanced nano shapes, such as Figure 6.3 (b) and (c), rhombicuboctahedrons that could be used to construct hollow nanostructures through the addition and etching process described by Rang Long et al. [5]. The lowest vacancy formation energy corresponds to an optimal combination of {100} surface, {110} edges and {111} corners.

For smaller particles, the surfaces were also investigated. In $3 \times 3 \times 3$, although the perfect cube $3_0$ has sides packed in {100} orientation, their binding points are in the range of 2.8 eV, resembling bulk {111} surface binding energies. Figure 6.4 shows the changes of binding site energies of different surface orientation against particle size. Besides the

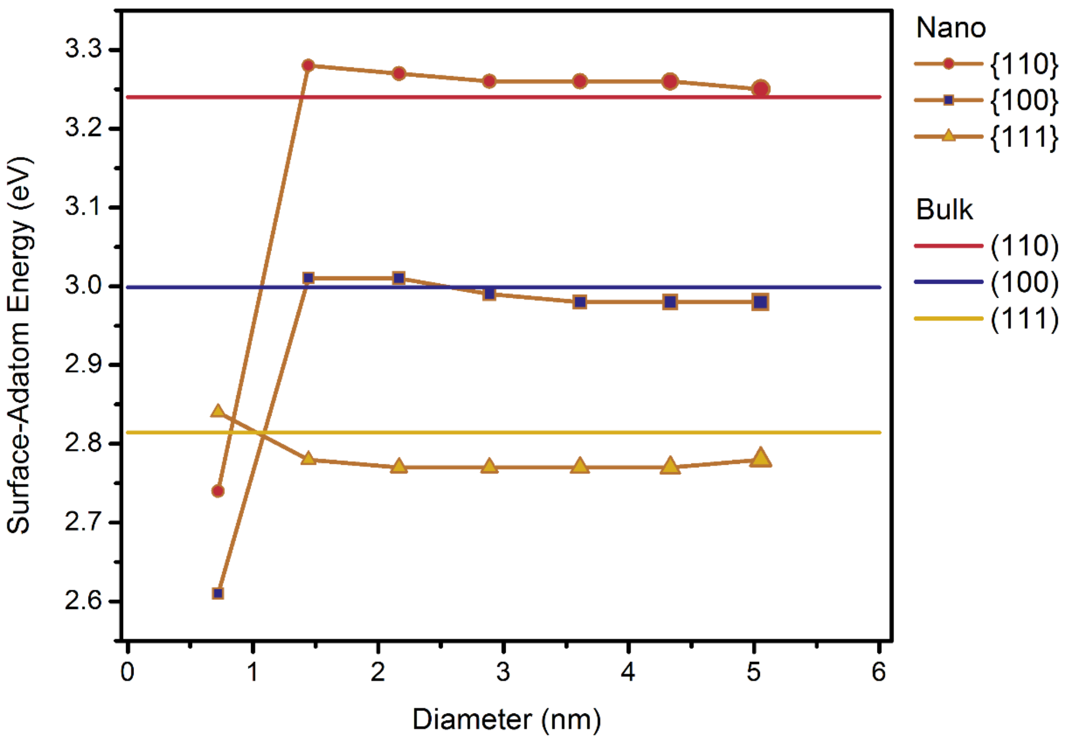


Figure 6.4. The average surface-adatom energy of different surface binding sites on nanoparticles of different sizes.

smallest particle, 3×3×3, which has a diameter of less than 1 nm, most particles have surface energies resembling bulk surfaces, with the largest nanocube having surface binding energies at the {111}-, {100}- and {110}-surfaces of 2.78 eV, 2.98 eV and 3.25 eV respectively. The main influence on the energy of a binding site was whether it bordered on another surface, as can be seen in Figure 6.3 (b) and (c) in the yellow points bordering the edges between the {100} and {110} surfaces.

## 6.5 The Vacancy Formation Energy

In this section the shape and size dependence on the vacancy formation energy $E_v$ was investigated.

### 6.5.1 Shape Dependence

Vacancy formation energy was calculated using equation 2.14, combining the volume extraction energy and the surface-adatom energy. Since the bulk value remains relatively constant for a particle of fixed size, variation in $E_v$ is thus largely a reflection of surface binding energies which is surface orientation dependent, and the results are presented in

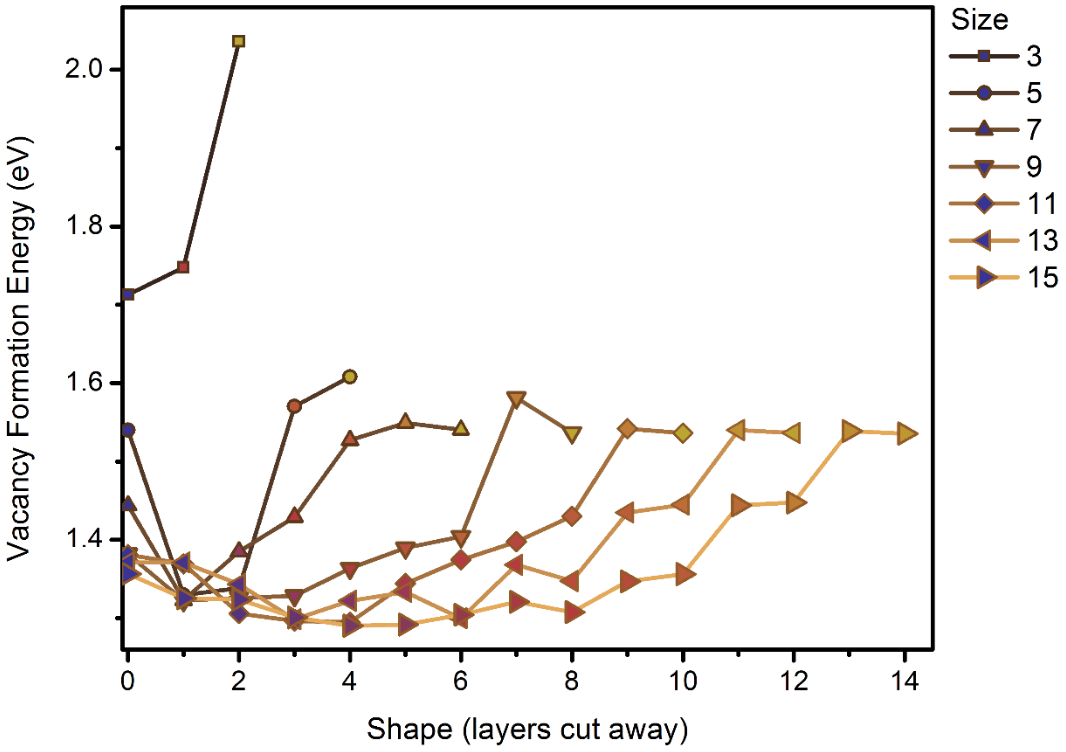
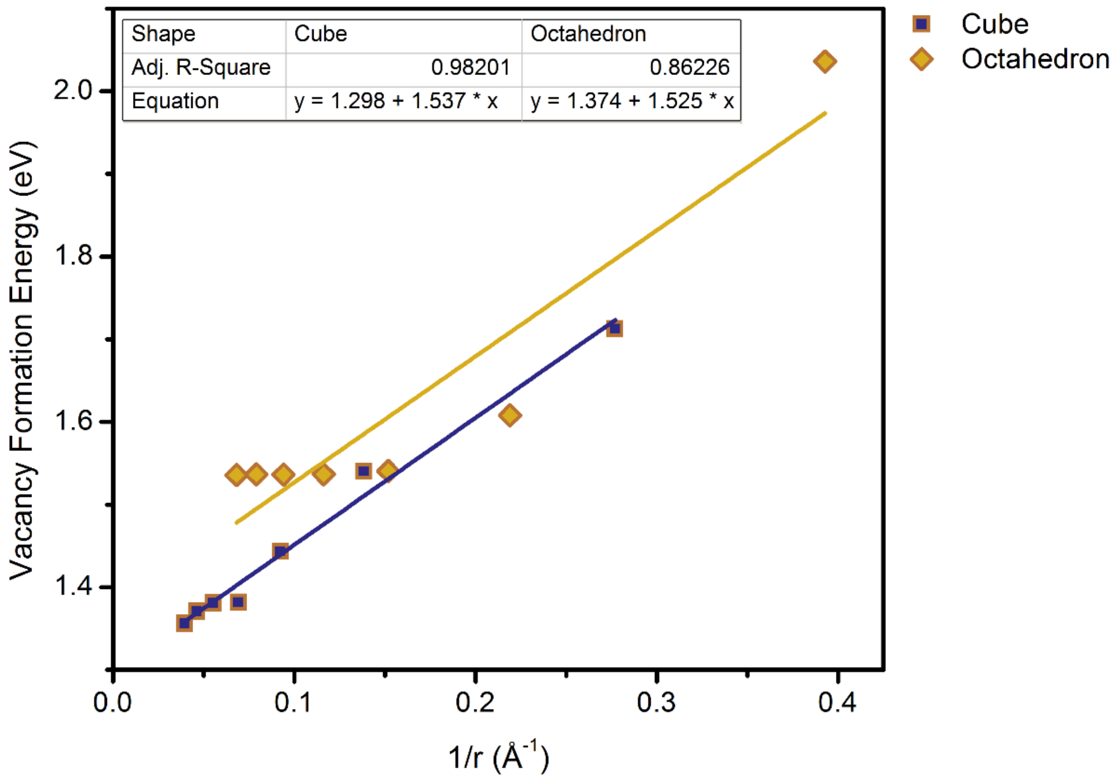


Figure 6.5. The vacancy formation energy $E_v$ presented for the range of shapes of nanoparticles of different sizes. The x-axis shows the variable *i* in the expression which represents the nanoparticle shape produced from shaving *i* layers of atoms from the edges and corners of a perfect $x \times x \times x$ -packed cube, and the legend represents the *x*.

Figure 6.5. The vacancy formation energy for the smallest particle is the highest. As edges and corners are cut away to change the shape and the particles consequently become smaller, average $E_v$ swiftly approached the average cohesion energy per atom. All the larger particles have comparatively similar vacancy formation energies, but display a stepped character during the progression of discrete shape modification. The minimum vacancy formation energy indicates a low activation energy threshold for creating vacancies in the particle, which is taken as an average over the entire surface of the particle and varies according to surface orientation, similar to the results in Figure 6.2.

The vacancy formation energy also depends on the extraction energy in the volume of the particle, which is influenced by the ratio of surface-to-volume atoms. This dependency on size is much weaker, however, than surface energy dependency on orientation. The surface should be characterized individually on a case by case basis for more specific properties. $E_v$ is strongly shape dependent, and the expected size-dependence is not obvious from Figure 6.5.



| Shape | Cube | Octahedron |
|---|---|---|
| Adj. R-Square | 0.98201 | 0.86226 |
| Equation | y = 1.298 + 1.537 * x | y = 1.374 + 1.525 * x |

Figure 6.6. The vacancy formation energy plotted against the inverse size of perfect nanocubes and octahedra.

## 6.5.2 Size Dependence

To eliminate shape-dependency and look at the influence of particle size on $E_v$, the energies of perfect cubes and octahedrons were plotted in Figure 6.6 against size as the inverse of the radius of each nanoparticle.

N.T. Gladkikh et al. calculated the vacancy formation energy in spherical particles and found a linear relation of a decrease in $E_v$ with the increase in particle size [6]. The expected decrease in vacancy formation energy with increase in size was not observed as predicted by the abovementioned study, and a linear trend line did not produce a satisfactory fit for octahedral shaped particles, though the inverse of the expected relationship is observed in cubic particles. This further suggests that surface effects and particle shape have a far greater effect on determining vacancy formation energy, something which was not taken into account in the study by N.T. Gladkikh et al., where spherical particles were used.

## 6.6 The Cohesive Energy

Next the average energy per atom in each particle was investigated. This is an indication of the overall stability of each nanoparticle. The higher the cohesion energy, the better the atoms "cling together" and the more stable the shape, as it takes more energy to break the particle apart.

### 6.6.1 Shape Dependence

Perfect cubes with six {100} faces, ranging from diameters of 0.7 nm to 5 nm, were simulated at 0 K to obtain the average cohesion energy per atom in each particle. The shapes were created by first simulating a perfect cube with six {100} facets, which is the first shape. The largest particle of 5 nm was packed $15\times15\times15$ atoms edge to edge, disregarding face-centred atoms in the count. Progressively shaving off one {110} layer of each edge and one {111} layer of the corners produced an imperfect cube with a reactive {110} surface on the edges but a less reactive {111} on the corners, and is called a rhombicuboctahedron. Once all the edges and corners have been shaved away, an octahedron shape resembling two pyramids attached top-to-bottom remains, which has 8 triangular {111} facets.

Figure 6.7 shows the particles of a $15\times15\times15$ perfect cube with 0 layers cut away, the particle with 3 layers cut away, denoted by $15_{-3}$, 8 layers cut away; $15_{-8}$, and 14 layers cut away to form an octahedron; $15_{-14}$ . Each shape has a different cohesion binding energy also plotted in Figure 6.7.

As mentioned before, the cohesion energy measures the amount of energy taken to break the particle apart and place the atoms infinitely far apart where their interactions with each other are negligible; thus, the greater this energy, the more stable the particle. From the maximum binding energy of the curve, the most stable particle can be identified.

As would be expected, the perfect cube with its sharp edges and corners is less stable than a rhombicuboctahedron, which more closely resembles a sphere. The octahedron is even less stable than the perfect cube. For the largest particle of 5 nm, packed $15\times15\times15$ atoms, cutting away 3 layers on the edges and corners, denoted $15_{-3}$, is the most stable shape. This process was repeated for each particle size ranging in odd numbers from $3\times3\times3$ atoms through $15\times15\times15$ atoms.

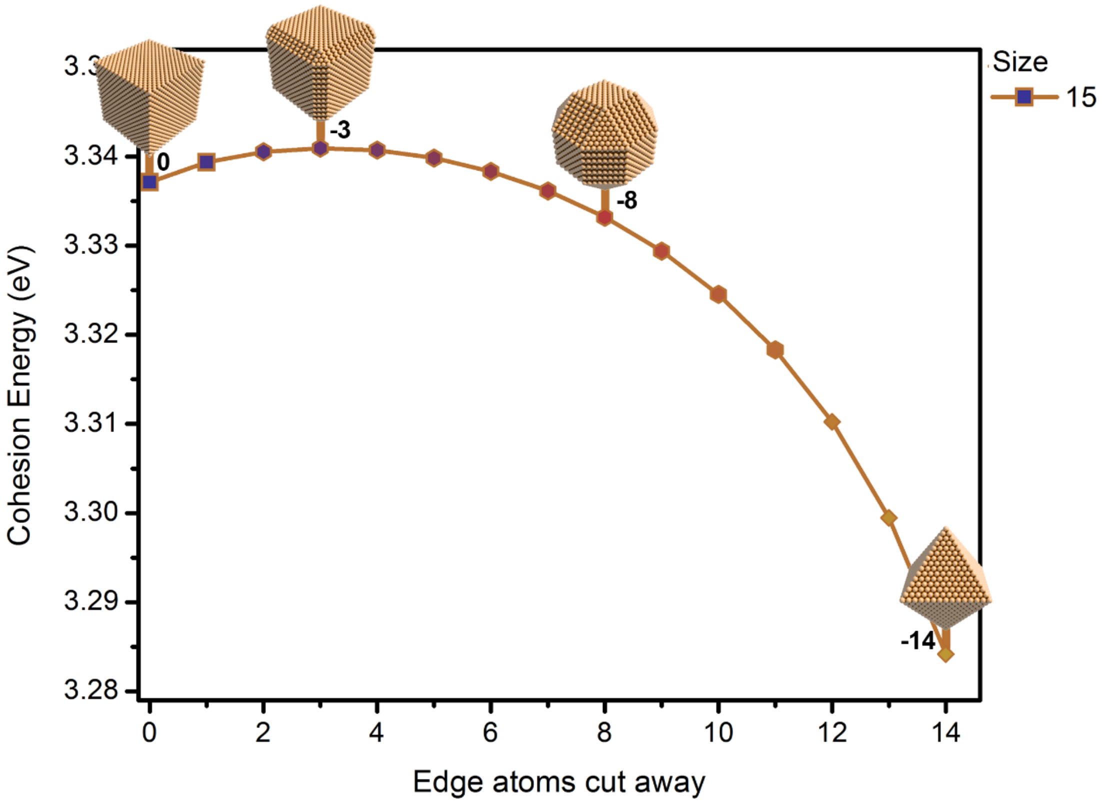


Figure 6.7. Showing how the most stable shape was determined by progressively cutting away edges and corners to change the particle shape and comparing the corresponding average binding energies. The x-axis shows the variable *i* in the expression which represents the nanoparticle shape produced from shaving *i* layers of atoms from the edges and corners of a perfect $x \times x \times x$ -packed cube.

Shown in Figure 6.8 is the cohesion energy for each particle size as the corners were systematically cut away to change particle shape. A comparison of all the calculated cohesion energies further show a progression in stability as the particle sizes increase. The smallest particle $3 \times 3 \times 3$ are by far the most unstable, and the cohesion energy increases as the size is scaled up, drastically at first, and then with smaller and smaller steps. From $9 \times 9 \times 9$ the particles take on a roughly similar cohesion energy which suggests that they are comparatively stable. For each size, the octahedron shape represented by $x_{-(x-1)}$ was the least stable shape with the lowest cohesion energy for any particle of that size.

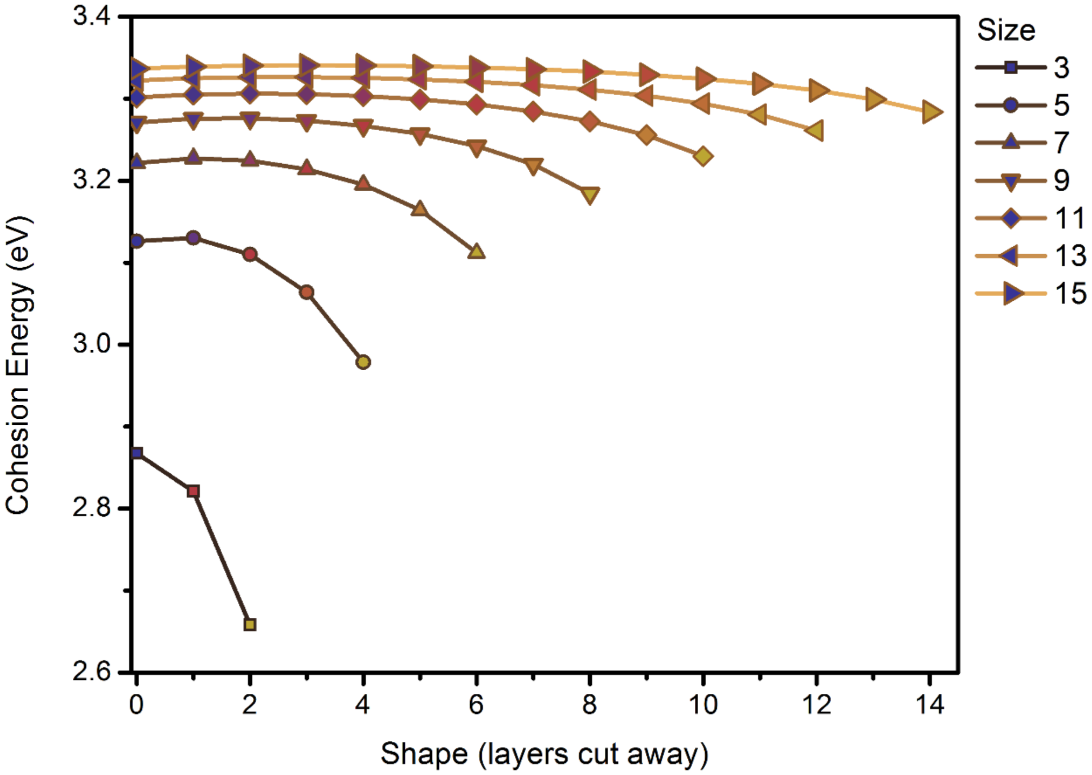


Figure 6.8. The x-axis again shows the variable *i* in the expression $x_{-i}$ which represents the nanoparticle shape produced from shaving *i* layers of atoms from the edges and corners of a perfect $x \times x \times x$-packed cube, where the legend represents the *x*. Cohesive energy for each particle shape and size is plotted, showing the most stable particle sizes are larger, and the more stable shapes are rhombicuboctahedrons.

For the smallest nanocube the cohesion energy is the highest at $3_0$ where no edges are cut away. This suggests that the particle is more stable when it is slightly larger. For $5\times5\times5$ the highest cohesion energy is not the perfect cube but the particle that has had one layer of atoms cut away from the edges, $5_{-1}$, exposing {110} on the edges and {111} on the corners. However, as more edge atoms are cut the particle becomes less stable again. Similarly, $7_{-1}$ is more stable than $7_0$. For the $9\times9\times9$particles, optimum stability shown by the maximum cohesion energy begins to shift to $9_{-2}$ as the particles become larger. This suggests that the particles are less stable as perfect nanocubes with sharp edges and corners and become more stable when one or two layers are cut away. This makes sense in terms of the bonds available to atoms sitting on edges and corners; they are in contact with fewer neighbouring atoms and are therefore not anchored as firmly. Once these unstable atoms are stripped away the underlying atoms that have more bonds to neighbouring atoms are more stable and increase the average cohesion energy per atom.

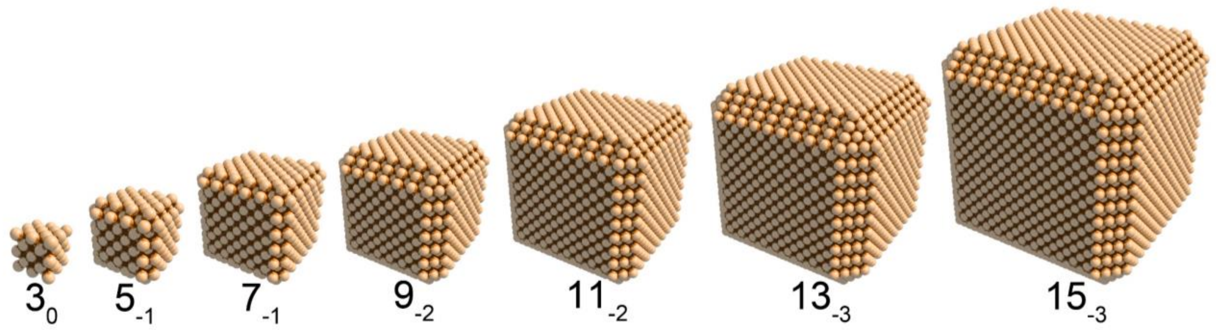
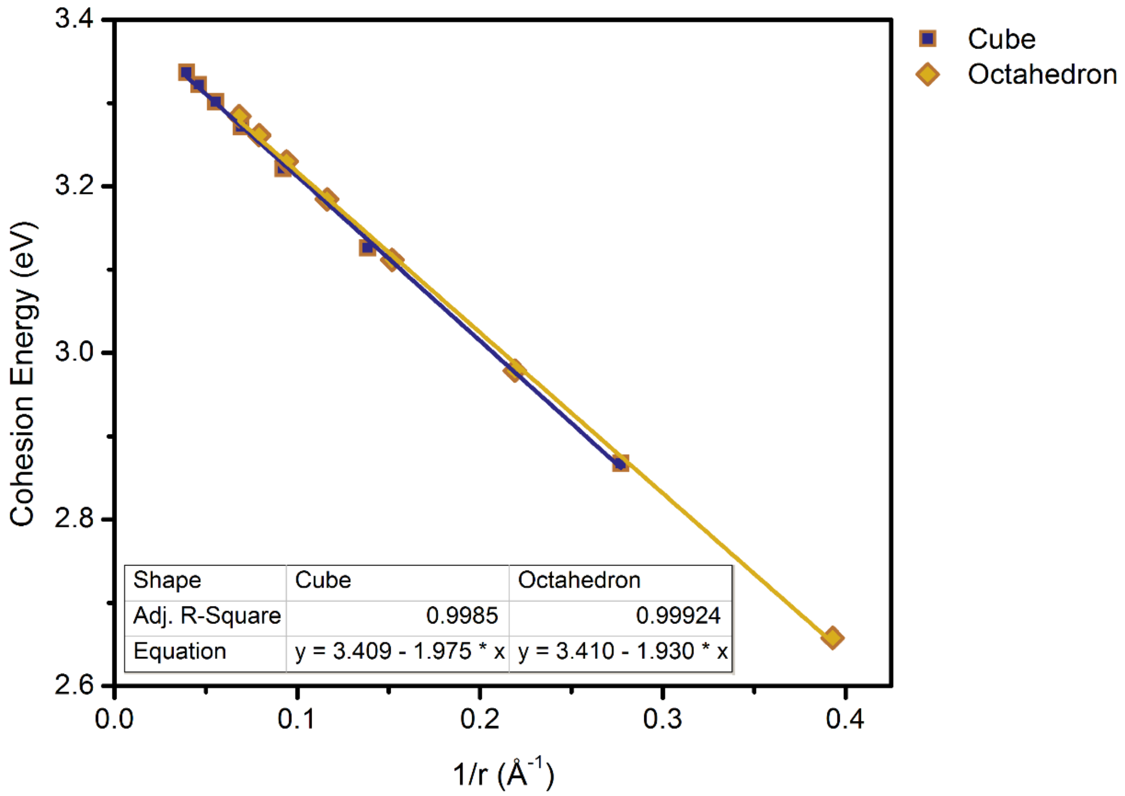


Figure 6.9. For each size of particle, the most stable shape is shown as determined by average cohesion energy of the particle.



| Shape | Cube | Octahedron |
|---|---|---|
| Adj. R-Square | 0.9985 | 0.99924 |
| Equation | y = 3.409 - 1.975 * x | y = 3.410 - 1.930 * x |

Figure 6.10. The cohesion energy plotted against the inverse size of perfect nanocubes and octahedra shows an inverse correlation to size in nanoparticles.

The series of cubes, or imperfect cubes, which represent the most stable form for each nanoparticle of a particular size, are presented in Figure 6.9. Using these more practically-shaped and sized particles, future studies can be done which may investigate other factors, such as temperature variation.

## 6.6.2 Size Dependence

Figure 6.10 shows the cohesion energy of cubes and octahedra of different sizes plotted against the inverse of their size.

When the shape factor is eliminated and cohesion energy $E_{coh}$ is plotted against the inverse of particle radius in Figure 6.10, the previously expected decrease in energy with decrease in size expected in Figure 6.6 is finally seen. Cohesive energy is not as strongly influenced by surface orientation and shape, and thus produces a linear character corresponding to the results of N.T. Gladkikh et al. [6].

## 6.7 Particle Stability and Reactivity

Another useful way to visualize the information of $E_v$ and $E_{coh}$ is to plot the two variables against each other to determine each particle's respective stability and surface reactivity, as shown in Figure 6.11.

In Figure 6.11, the smallest particles are clearly shown to be the least stable with the most reactive surfaces. As the particles become larger their energy values become more similar; they become more stable, with stronger cohesion binding energies between atoms, but the surface shows a small range of reactivity depending on its shape. Octahedron shapes are much less reactive, given in yellow, with larger vacancy formation energies. Rhombicuboctahedrons, given in pink, are the most reactive.

Vacancy formation energy indicates surface reactivity, as the larger $E_v$ value corresponds to a larger area of densely packed (111) surface where atoms bind more weakly to the surface. A low $E_v$ corresponds to more (110) and (100) surfaces where atoms can bond more strongly to the surface. (110) only occurs in rhombicuboctahedrons, where there are (100) and (111) surfaces, which limit overall surface reactivity.

Cohesion energy indicate the energy per atom, and a lower $E_{coh}$ leads to instability, such as in octahedral with many sharp corners where atoms cannot bond strongly to many nearest neighbours. This can lead to rearrangement and phase changes of the particle, as the bonds of these unstable corners of the particle are easily broken. The same is true of perfect cubes with sharp edges and corners. The spherical shape is the most stable as it allows most surface atoms to maximize their number of nearest neighbours, so particles with a roughly spherical shape and few corner atoms with unstable bonds will have larger cohesion energies and better stability.

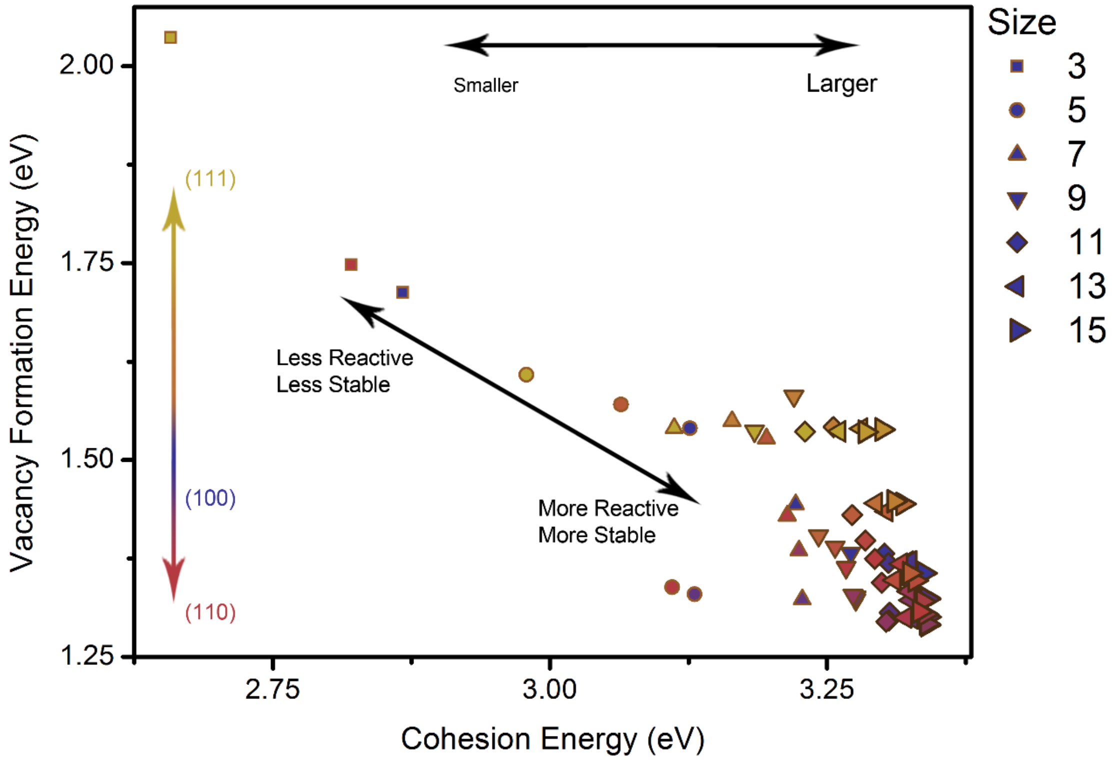


Figure 6.11. This is a plot of particle stability on the x-axis and surface reactivity on the y-axis. This figure presents all the particle sizes.

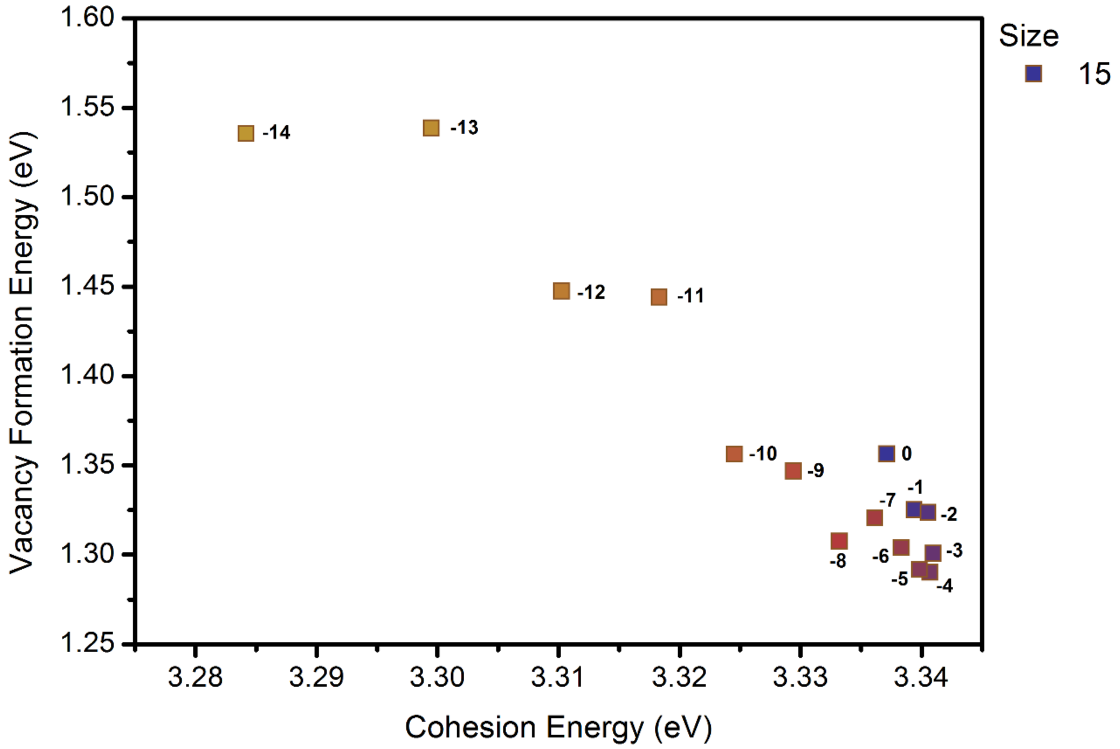


Figure 6.12 This figure examines the largest particles of 15 layers wide, in more detail. The labels indicate the shape of the particle, indicating how many layers have been cut off the edges and corners.

Figure 6.12 shows the range of shapes of the $15\times15\times15$, 5 nm diameter particles, where the label on the symbol represents $15_{-i}$ . Thus “0” represents the $15_0$ perfect cube, “-1” represents $15_{-1}$ , and so on. From the graph the most stable, more reactive, particles are $15_{-4}$ , $15_{-5}$ , and $15_{-3}$, in that order. The less reactive stable particles are $15_{-2}$ and $15_{-1}$. $15_{-14}$ is both unreactive and unstable. Depending on the level of surface reactivity required of a particle, this type of characterization of particles may be useful.

## 6.8 Summary

Nanoparticles of FCC Cu ranging in size from 0.7 nm to 5 nm, with shapes ranging from perfect cubes, to rhombicuboctahedrons, to perfect octahedrons, were simulated and characterized. A study was made of the interactions between the size and shape, and the vacancy formation and cohesion energies of the particles. The model could effectively simulate large particles of over 12 000 atoms in relatively short amounts of time of a few days per run, and could produce models of the surfaces, the corresponding binding energies of those surfaces and the internal cohesion binding energies of each particle. It was found that particles less than 1 nm in diameter were too unstable to be practical for analysis. The larger the particle was, the more stable it was internally. Even so, the different shapes had a wide range of surface reactivities, where a large number of {110} binding points correspond to a more reactive particle, while the octahedron shapes were the least reactive and least stable. While $E_v$ varied extensively based on shape but not particularly strongly influenced by particle size, $E_{coh}$ was dependent both on nanoparticle size and shape. Using $E_{coh}$ as a guide, particles were chosen which had the most stable shape. In the next chapter, the behaviour of these stable particles are studied over temperature; the melting temperatures, surface energies, extraction energies, and vacancy formation energies are investigated.

## 6.9 References

[1] Y. H. Wang, P. L. Chen and M. H. Liu, *Nanotechnology* **17**, (2006): 6000.

[2] A. P. Sutton and J. Chen, *Philosophical Magazine Letters* **61** (1990): 139.

[3] C. van der Walt, J. J. Terblans and H. C. Swart, *Computation Materials Science* **83** (2014): 70.

[4] R. A. Molecke, “Characterization, modeling, and simulation of Multiscale directed-assembly systems” (PhD diss., The University of New Mexico, 2011).

[5] R. Long, D. Wu, Y. Li, Y. Bai, C. Wang, L. Song and Y. Xiong, *Nano Research* **8** (2015): 2115.

[6] N. T. Gladkikh and O. P. Kryshtal, *Functional Materials* **6** (1999): 823.

# Chapter 7 - Thermodynamics in Cu Nanocubes

## 7.1 Introduction

Cu nanocubes have a wide range of uses and been created experiemntally, and there now exists ever more efficient ways to control size, shape, and the surface orientation of the exposed surface [1]. In the previous chapter, Cu nanocubes of different sizes were simulated. For each size the rhombicuboctahedron shape that minimized the internal cohesion energy of the particle was chosen. In the previous chapter, characterization ignored temperature dependence, keeping particles at 0 K.

Diffusion limited segregation in nanoparticles is of interest for use in catalysts because it determines the element available for reaction at active binding sites on the surface, and the study of this segregation means that diffusion kinetics under various conditions need to be understood. Having previously studied the diffusion kinetics of small nanoparticles under strictly regulated conditions, this chapter aims to investigate the behaviour of nanoparticles under the influence of a range of temperatures.

In this chapter, each of the chosen particle’s thermodynamic properties were investigated by calculating the average potential energy per atom for each particle over temperature. The vacancy formation energy of the particles as well as the internal binding energy and surface binding energies of each particle were also calculated. The nanoparticle melting temperatures, surface energies and vacancy formation energy were analysed. All energies given in electron Volt (eV) are given per atom, even if not stated to be per atom.

## 7.2 Methods

Using the Sutton Chen molecular dynamics algorithm, several nanocubes were simulated ranging from size 0.7 nm to 5 nm. The nanocubes pictured in Figure 6.9 were chosen for the greatest stability by comparing the average cohesion binding energy of different shapes of nanocubes, octahedrons and rhombicuboctahedrons [2].

The melting behaviour of each particle was studied by monitoring the potential energy (PE) [3, 4] of each crystal as atoms were relaxed for 20 000 time steps of 1 picosecond. As the total potential energy is $U_{perfect}^{total}$, average PE/atom can be calculated similarly to the cohesion energy with equation 3.22. A smaller time step allowed for the minimizing of both rounding errors and truncation errors caused by using a truncated Taylor expansion of the time integration algorithm, to more accurately model FCC atomic interactions [5]. The process was repeated at various temperatures ranging from 0 K to 850 K in discrete intervals of 50 K. The nanoparticles were heated up to 850 K, as the previous molecular dynamics studies of Cu bulk crystals in Chapter 5 showed surface premelting and the formation of an adlayer around 900 K [6-8]. The Berendsen thermostat was used for temperature control. Whereas in Chapter 5 the periodic boundaries prevented significant displacement of the crystal, and in Chapter 6 the 0 K temperature severely restrained displacement, in this chapter atoms were allowed to relax, and moved at ever higher velocities. As such, the Dissipative Particle Dynamics (DPD) algorithm described in section 3.6.4 was used to prevent the so-called "flying ice-cube" effect.

This study aims to investigate the solid state behaviour of the nanoparticles and thus the range of temperatures where melting occurs, falls outside the scope of this study. Furthermore, the interest of this study is in studying FCC crystals only, as the Sutton-Chen deals with modelling FCC metal properties.

## 7.3 Thermodynamics and Potential Energy

Literature study indicates that melting in nanoparticles occurs not at one temperature but over a range of temperatures, where the surface melts first and then deeper layers start to melt and disorder at successively higher temperatures [6]. One way to discern this process is by observing the potential energy per atom (PE/atom) over temperature [3, 4]. This can be observed, both over time as the crystal is relaxed [9], and as a running average energy at that particular temperature [3]. Observing the total PE/atom over time for a particular temperature, melting will appear as an increase in the total PE/atom.

Using averages of crystal energies over the relaxation run, where the crystal temperature most closely resembled the desired temperature, an average vacancy formation energy for a Schottky vacancy with the surface as a defect was calculated. This allowed for the

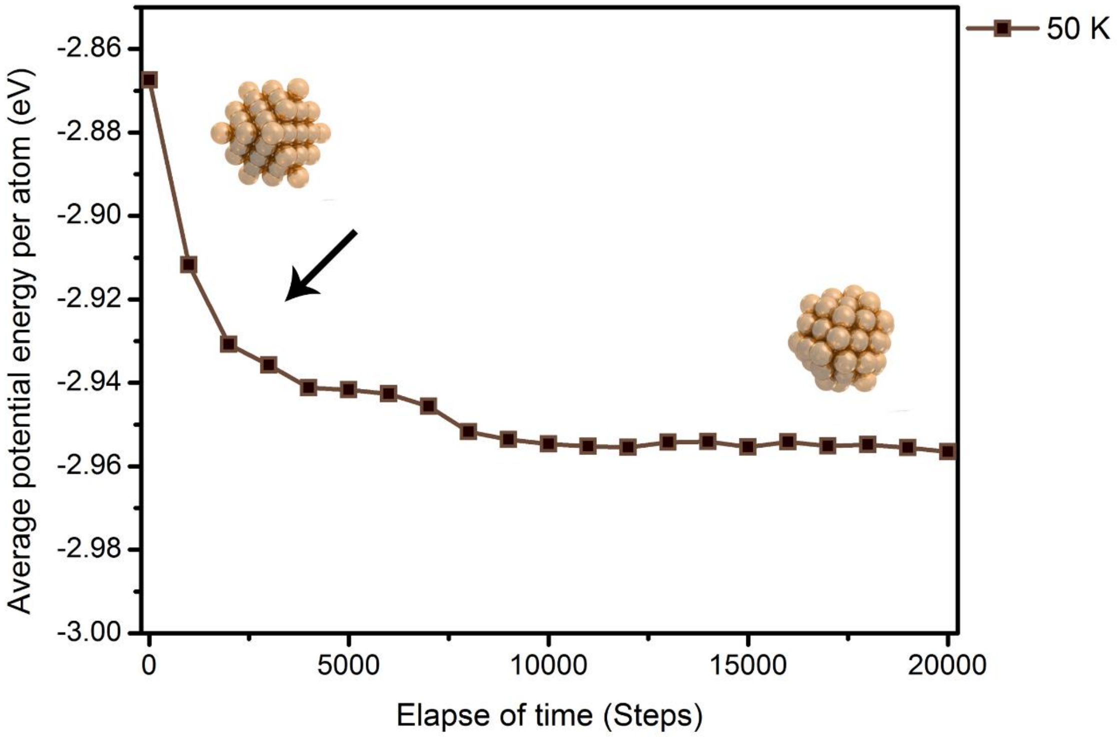


Figure 7.1. The previously $3 \times 3 \times 3$ packed FCC structure $3_0$ shows a marked decrease in average PE/atom at 50 K, and the structure itself had changed to a HCP structure.

characterization of vacancy formation energy in Cu nanocrystals over a variety of temperatures and sizes.

### 7.3.1 Average PE Over Time

Nanocubes with stables shapes having {100} faces, {110} edges and {111} corners, ranging from diameters of 0.7 nm to 5 nm, were simulated. Consequently, the behaviour of the chosen stable shapes was investigated over different temperatures. Each particle was simulated for 20 000 steps at a stable temperature that ranged from 0 K to 850 K in steps of 50 K.

For the $3_0$ particle at temperatures as low as 50 K the atoms rearranged themselves into a hexagonal close packed, or HCP, lattice. For the slightly larger $5_{-1}$ particle, the same was observed at 450 K. A marked change in the average PE/atom can also be observed; the energy decreases markedly, indicating a phase change. This can be seen for the average PE/atom of the $3_0$ particle in Figure 7.1, and for the $5_{-1}$ particle in Figure 7.2.

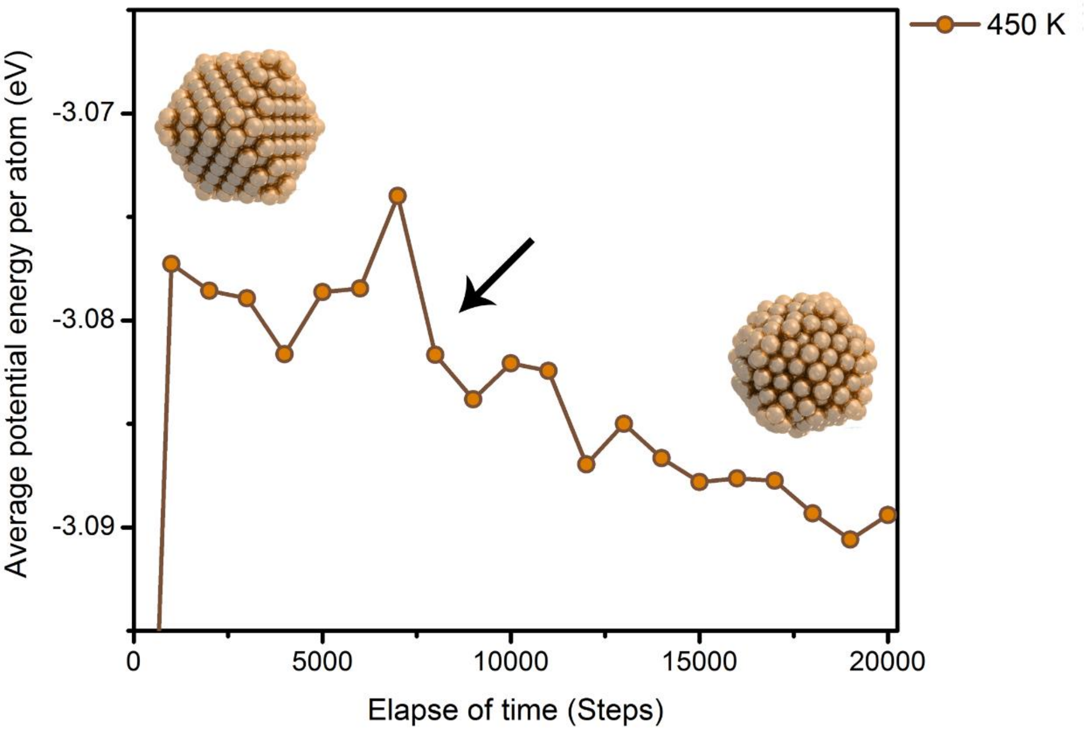


Figure 7.2. The PE/atom of $5_{-1}$ also decreased, though less drastically than $3_0$, and the particle rearranged itself into an HCP structure.

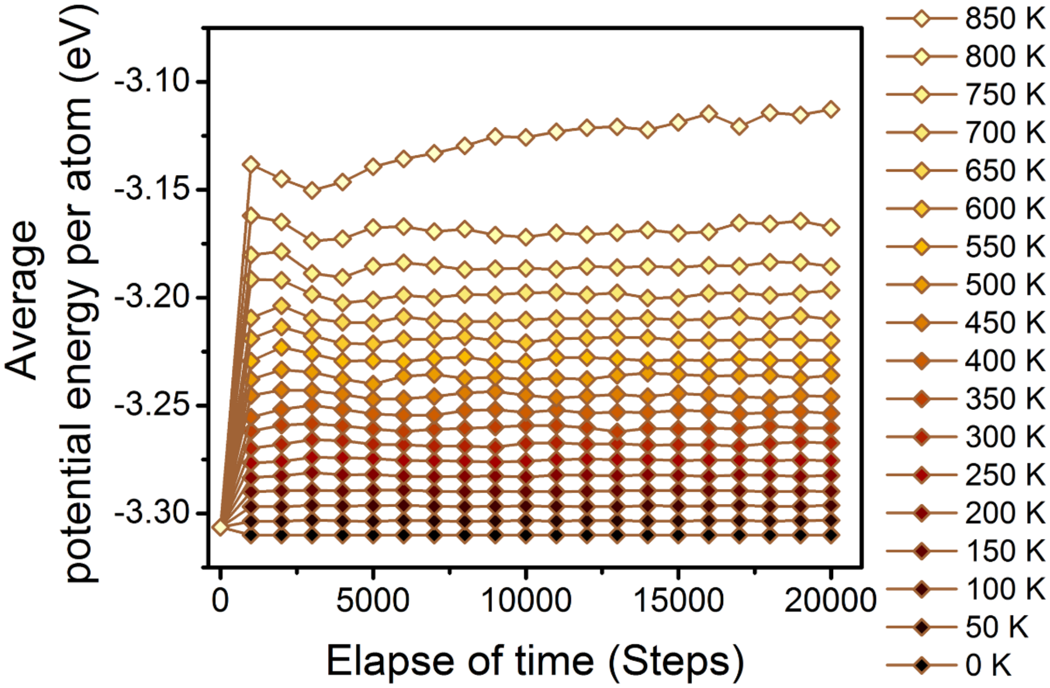


Figure 7.3. The average PE/atom of the $11_{-2}$ crystal. The particle was simulated for 20000 steps at each temperature. At 850 K the average PE increases, and the positive slope is an indication of melting.

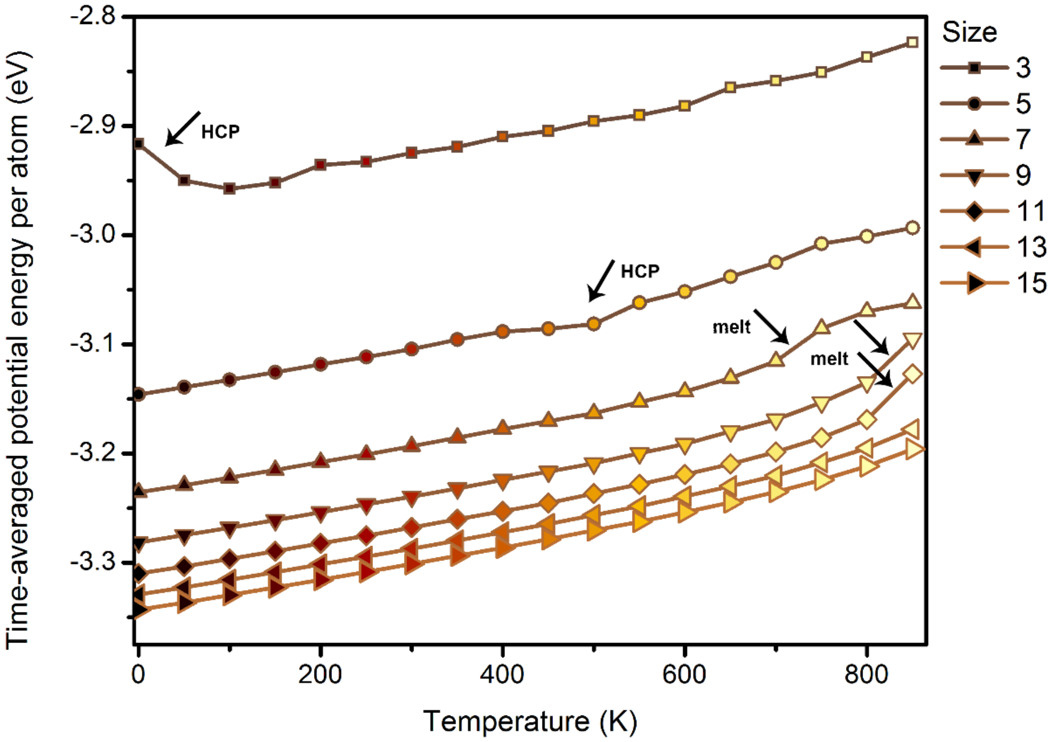


Figure 7.4. Time-averaged PE/atom plotted against temperature shows at which temperature each particle melts. The two smallest particles show a dip which indicates an FCC-HCP transition, whereas $7_{-1}$, $9_{-2}$, and $11_{-2}$ particles show a rise pointing to melting which occurs before the cut-off temperature of 850 K.

Larger shapes retained their FCC packing until they began showing slight premelting, before finally melting completely. As the particle melts, the average PE increases notably as much of the system's energy is converted into kinetic energy. Figure 7.3 shows the $11_{-2}$ particle which melts at the slightly lower temperature of 850 K.

The curve over simulated time steps for the 850 K simulation behaves differently from the other runs; the energy has a positive slope as the average PE/atom increases. The smaller the particle, the lower the melting temperature. Premelting is also observed, most notably in the corners. This suggests that the slightly cubic shape chosen at 0 K is not as stable at higher temperatures, and a more spherical shape may be more resistant to premelting at higher temperatures. It may be pertinent to test the relative stability of each different shape at different temperatures to determine the most desirable shape for each temperature.

## 7.3.2 Time Averaged PE Per Temperature

Having investigated the PE changes at various temperatures, the data is summarized in Figure 7.4. Each run shown in Figure 7.3 was averaged to create a single data point.

The initial phase of each run from Figure 7.3, where temperature changes as the crystal settles from the unrelaxed initial state, was omitted in order to use data points corresponding to the desired simulated temperature. These points were plotted against their corresponding temperature, to produce a new curve. When viewing the average PE/atom for the different temperatures, melting can be seen as a sharper upward slope in the line, a sudden increase, an interruption in the steady progression in energy. For the $11_{-2}$ curve, the sudden increase in slope between 800 K and 850 K is apparent, indicating a phase change. Thus, by 850 K, the $11_{-2}$ rhombicuboctahedron undergoes melting. This "bump" is also observed in the other particle sizes, most notably $7_{-1}$ and $9_{-2}$, which show melting at 750 K and 850 K each. The FCC – HCP phase change gives rise to a decrease in slope. In $3_0$ this decrease is obvious, less so in $5_{-1}$ where it only produces a small plateau. On the other end of the scale, the largest particles melt around the same temperatures as the bulk. Having determined the melting behaviour of the simulated particles, the vacancy formation energy behaviour with temperature can be determined.

## 7.4 Vacancy Formation Energy Over Temperature

Using the range of temperatures where the particles are unmelted, the vacancy formation energy was determined for several time steps at each temperature: the same range of stable temperature used to draw Figure 7.4. The fluctuations in crystal energy created a large spread in the data, and error bars are included to show the range of values for each temperature run.

Figure 7.5 shows that the average monovacancy formation energy of each particle does not change significantly with temperature - the greatest decrease is 0.13 eV, which lies within the error margin on the high temperature values of 0.14 eV. However, a trend of a slight decrease with temperature may be observed. This dip most often occurs close to melting and may also indicate pre-melting as the crystal begins to expand [6], and mirrors the dip in PE/atom seen in Figure 7.4. The vacancy formation energy of smaller particles tended to oscillate more wildly at lower temperatures as they undergo phase changes at lower temperatures. The vacancy formation energies of both the $3_0$ nanocube at 0 K, and the $5_{-1}$ imperfect crystal at 400 K, showed a sudden increase in $E_v$, shortly before undergoing phase change and rearranging into HCP structures.

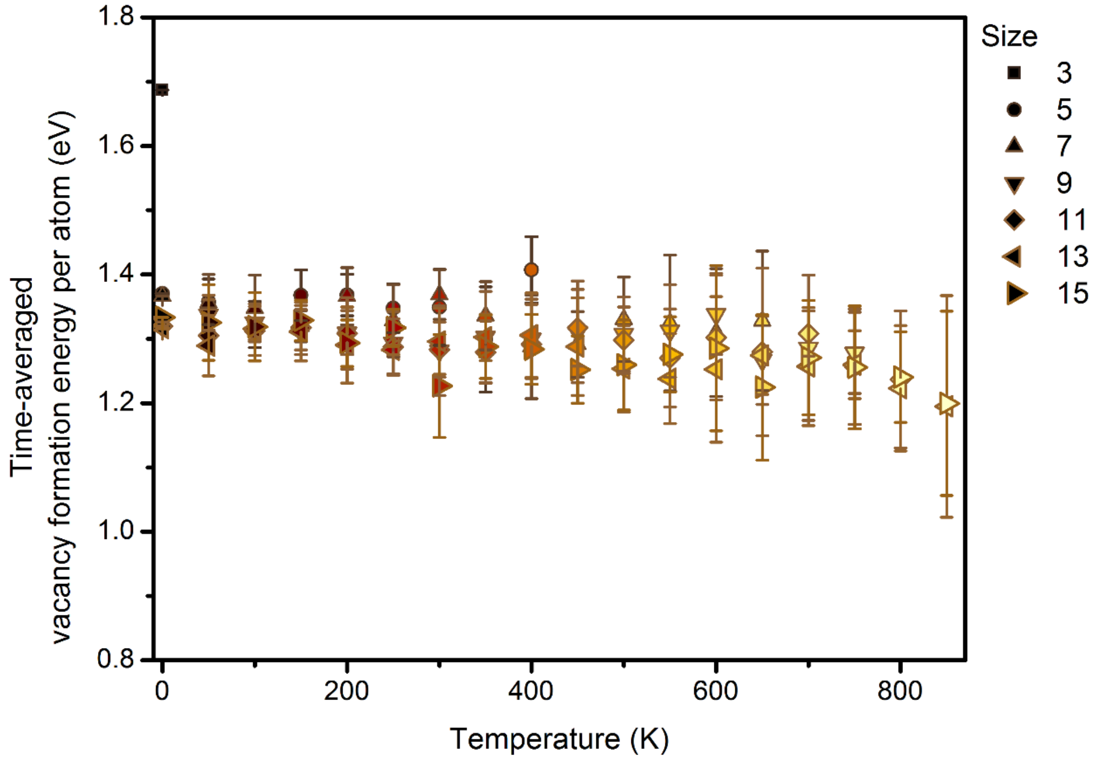


Figure 7.5. The vacancy formation energy of the nanoparticles with increased temperature showed a slight decrease, particularly approaching high temperatures.

### 7.4.1 Surface Binding Energy and Extraction Energy Over Temperature

The previous study of bulk Cu crystal with the same SC software may be helpful to aid in interpreting the data. The study showed the vacancy formation energy is strongly influenced by surface orientation, and only slightly by bulk binding energy inside the crystal. The surface binding energies of {111} and {110} surfaces changed with temperature, though the {100} surface showed little change in its binding energy. The bulk showed only a small, steady decrease in binding energy associated with the expansion of the crystal as it heats up.

As the nanoparticles have mainly {100} surfaces, this energy is not expected to change significantly with increase in temperature, though the energies on the edges and corners are expected to show a greater change, more closely resembling {100} binding energies, showing only slight change as seen in Figure 7.6 in the right-hand figure. However, as the particle approaches melting point, the surface becomes increasingly disordered, including the {100} surface, changing the surface energy more. The crystal is expected to expand with increasing temperature, corresponding to a slight decrease in bulk binding energies, which is reflected in the decreases in bulk binding energies in Figure 7.6 in the left-hand figure,

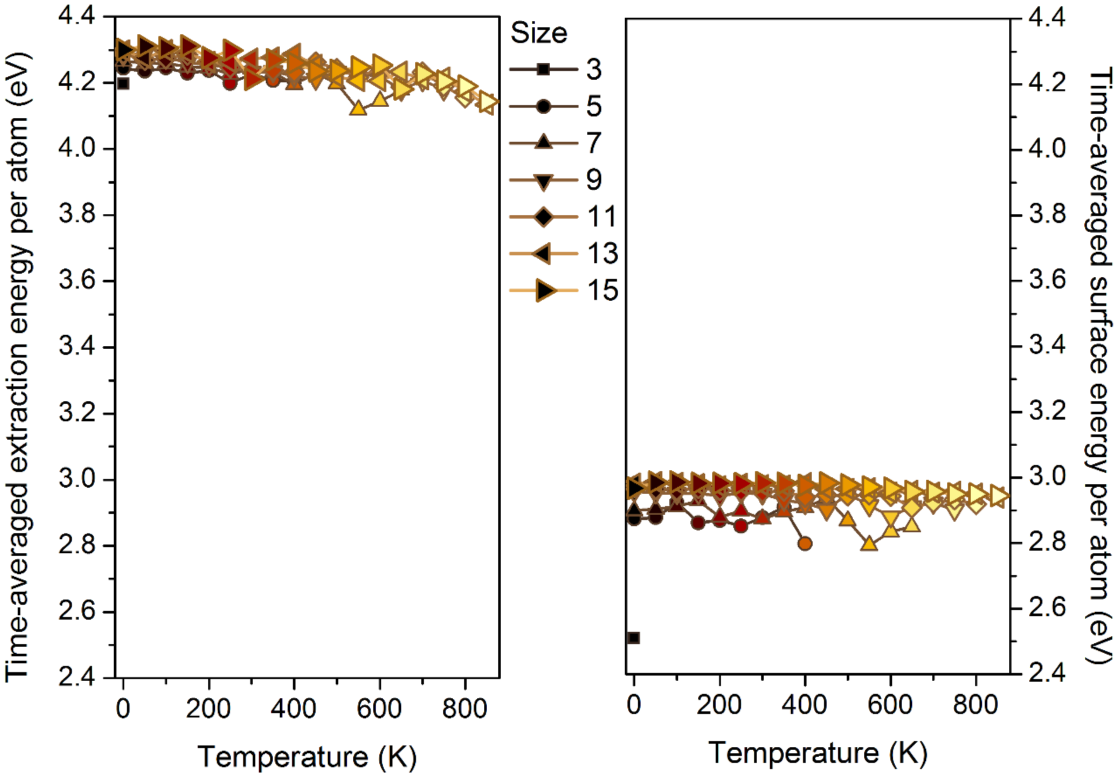


Figure 7.6. The left-hand figure shows bulk binding energies decreasing slightly with increasing temperature, reflecting the expansion of the nanocrystal. The right-hand figure shows the change in surface energies, which doesn't show a significant decreasing trend compared to bulk energies.

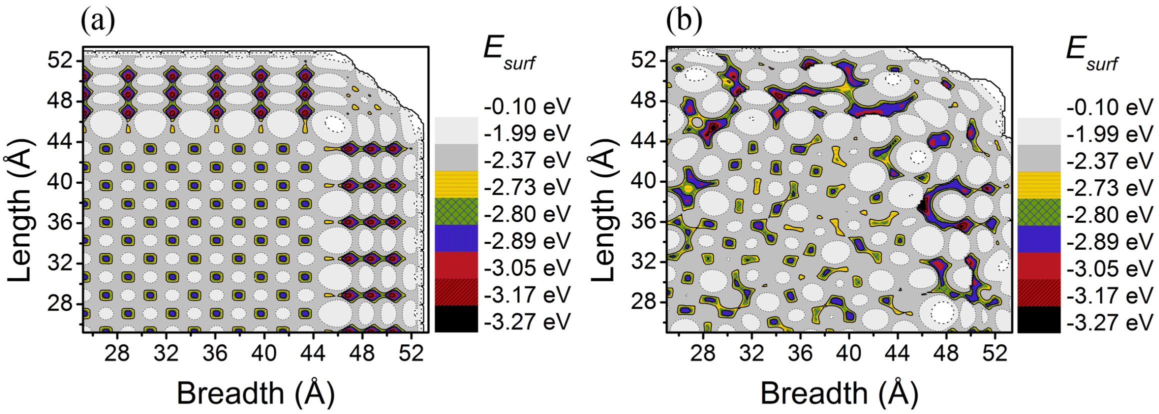


Figure 7.7 (a, b). The contour plot of a corner of the $15_{-3}$ particle at 0 K shows distinct surface orientations on the different facets in (a), and the size of the particle is small enough to be bounded by the graph. (b) The contour plot of the same particle at 800 K shows disorder on the surface though the different orientations are still identifiable. The size of the particle has swollen to become larger than the range of the graph.

particularly from 750 K. The net result is a slight negative slope in the vacancy formation energy over temperature, which becomes more pronounced than was observed in bulk, as it approaches its melting point, which is what is nominally observed in Figure 7.5.

A similar effect can be observed from the contour plots of the $15_{-3}$ surface energies shown in Figure 7.7 (a) and (b). Figure 7.7 (a) shows the nanocrystal at 0 K, with the different surface orientations clearly distinguishable with distinct binding energies. In Figure 7.7 (b) the $15_{-3}$ crystal is shown at 800 K. The change in the bulk energy is reflected in the visible expansion of the crystal size. As the crystal is centred and the scale is fixed, it is clear that crystal boundaries distinctly visible in Figure 7.7 (a) have disappeared in (b). The surface is in flux, but the overall binding energy averages out to that of a {100} surface. With higher temperatures, the orientations become less distinct as surface premelting occurs.

### 7.4.2 Surface Orientation Over Temperature

To investigate the effect of heating on the surface energies and compare it to the data in Figure 7.6, a sampling of the surface energies of different surface orientations in a small $7_{-1}$ particle was made. This particle melts at a relatively low temperature and thus surface behaviour can be observed from frozen, all the way to melting. The surface has {100} on the faces, a row of {110} binding sites on the edges, a single {111} point on the corners, and a few binding points on the ridges between the differently orientated surfaces that have uncharacteristically low binding energies. From the {100} points and {110}, sufficient sites could be sampled to produce a representative average. These could be compared to bulk values of the SC simulated Cu with periodic boundaries used to interpret Figure 7.6, and compared in Figure 7.8 (a).

The {100} energies of both the small nanoparticle and the bulk were not appreciably different, and this small difference is attributed to the small size of the particle and the surface-to-volume ratio of the nanoparticle. The interesting comparison is found in the {110} points, which have an appreciably lower value at 0 K than is observed for bulk. As soon as the particle is heated even slightly, the energies increase. This is reflected in Figure 7.8 (b), where the bottom figure shows more red dots as the {110} binding sites become stronger. A larger sample size is needed to interpret the behaviour of the {110} energies more fully. However, it does appear to approach {100} energies at temperatures close to melting. The average surface energy shown at 750 K also averages out to resemble that of {100}, as expected from the bulk study.

The nano data points in Figure 7.8 (a) represent only a small sample size, but show interesting interactions between neighbouring surfaces which bear more study. The shape,

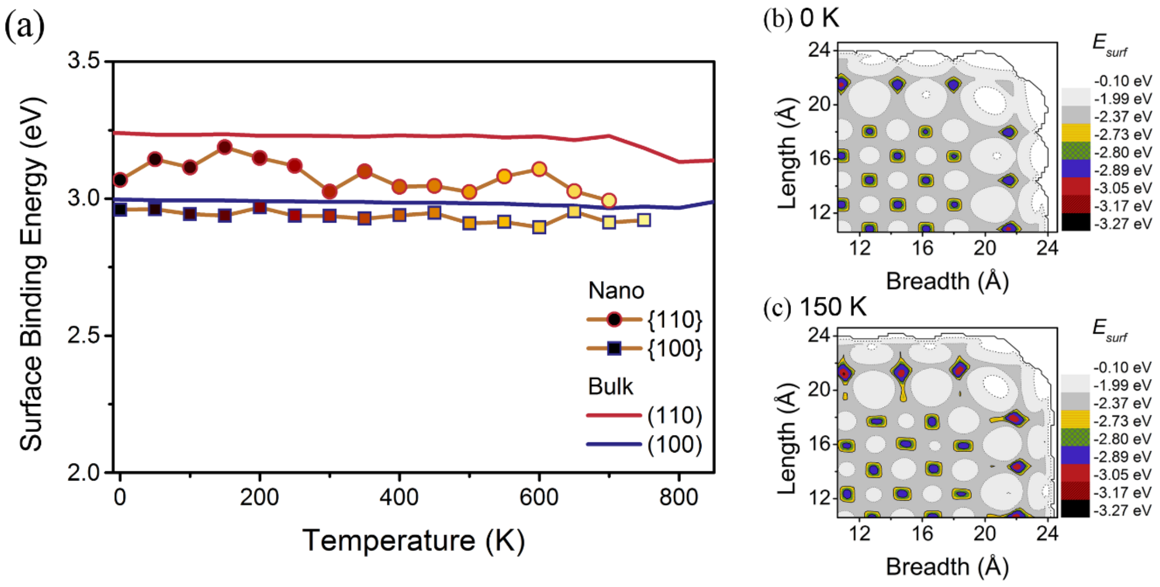


Figure 7.8 (a-c). (a) A sampling of the surface binding energies of a small $7_{-1}$ particle is compared to surface energies from bulk materials. The {100} are similar, but the {110} energies range widely and differ substantially from the bulk values. (b) The contour plots of the $7_{-1}$ particle at 0 K are easily compared visually with the same particle at 150 K in (c), where red spots indicate a stronger binding energy for 150 K.

particular temperature, and its interaction with the different surfaces could have interesting implications for directed diffusion and segregation, particularly in bimetallic segregation and the manufacture of complex nanoparticles [10].

## 7.5 Summary

This study allowed the investigation of the molecular dynamics present at different temperatures of entire nanoparticles of several thousand atoms each. The smallest nanoparticles were too unstable in cube form, indicating that nanoparticles have a minimum size requirement of 1 nm diameter. As particle sizes increased, the most stable shapes also changed. At higher temperatures, the surfaces changed and particles assumed more spheroid shapes, indicating that the shapes chosen as most stable at 0 K may not be the most stable at higher temperatures, particularly close to melting temperatures.

Each particle reacted slightly differently with regards to melting behaviour according to its size. Smaller particles melted faster. Particles showed surface disordering which decreased their vacancy formation energies at temperatures up to a 100 K below melting point, and the binding energies inside the bulk of the nanocrystals decreased due to nanoparticle expansion.

The surface energies of the {100} faces acted much as in bulk material with a {100} surface when simulated with periodic boundaries, but edges and corners showed interesting behaviour due to interactions between different orientated surfaces and changes in shape with increased temperatures. These properties can have interesting implications in directed segregation and creation of complex nanoparticles.

## 7.6 References

[1] Y. H. Wang, P. L Chen and M. H. Liu, *Nanotechnology* **17** (2006): 6000.

[2] R. A. Molecke, “Characterization, modeling, and simulation of Multiscale directed-assembly systems” (PhD diss., The University of New Mexico, 2011).

[3] Alavi, Thompson, *Journal of Physical Chemistry A* **110** (2006): 1518.

[4] L. V. Sang, V. V. Hoang and N. T. T. Hang, *The European Physical Journal D* **67** (2013): 64.

[5] F. Ercolessi (October 9, 1997). “Time integration algorithm,” accessed February 6, 2011, http://www.fisica.uniud.it/~ercolessi/md/md/node20.html.

[6] R. N. Barnett and U. Landman, *Physical Review B: Condensed Matter* **44** (1991): 3226.

[7] F. J. Resende, V. E. Carvalho, B. V. Costa and C. M. C. de Castilho, *Brazilian Journal of Physics* **34** (2004): 414.

[8] C. van der Walt, J. J. Terblans and H. C. Swart. *Computation Materials Science* **83** (2014): 70.

[9] S. I. Abdul-Hadi, “The Melting of Nanoscale Pd Clusters: A Molecular Dynamics Study Using the Modified Embedded Atom Method,” (MSc thesis, An-Najah National University, 2003).

[10] Y. Bai, R. Long, C. Wang, M. Gong, Y. Li, H. Huang, H. Xu, Z. Li, M. Deng and Y. Xiong, *Journal of Materials Chemistry A* **1** (2013): 4228.

# Chapter 8 - Diffusion Dynamics in Pure Cu and Ag Crystals

## 8.1 Introduction

In this chapter, the properties of pure Cu and pure Ag were studied, in preparation for the final chapter, where the mixed potential was used to study segregation and diffusion in mixed Cu/Ag. Using the Sutton-Chen potential, Molecular Dynamics simulations were done of Ag and Cu bulk and nanocrystals and the vacancy formation energy, migration energy, and diffusion activation energy were calculated. The migration energy along a diffusion path was studied for different low index surface orientations. With these energy values, interactions in pure metals and mixed metals will be compared in the following chapter.

Part of accurately modelling thermodynamic properties of modelled nanoparticles in this chapter included the optimization of the Sutton Chen parameters used in further calculations. In previous chapters, only the vacancy formation energy $E_v$ and cohesion binding energy $E_{coh}$ were calculated. In the following two chapters, the calibration of the Sutton-Chen parameters are revisited in order to optimize for the calculation of the migration energy $E_m$, diffusion activation energy $Q$, and, in the next chapter, segregation energy $\Delta G$. All energies are given per atom.

## 8.2 Methods

Using large crystals of Ag and Cu with a {100} surface to approximate a large bulk crystal, the vacancy formation energy, migration activation energy, diffusion activation energy and segregation energies were determined using the original Sutton-Chen parameters, the Rafii-Tabar Sutton adjustment [1], the Classical Sutton Chen parameters, and the Quantum Sutton Chen parameters respectively. By comparing the data to literature values, the best set of parameters could be chosen.

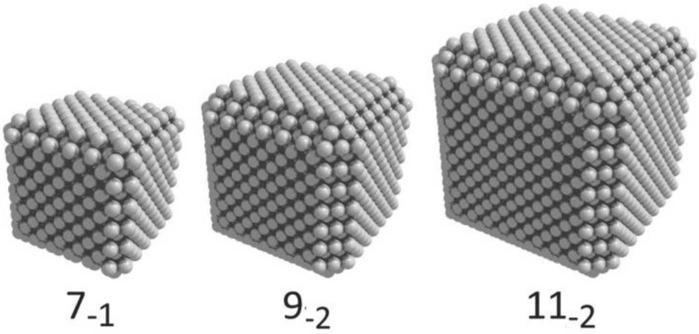


Figure 8.1. Three sizes of rhombicuboctahedrons were simulated, such as the Ag nanoparticles pictured here, to investigate the effect of crystal size on the various energies calculated with the Sutton-Chen potential. The crystal denoted $7_{-1}$ indicates a $7 \times 7 \times 7$ packed crystal where one row of atoms has been removed from the edges and corners.

Employing the most effective parameters for calculating the energies for Cu and Ag, the influence of open surfaces could then be determined, by first simulating large crystals with {110}, {111} and {100} free surfaces to approximate bulk crystals, and then nanoparticles, as previously produced in a lab and characterized in Chapters 6 and 7. Based on these studies [2, 3], rhombicuboctahedrons with a stable form were chosen, which have six {100} faces, slight edges of {110} and corners with {111} orientated surfaces. Particles below 1 nm in diameter were unstable and so were thereafter ignored. The nanoparticles shown in Figure 8.1 ranged from 1 nm to 4 nm in diameter, where crystals began to display melting behaviour similar to bulk crystals. In addition to vacancy formation energies $E_v$ for each surface orientation, the migration activation energies, $E_m$ and from the sum of $E_v$ and $E_m$, the diffusion activation energies $Q$, for each respective surface orientation were also determined.

## 8.3 Comparing Sutton-Chen Parameters

The four different sets of parameters for the Sutton-Chen model are given in Table 8.1. Each set was used to simulate large Ag and Cu crystals representing a bulk crystal with (100) free surface at 0 K, similar to the (100) crystal shown in Figure 4.3. For each set of parameters, the cohesion binding energy, the migration activation energy, and the vacancy formation energy were calculated, and by extension, the diffusion activation energy $Q$, where

$$Q = E_v + E_m$$

as given by equation 2.12.

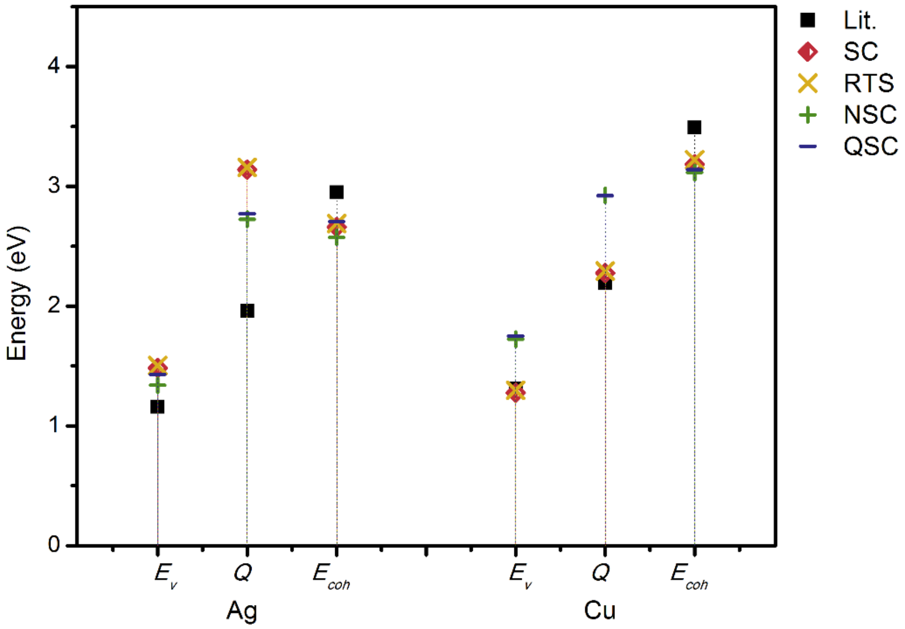


Figure 8.2. Energy values as obtained from the Sutton-Chen parameters tabled in Table 1 and compared to values from literature. Literature values of $E_{coh}$ are from reference [6], $E_v$ from reference [7], and Q is from reference [8].

Table 8.1 The Sutton–Chen parameters (SC), Rafii-Tabar – Sutton parameters (RTS), Classical Sutton-Chen (NSC) and Quantum Sutton-Chen (QSC) for Cu, and Ag [1, 4, 5].

| **Element** | **Parameters** | ***ε* (eV)** | ***a* (Å)** | ***c*** | ***n*** | ***m*** |
|---|---|---|---|---|---|---|
| **Cu** | SC | $1.2382\ 10^{-2}$ | 3.6100 | 39.432 | 9 | 6 |
| | RTS | $1.2351\ 10^{-2}$ | 3.6100 | 39.756 | 9 | 6 |
| | NSC | $5.9066\ 10^{-3}$ | 3.6030 | 83.073 | 10 | 5 |
| | QSC | $5.7921\ 10^{-3}$ | 3.6030 | 84.843 | 10 | 5 |
| **Ag** | SC | $2.5415\ 10^{-3}$ | 4.0900 | 144.410 | 12 | 6 |
| | RTS | $2.5330\ 10^{-3}$ | 4.0900 | 145.658 | 12 | 6 |
| | NSC | $4.0072\ 10^{-3}$ | 4.0691 | 94.948 | 11 | 6 |
| | QSC | $3.9450\ 10^{-3}$ | 4.0691 | 96.524 | 11 | 6 |

The resulting cohesion binding energies, vacancy formation energies and activation energies are given in Figure 8.2, along with experimental literature values [6-8]. Statistical analysis calculating deviation from the mean indicated that the Quantum Sutton Chen parameters gave the closest approximation to values obtained in literature. However, it is clear that these parameters, which were optimized to calculate the lattice parameter, cohesion energy, phonon dispersion, elastic constants, bulk modulus, surface energy and vacancy formation energy, deviate on diffusion activation energy values. Diffusion orientated recalibration of

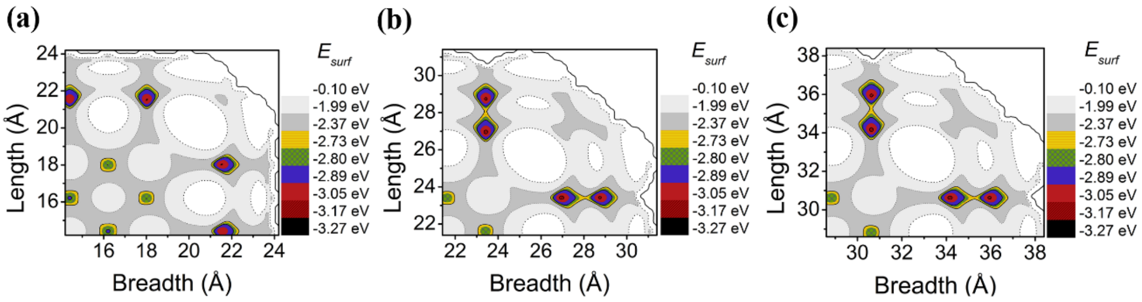


Figure 8.3 (a-c). A portion of the surface energy of the Cu $7\times7\times7-1$ particle (denoted $7_{-1}$), the $9\times9\times9-2$ particle (denoted $9_{-2}$), and the $11\times11\times11-2$ particle (denoted $11_{-2}$) surface showing the different surface orientations.

Sutton-Chen parameters for specific use in calculating diffusion-related energies is worth future consideration.

# 8.4 Diffusion Energies in Bulk and Nano-crystals

The following section investigated the vacancy formation energy, migration energy, and the diffusion activation energy of Cu and Ag bulk- and nanocrystals in detail.

## 8.4.1 Vacancy Formation Energy

In order to simulate bulk crystal reactions, large crystals of approximately 2 000 atoms each were created, with free surfaces of each of the three surface orientations (100), (110) and (111), similar to those in section 5.4.1.2 shown in Figure 5.1 (a-c).

Considering Figure 8.3 (a-c), the free surfaces of the representative section of the three nanocrystals of Figure 8.1 reveal a similar trend, with the most shallow, weakest energies on the {111} corners, and the deepest, strongest binding energies in the troughs of the {110} edges, with the binding points on the {100} faces falling in between. The energies of the $9_{-2}$ rhombicuboctahedron in Figure 8.3 (b) and the $11_{-2}$ particle in Figure 8.3 (c) look the same. By following the method explained in section 2.2.3 and noting the differences in total system energy, the extraction energy $E_{extr}$ and surface energy $E_{surf}$ were obtained and plotted in Figure 8.4 for Ag and Figure 8.5, for Cu. $E_{extr}$ was calculated by taking the differences between a perfect crystal and the crystal with a point vacancy, and $E_{surf}$ was obtained by adding the extracted atom, or adatom, back onto the surface at the strongest binding points and noting the energy difference. In Figure 8.4 and Figure 8.5, in both Ag

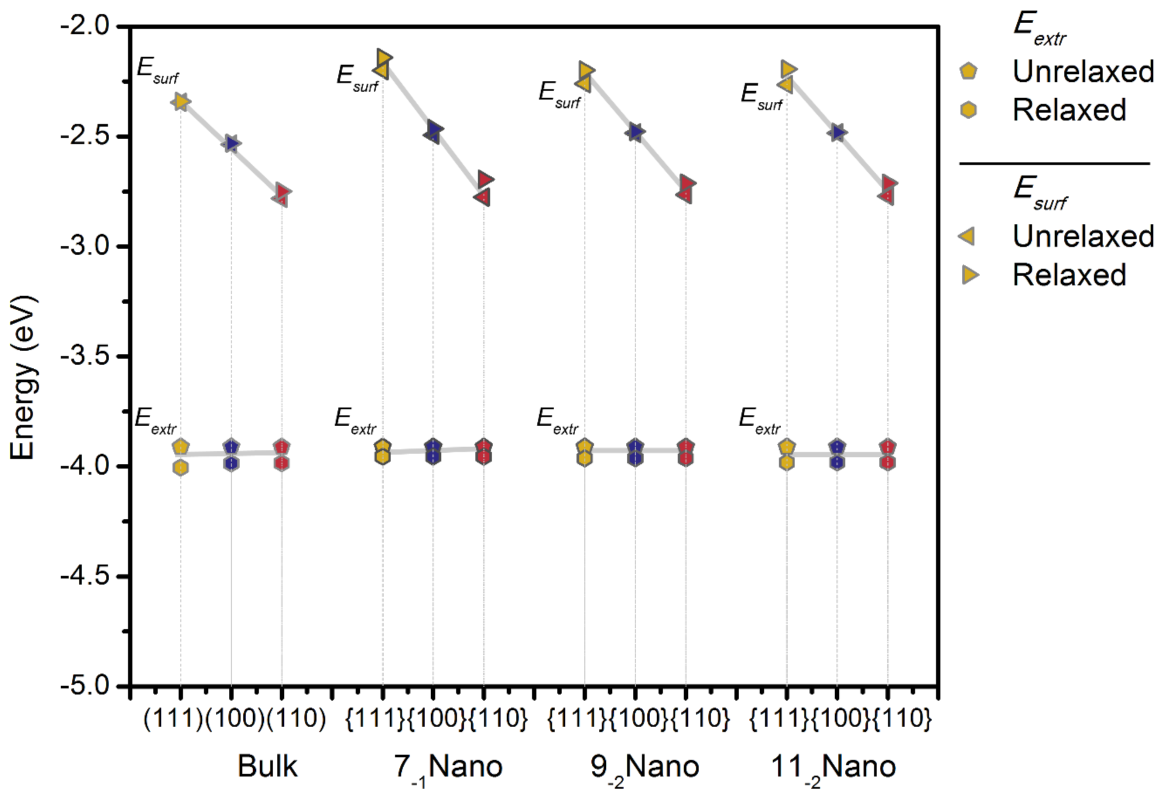


Figure 8.4. The surface energies and extraction energies of the adatom for the various Ag crystals.

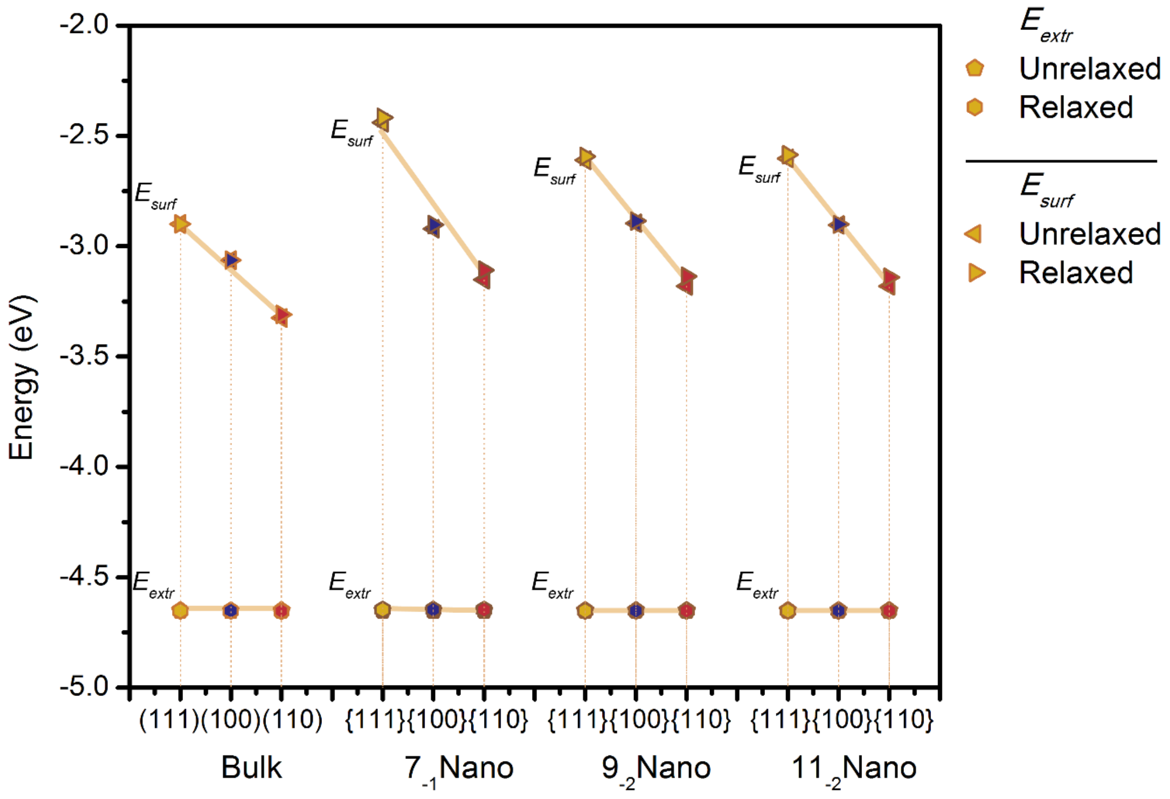


Figure 8.5. The surface energies and extraction energies of the adatom for the various Cu crystals.

and Cu respectively, the extraction energy is independent of surface orientation, as it is too far removed from a free surface, while the surface energy depends upon the surface orientation, and $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$.

The surface dependence is more pronounced in the smallest rhombicuboctahedrons, where the {111} surface is on an interface between other orientations. It is worth noting that whereas the Cu $E_{extr}$ values barely changed after the crystal was relaxed for 20 000 steps, the Ag values changed slightly, indicating that the crystal relaxed further.

## 8.4.2 Migration Energy

***Bulk Crystals***

From the extraction and surface energies, the vacancy formation energy can be calculated. To find the diffusion activation energy $Q$ from equation 2.12, the migration energy of the atom also needs to be determined. Following the original diffusion path shown in Figure 4.24, the energy differences of a crystal with a single point vacancy, where an atom is migrating to that vacancy, was noted. The maximum peak height of energy increase corresponds to the migration activation energy. Figure 8.6 shows the energy profiles of all 3 bulk crystals for Ag atoms migrating from one layer to another. The calculation yields close to 50 points for each peak, thus every second point was plotted for better clarity.

The depth profile is not continuous as, after the atom has migrated into the point vacancy, for the atom to proceed in the same direction, the vacancy must move from behind it to ahead in its path, as seen in Figure 8.8 (a). The vacancy being before, and then after, the migrating atom, correspond to two different states, causing the discontinuity. On the other hand, Figure 8.7 follows the vacancy on a continuous path as different atoms move through it, shown in Figure 8.8 (b). As there is no change in the crystal state, there is no discontinuity.

It is further pertinent to note that the different surface orientations have different activation energies and turning points. In Figure 8.6 the left-most minimum corresponds to the atom being in the surface layer. The diffusion path followed by this atom in the (110) surface is exposed to the surface for two steps of its migration, namely, from the crest into a trough, and from the trough into the subsurface layer. As the atom moves to the subsurface layer, the vacancy moves to the surface, which is the most stable configuration of the crystal containing a point defect.

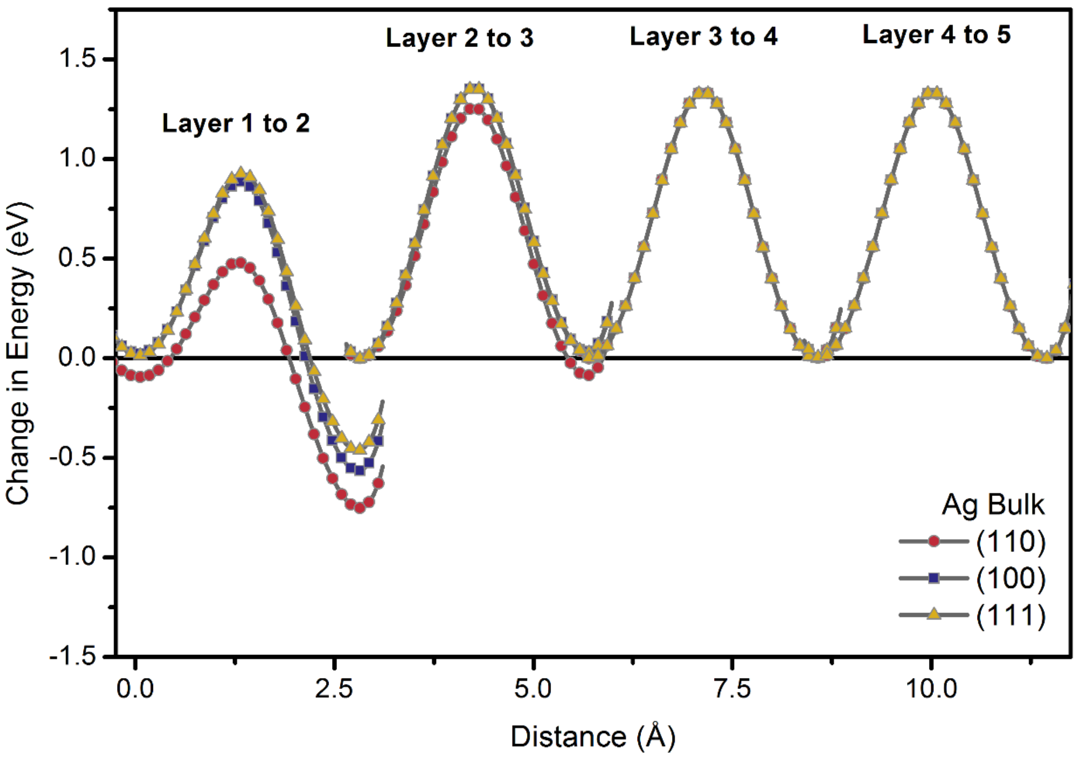


Figure 8.6 shows the change in crystal energy following the diffusion path of a migrating atom in Ag crystals with (110)-, (100) and (111) free surfaces.

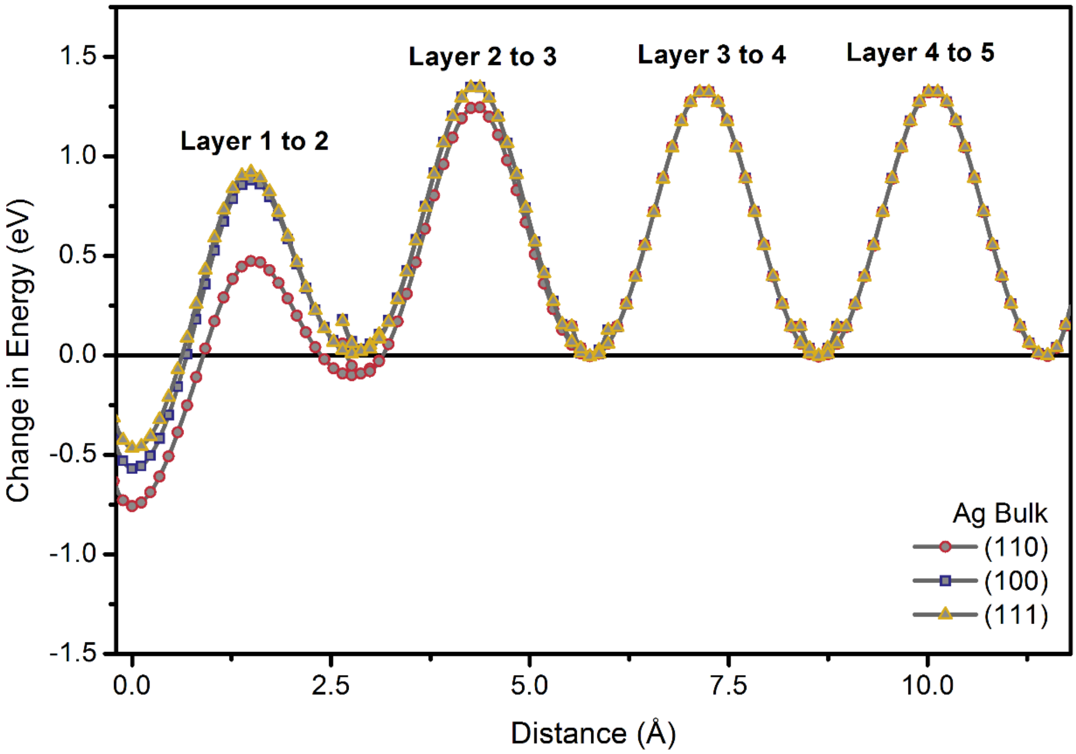


Figure 8.7 shows the change in energy following the segregation path of a point vacancy in the Ag bulk crystals.

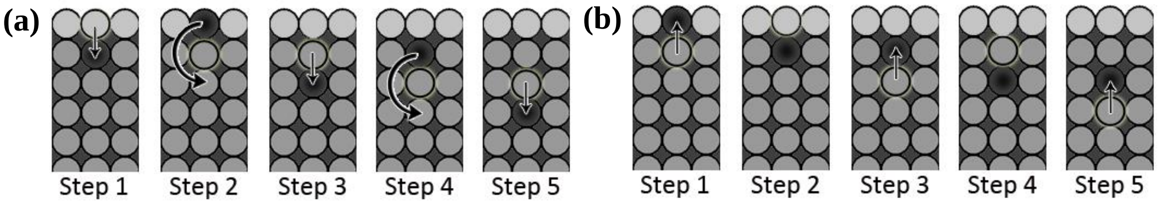


Figure 8.8 (a, b). The route of an atom as it follows a diffusion path inwards from the surface is shown in (a), while (b) shows the same process from the perspective of the vacancy's diffusion path. In (a) The process starts where a subsurface vacancy is created to open a path for diffusion for the surface atom. In step 2, after the surface atom moved into the vacancy, the vacancy is exchanged with a position further down into the crystal, so that the diffusing atom can continue ever deeper. The atom moves into the new vacancy in step 4, and step 3 is repeated in step 5. In (b), the same process is seen from the perspective of the vacancy. In this case the atom migrated upwards into the vacant position, and the vacancy shifts downwards (step 2). Step 1 and 2 are repeated in 3 and 4, and so on.

This accounts for the much lower minimum on the right-hand side of the peak. This is also borne out by investigating the vacancy segregation profile in Figure 8.7 which clearly shows the vacancy's preference for being in the surface layer which produces a much lower minimum. It is also interesting to note that the (110) surface shows lower activation energies close to the surface and a higher tendency for vacancies to want to segregate to that surface. The migration profiles of adatoms and vacancies in Cu bulk crystals resembled those of Ag in Figure 8.6 and Figure 8.7.

***Nano-crystals***

In comparison, the energy depth profile of the diffusion path measured on the Cu rhombicuboctahedrons shown in Figure 8.9-Figure 8.11 are like the bulk in the deeper layers, with slight differences appearing on the surface. The most marked difference was the {111}, and {110} diffusion path energies in the $7_{-1}$ imperfect cube of Figure 8.9. As the atoms on these surfaces are on the edges between different surface orientations, and are also sitting at the extreme edges of the crystal with few bonds to first, second, etc., nearest neighbours, it makes sense that removing one of these nearest neighbours to create a point defect will change the crystal energy drastically, and more so if the vacancy has migrated to take the place of the unstable edge atom. As such, it is no surprise that the lowest minimum turning point occurs when the vacancy occupies the position of the {111} migrating atom which is located on the corner of the crystal. Once again, the activation energy for the {110} surface migration path is the lowest in all three crystals, but for the smallest crystal, the activation energy for subsurface-to-surface migration in {111} and {110} is lower than in

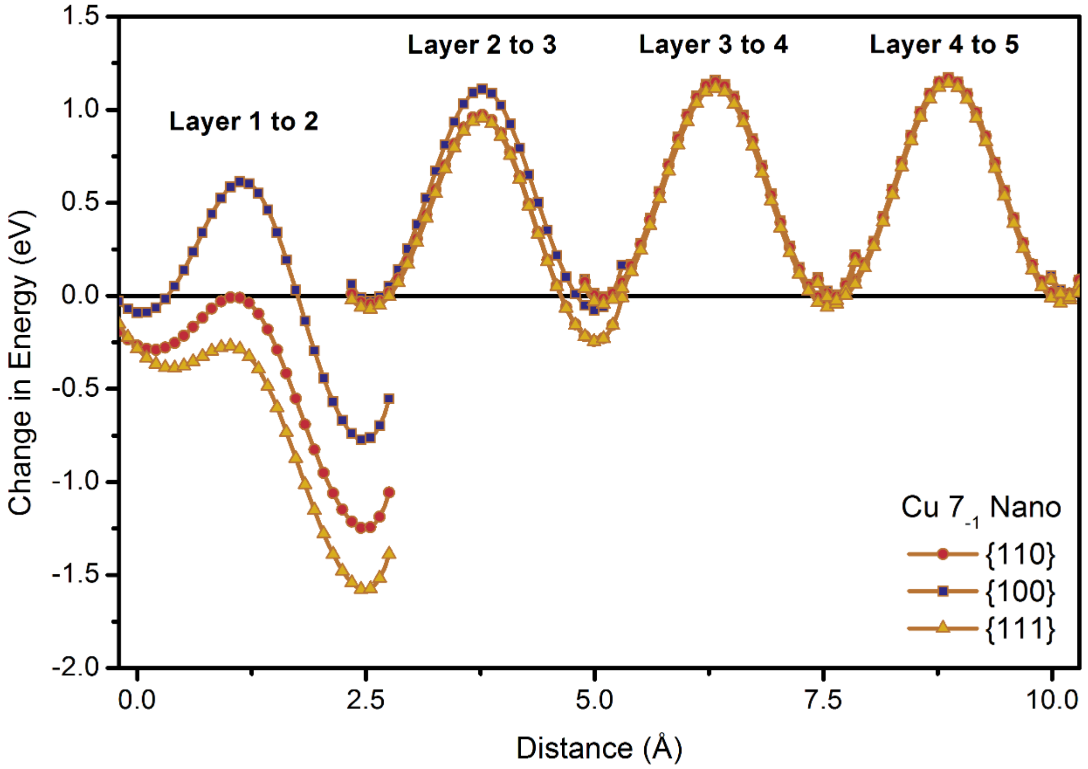


Figure 8.9. The change in crystal energy following the diffusion path of a migrating atom from three different positions in the Cu $7_{-1}$ imperfect cube.

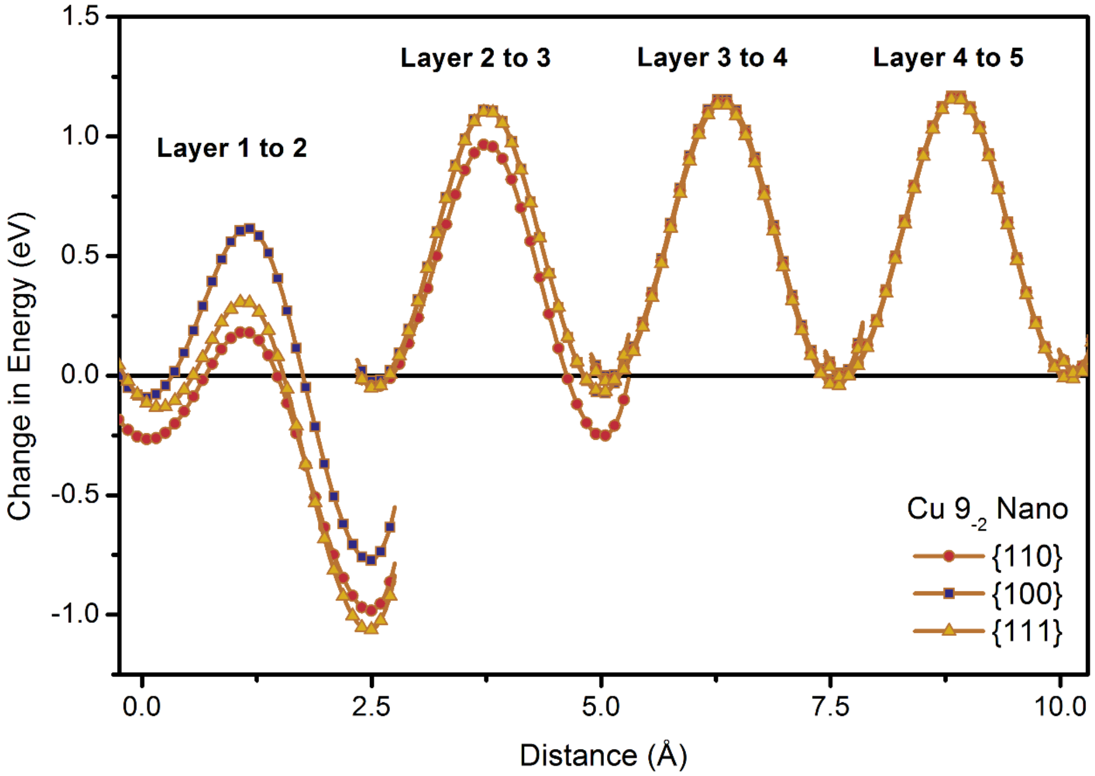


Figure 8.10. The same change in crystal energy following the diffusion path of a migrating atom in the Cu $9_{-2}$ particle.

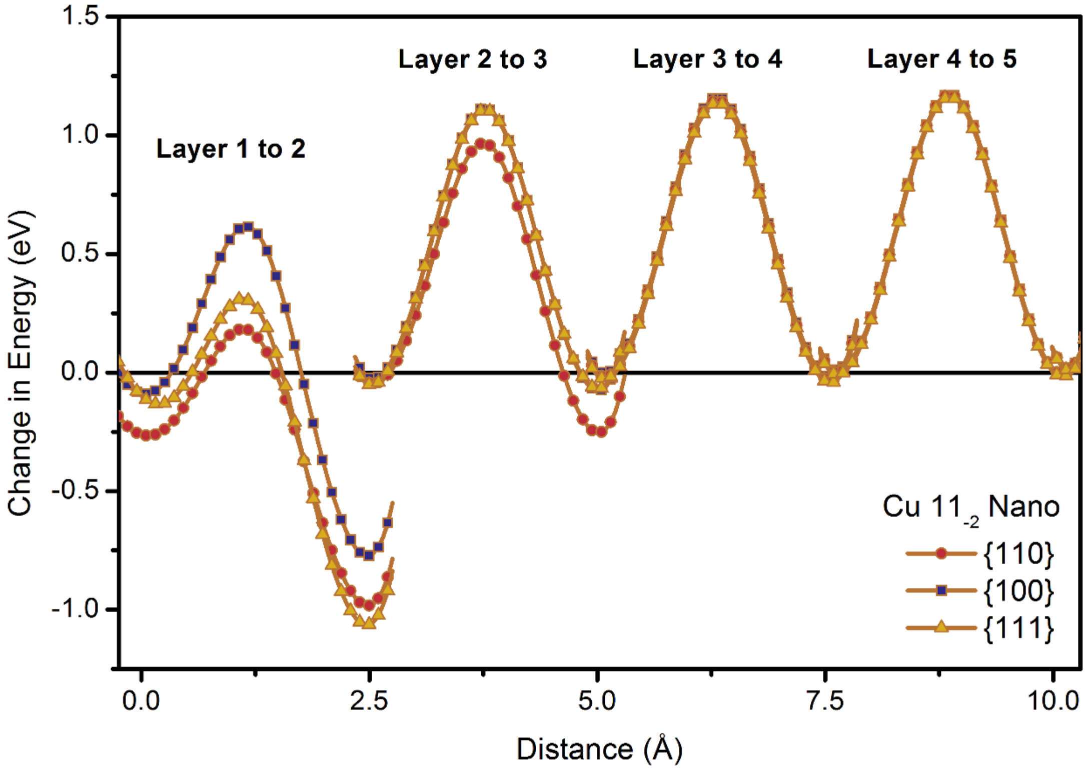


Figure 8.11 shows the change in crystal energy following the diffusion path of a migrating atom in the Cu $11_{-2}$ imperfect cube, which resembles the energy profile for the $9_{-2}$ imperfect cube.

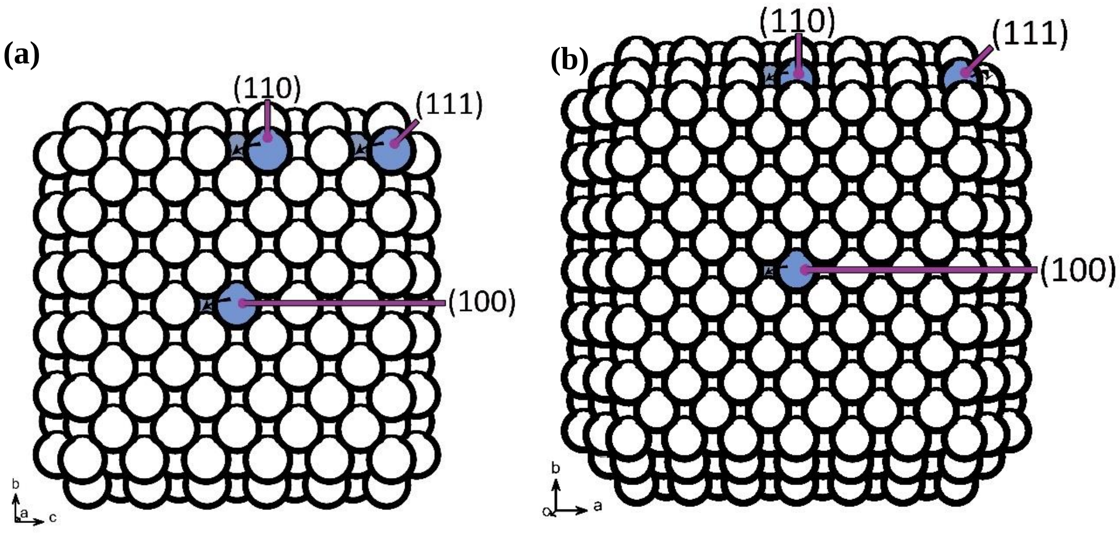


Figure 8.12 (a, b). The surface and subsurface atoms chosen for the diffusion paths on the $7_{-1}$ imperfect cube in (a), and the $9_{-2}$ imperfect cube in (b).

the other two nanocrystals. The chosen diffusion paths from the various surfaces of the $7_{-1}$ crystal are shown in Figure 8.12 (a).

The {111} surface atom appears particularly unstable, and as the next atom down is a trough atom of the {110} edge, it has a similar energy to that of the {110} diffusion path's trough atom. There is little difference between the $9_{-2}$ and $11_{-2}$ particle profiles shown in Figure 8.10 and Figure 8.11, and their diffusion paths also differ slightly from the smaller rhombicuboctahedron, as shown in Figure 8.12 (b). In particular, the {111} subsurface atom is obscured by a larger tightly packed {111} surface, so that the migration energies measured for the subsurface position approximately resemble those of the bulk crystal. However, the surface atom is still on the edge of the {111} surface, and that gives a slight difference to the bulk values. The {100} atom diffusion path energy profile resembles the bulk (100) energy profile in all three crystals. The similarities in the $9_{-2}$ and $11_{-2}$ crystal profiles suggest that the differences are influenced by crystal shape but not crystal size. Atoms found on the borders between surface orientations will most likely have energy profiles which deviate from bulk values, and those in large surfaces will resemble bulk profiles.

### 8.4.3 Diffusion Activation Energy

Figure 8.13 and Figure 8.14 show the vacancy formation energy $E_v$, migration energy in the bulk $E_m$ as the amplitude of the migration peaks deep within the crystal, and diffusion activation energy $Q$. These values were calculated using the data tabulated in Table 8.2 and presented in Figure 8.4, Figure 8.6 and Figure 8.9. As the migration energy is measured in the bulk, it is not surface orientation dependent. However, since $E_v$ is surface orientation dependent, with $E_v^{(110)} < E_v^{(100)} < E_v^{(111)}$, $Q$ shows the same dependency, with $Q^{(110)} < Q^{(100)} < Q^{(111)}$. This can be seen in both Figure 8.13 and Figure 8.14. This relationship is more pronounced in the smaller crystal, especially in the Cu $7_{-1}$ rhombicuboctahedron shown in Figure 8.14. However, as the {111} surface is at an extreme corner of the crystal at a junction between {100}- and {110}-surfaces, this is most likely because of edge-effects and not crystal size.

Once again Ag values changed slightly as the crystal was allowed to relax for 20 000 steps, slightly increasing the activation energies. This may be due to unoptimized QSC fitting parameters for Ag. There is good agreement with the experimental literature values of $E_v$ obtained from electrical resistivity measurements [7], although the $Q$ values from

Table 8.2 Summary of calculated and literature diffusion parameters for Ag and Cu crystals. All energies are given per atom.

| | | $E_{extr}$ **(eV)** | | $E_{surf}$ **(eV)** | | $E_v$ **(eV)** | | $E_m$ **(eV)** | | $Q$ **(eV)** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** |
| **Ag** | Bulk (111) | -3.91 | -4.01 | -2.34 | -2.34 | 1.57 | 1.66 | 1.33 | 1.33 | 2.90 | 2.99 |
| | Bulk (100) | -3.91 | -3.99 | -2.54 | -2.53 | 1.37 | 1.46 | 1.33 | 1.33 | 2.71 | 2.79 |
| | Bulk (110) | -3.91 | -3.99 | -2.78 | -2.75 | 1.13 | 1.24 | 1.33 | 1.33 | 2.46 | 2.57 |
| | Lit | | | | | | 1.16<br>[7] | | | | 1.96<br>[8] |
| | Nano7-1 {111} | -3.91 | -3.96 | -2.20 | -2.14 | 1.71 | 1.81 | 1.33 | 1.33 | 3.04 | 3.14 |
| | Nano7-1 {100} | -3.91 | -3.96 | -2.49 | -2.47 | 1.42 | 1.49 | 1.33 | 1.33 | 2.75 | 2.82 |
| | Nano7-1 {110} | -3.91 | -3.96 | -2.78 | -2.70 | 1.13 | 1.26 | 1.33 | 1.33 | 2.47 | 2.59 |
| | Nano9-2 {111} | -3.91 | -3.96 | -2.26 | -2.20 | 1.65 | 1.76 | 1.33 | 1.33 | 2.98 | 3.09 |
| | Nano9-2 {100} | -3.91 | -3.96 | -2.49 | -2.48 | 1.43 | 1.49 | 1.33 | 1.33 | 2.76 | 2.82 |
| | Nano9-2 {110} | -3.91 | -3.96 | -2.77 | -2.71 | 1.15 | 1.25 | 1.33 | 1.33 | 2.48 | 2.58 |
| | Nano11-2 {111} | -3.91 | -3.98 | -2.26 | -2.19 | 1.65 | 1.79 | 1.33 | 1.33 | 2.98 | 3.12 |
| | Nano11-2 {100} | -3.91 | -3.98 | -2.48 | -2.48 | 1.43 | 1.50 | 1.33 | 1.33 | 2.76 | 2.83 |
| | Nano11-2 {110} | -3.91 | -3.98 | -2.77 | -2.71 | 1.14 | 1.27 | 1.33 | 1.33 | 2.47 | 2.60 |
| **Cu** | Bulk (111) | -4.65 | -4.65 | -2.90 | -2.90 | 1.75 | 1.75 | 1.18 | 1.18 | 2.93 | 2.93 |
| | Bulk (100) | -4.65 | -4.65 | -3.06 | -3.06 | 1.59 | 1.59 | 1.18 | 1.18 | 2.77 | 2.77 |
| | Bulk (110) | -4.65 | -4.65 | -3.33 | -3.31 | 1.33 | 1.34 | 1.18 | 1.18 | 2.50 | 2.52 |
| | Lit | | | | | | 1.31<br>[7] | | | | 2.19<br>[8] |
| | Nano7-1 {111} | -4.65 | -4.65 | -2.44 | -2.42 | 2.21 | 2.23 | 1.17 | 1.18 | 3.38 | 3.41 |
| | Nano7-1 {100} | -4.65 | -4.65 | -2.92 | -2.90 | 1.73 | 1.74 | 1.17 | 1.19 | 2.90 | 2.93 |
| | Nano7-1 {110} | -4.65 | -4.65 | -3.15 | -3.11 | 1.50 | 1.54 | 1.17 | 1.19 | 2.67 | 2.72 |
| | Nano9-2 {111} | -4.65 | -4.65 | -2.61 | -2.59 | 2.04 | 2.06 | 1.17 | 1.17 | 3.21 | 3.23 |
| | Nano9-2 {100} | -4.65 | -4.65 | -2.90 | -2.89 | 1.75 | 1.77 | 1.17 | 1.17 | 2.93 | 2.94 |
| | Nano9-2 {110} | -4.65 | -4.65 | -3.18 | -3.14 | 1.47 | 1.51 | 1.17 | 1.17 | 2.64 | 2.69 |
| | Nano11-2 {111} | -4.65 | -4.65 | -2.60 | -2.59 | 2.05 | 2.07 | 1.17 | 1.17 | 3.22 | 3.24 |
| | Nano11-2 {100} | -4.65 | -4.65 | -2.90 | -2.90 | 1.75 | 1.75 | 1.17 | 1.17 | 2.92 | 2.92 |
| | Nano11-2 {110} | -4.65 | -4.65 | -3.18 | -3.14 | 1.47 | 1.51 | 1.17 | 1.17 | 2.64 | 2.68 |

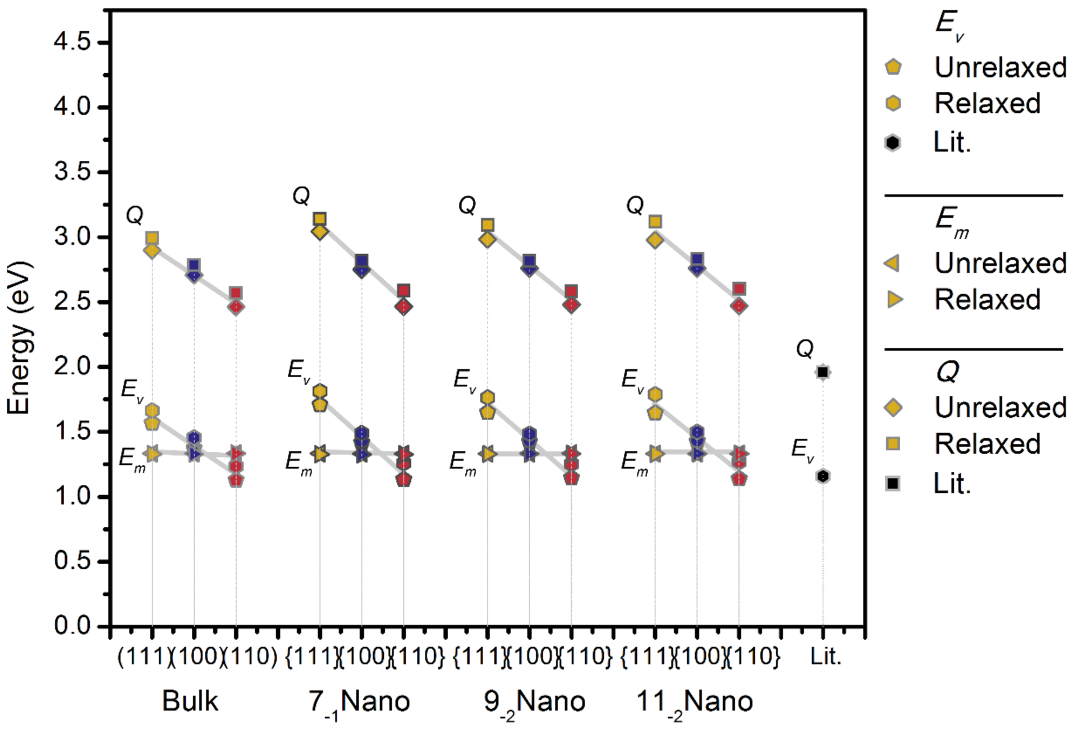


Figure 8.13. The vacancy formation energy $E_v$, migration energy in the bulk $E_m$ and diffusion activation energy $Q$ for the various Ag crystals. The literature values of $E_v$ are from reference [7], and $Q$ values from reference [8].

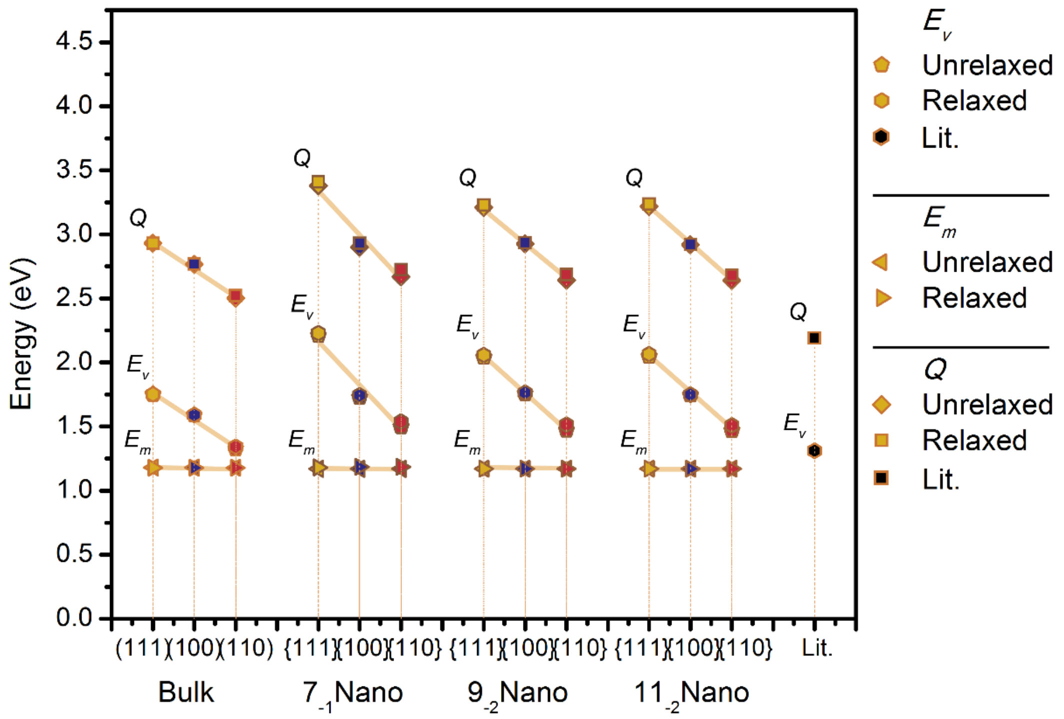


Figure 8.14. The vacancy formation energy $E_v$, migration energy in the bulk $E_m$ and diffusion activation energy $Q$ for the Cu crystals. Literature values of $E_v$ are from reference [7], and $Q$ from reference [8].

radioactive tracer diffusion data [8] show less correspondence. This is most likely due to two factors: the fitting of the QSC parameters, suggesting the parameters are not optimized to model migration in the crystal, and the artificially large $E_m$ value, as a result of freezing the crystal during migration activation energy measurements. Fitting SC parameters for calculating diffusion activation energies, and allowing the crystal to relax during migration energy measurements, could improve the ability of the model to predict diffusion activation energies

## 8.5 Summary

A thorough investigation of diffusion energies at 0 K in pure Ag and Cu was conducted using the Sutton-Chen potential. Fitting different Sutton-Chen parameters to literature diffusion data indicated that the Quantum Sutton-Chen provided the results closest to experimental values for all the energies studied in Cu and Ag. However, the QSC was not able to model the Ag diffusion activation energy $Q$ well. Nevertheless, modelling different bulk and nanocrystals of Ag and Cu showed that certain energies were dependent on the surface orientation, because a component of that energy depended on a surface energy $E_{surf}$, and $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$. As this was a component of the vacancy formation energy $E_v$, it is also surface orientation dependent, and $E_v^{(110)} < E_v^{(100)} < E_v^{(111)}$. $E_v$ is used to calculate the diffusion activation energy $Q$, and thus it, too, is surface orientation dependent with $Q^{(110)} < Q^{(100)} < Q^{(111)}$. The other crystal energies, the extraction energy $E_{extr}$, and the migration energy in the bulk $E_m$, are measured deep in the crystal and are thus surface orientation independent. However, close to the surface the migration peaks and troughs of the energy profile have a similar surface-orientation dependent character in the bulk crystals. In nanocrystals, the surface-orientation dependence is also influenced by edge effects caused by the proximity of other orientations, and whether a surface is on a sharp edge or corner with fewer nearest neighbours. Relaxing the crystals changed the Ag crystal energies more significantly than the Cu crystals, and the diffusion values calculated corresponded less to literature values, suggesting that the QSC parameters are not ideally suited to simulating diffusion in Ag. These energy relationships indicate that in self-diffusion and vacancy segregation, the surface orientation influences diffusion. In rhombicuboctahedrons, with the lower subsurface activation energy in {111}, vacancies will preferentially migrate there.

## 8.6 References

[1] H. Rafii-Tabar and A. P. Sutton, *Philosophical Magazine Letters* **63** (1991): 217.

[2] C. van der Walt, J. J. Terblans and H. C. Swart, *Computation Materials Science* **83**, (2014): 70.

[3] C. van der Walt, J. J. Terblans and H. C. Swart. (2016). “Calculated nanocube vacancy formation energy and cohesion energy at 0 K,” manuscript submitted for publication.

[4] Y. Kimura, Y. Qi, T. Çağin and W. A Goddard III (1998). “The quantum Sutton-Chen many-body potential for properties of fcc metals,” *Physical Review*, manuscript submitted for publication.

[5] A. P. Sutton and J. Chen, *Philosophical Magazine Letters* **61** (1990): 139.

[6] C. Kittel, *Introduction to Solid State Physics*, 8th ed. (New York: John Wiley & Sons Inc., 2005).

[7] J. J. Wollenberger, *Physical Metallurgy*, eds. R. W. Cahn and P. Haasen (Amsterdam, North-Holland: Elsevier Science B. V., 1983).

[8] R. C. Weast, M. J. Astle, eds., *CRC Handbook of Chemistry and Physics*, 63rd ed. (Boca Raton, FL: CRC Press, 1982).

# Chapter 9 - Diffusion and Segregation in Alloy Cu and Ag Crystals

## 9.1 Introduction

Having investigated the diffusion parameters as modelled by the Sutton-Chen Molecular Dynamics simulation for pure Cu and Ag bulk and rhombicuboctahedrons in the previous chapter, the next step is to investigate its mixing potential and its ability to model segregation which is known to occur in Ag and Cu mixtures. Using the mixed form of the potential for bimetallic interactions with a slight adjustment, the interactions between Ag and Cu were simulated. As before, vacancy formation energies were calculated. Migration energy depth profiles and energies allowed the calculation of the diffusion activation energy. Additionally, segregation energies at different depths were studied. These energies were evaluated for a foreign adatom of Ag in the Cu crystals, and a Cu adatom in the Ag crystals. The investigation of the segregation energy $\Delta G$ allowed the modelling of the change in free energy of mixed Ag and Cu in bulk and nanocrystals. The segregation energies were calculated by starting with the foreign adatom at the surface and successively exchanging it with a first nearest neighbour deeper within the crystal along the diffusion path, noting the change in free crystal energy each time. All energies are given per atom. The final part of this chapter presents the outcomes of a time-lapse relaxation run of a mixed rhombicuboctahedron of Cu/Ag, proving definitively whether the new mixed potential can model surface segregation in this alloy.

## 9.2 Calculated Diffusion Energies

This section revisited the diffusion energies from section 8.4, calculated for a foreign element adatom. The vacancy formation energy, migration energy and diffusion activation energy were calculated for a Cu atom in Ag bulk- and nanocrystals, and the same was done for a Ag atom in Cu bulk- and nanocrystals.

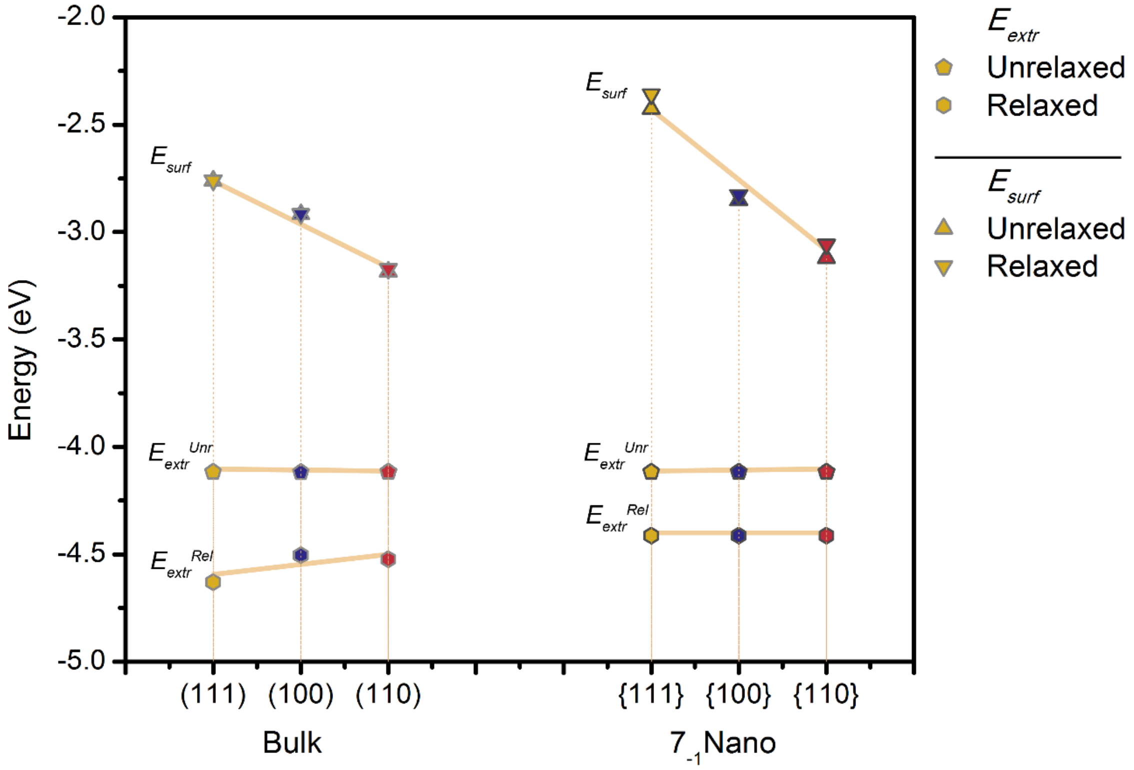


Figure 9.1. The surface energies and extraction energies of the Ag adatom for the Cu bulk and nanocrystals.

## 9.2.1 Vacancy Formation Energy

For the mixed metals, only the energies of the bulk crystals and the $7_{-1}$ rhombicuboctahedron shown in Figure 8.12 (a) were investigated, as the values of the smallest crystal diverged the most from bulk values and provide values that can be contrasted and compared.

As before with the pure metals, an adatom was extracted from the bulk and then brought to the surface, though in this case, in a Cu crystal the adatom was Ag, and in the Ag crystal the adatom was Cu. In Figure 9.1 and Figure 9.2 the same trends from Figure 8.4 and Figure 8.5 are repeated, with the same orientation dependence for the surface energy, as would be expected from our inspection of Figure 5.1 and Figure 8.3. Even though the adatom element has changed in Figure 9.1 and Figure 9.2, the values are still similar to those for the pure metals, as shown in Figure 8.4 and Figure 8.5. However, the Ag adatom's extraction changed dramatically by up to 0.51 eV/atom when the crystal was relaxed, suggesting a distortion in the lattice as the Ag atom pushed against the more closely packed Cu lattice. The extraction energy for the Ag atom in Figure 9.1 became significantly more negative, meaning that it binds more strongly in the bulk.

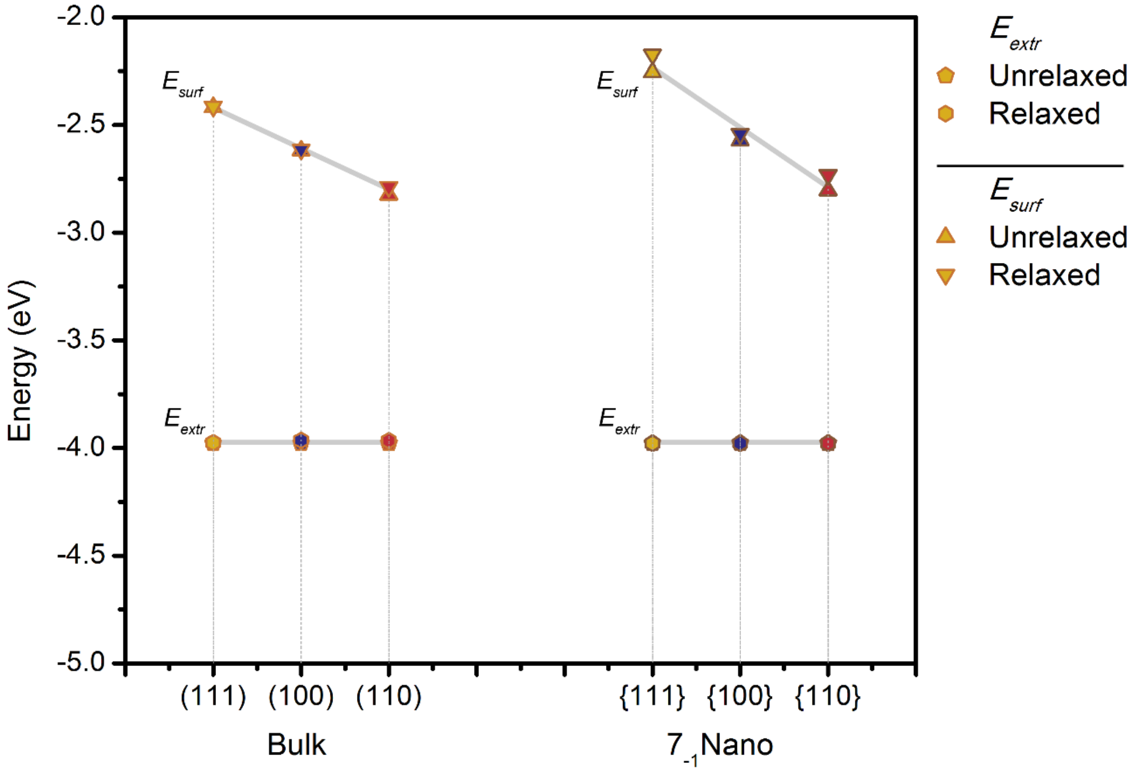


Figure 9.2. The surface energies and extraction energies of the Cu adatom for the Ag bulk and nanocrystals.

## 9.2.2 Migration Energy

With the vacancy formation energy components calculated, the bulk migration energy is next. Once again, a similar method to the process in Chapter 8 is followed, but the migrating atom's element is set to Ag in the Cu crystal, and Cu in the Ag crystal, such that there is only one foreign element in the crystal at one time.

### 9.2.2.1 Ag Atom Migration in Cu Crystals

***Bulk Crystals***

As in the previous chapter, the energy depth profile following the diffusion path of the foreign element shows a discontinuity where the vacancy is swopped out. In Figure 9.3 the (110) energies are not as distinct from the other orientations for a Ag atom in a Cu bulk crystal, though the vacancy still has a slight preference for segregating to the (110) surface, as seen Figure 9.4. Another interesting feature of this profile is that as the atom of the migrating element is changed so there is only one foreign element in the crystal at a time; the crystal state is slightly changed from one jump to the next. As seen in Figure 9.5, closer

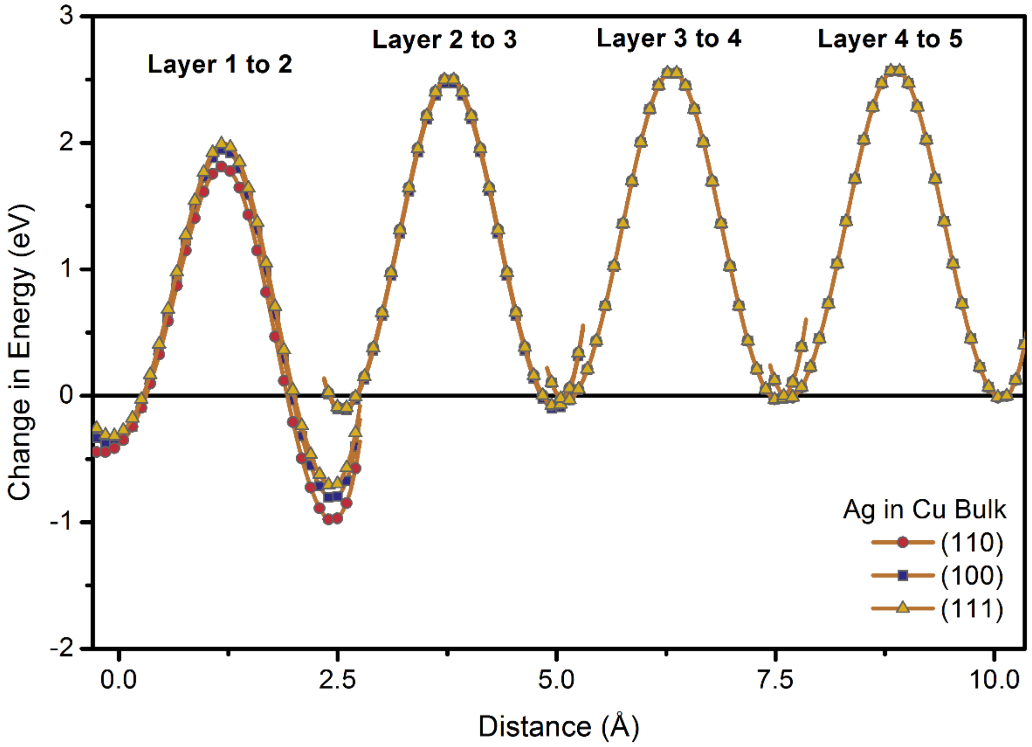


Figure 9.3 shows the change in crystal energy following the diffusion path of a migrating Ag adatom in the Cu bulk crystals.

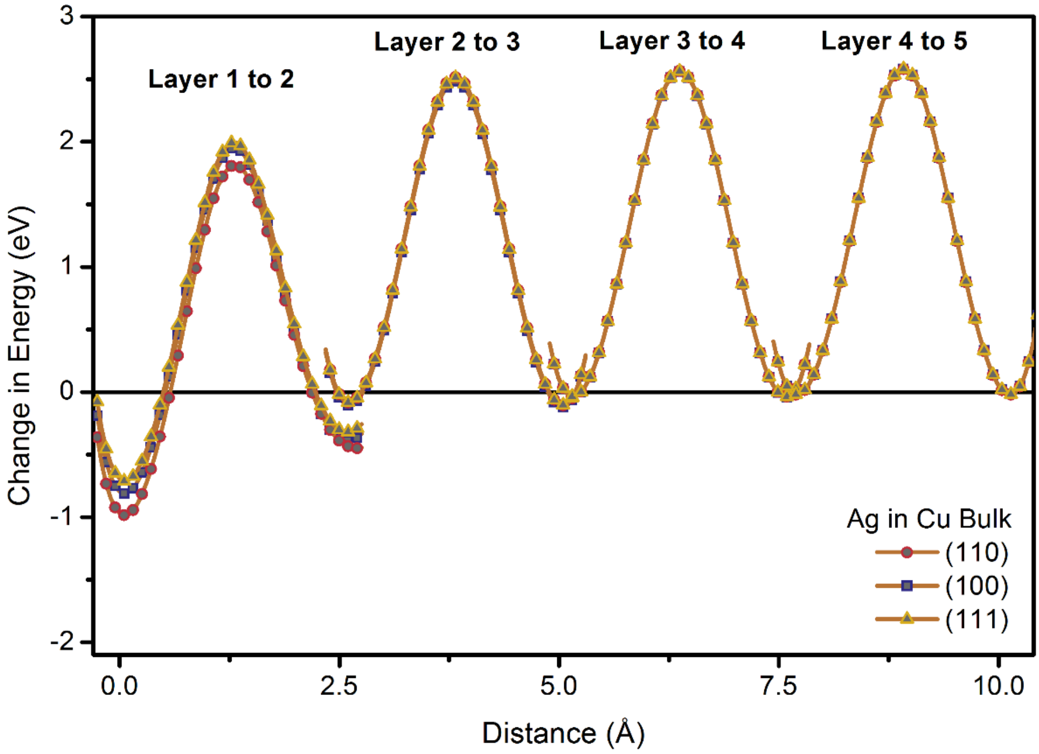


Figure 9.4 shows the segregation of a point vacancy coupled to a migrating Ag adatom in the Cu bulk crystals.

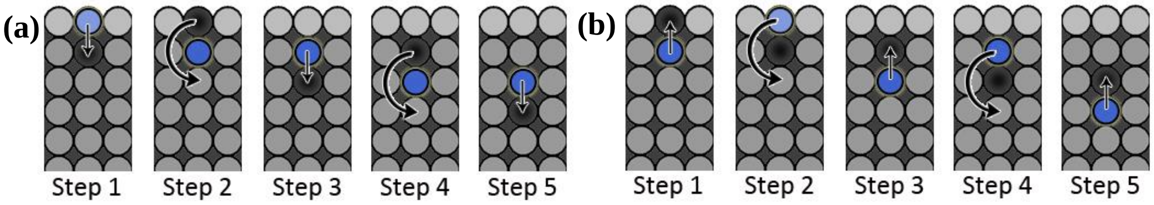


Figure 9.5 (a, b). The process is shown for determining step-by-step the energy following the diffusion path of (a) a foreign element from the surface to the bulk, or (b) a vacancy tied to a foreign element migrating from the surface to the bulk. The process in Figure 9.5 (a) is similar to Figure 8.8 (a), except that the diffusing atom is of a foreign element. As before, after the surface atom moved into the vacancy, the vacancy is exchanged with a position further down into the crystal, so that the diffusing atom can continue ever deeper. In (b), from the perspective of the vacancy, the foreign element migrated upwards into the vacant position, and the vacancy shifts downwards. However, for the migration energy measured to be that of a foreign element, the foreign element atom has to be exchanged from above the vacancy to below it in step 2. In step 3 the vacancy can then continue deeper into the crystal, while measuring the migration of a foreign element. Step 1 and 2 are once again repeated in 3 and 4, and so on.

to the surface, this also leads to discontinuities in the energy profile of the vacancy segregation path.

Even though the vacancy and foreign element are inextricably linked and the graph in Figure 9.3 cannot be used to discern the segregation energy of the foreign element alone, the slight decrease in energy of the minima turning points, on either end of the bell curves, as the atom approaches the surface gives a clear indication of segregation. It also suggests that the foreign element and vacancy both have a slight preference for segregating to the more open (110) surface, and the least preference for the closely-packed (111) surface. In addition, the amplitude of the activation energy peaks are much larger than in the pure metals (2.60 eV as opposed to 1.33 eV in pure Ag and 1.18 eV in Cu), with the larger Ag needing more energy to move through the more tightly packed Cu lattice.

***Nano-Crystals***

Figure 9.7 9.6 and Figure 9.7 show similar plots for the Cu rhombicuboctahedron with a Ag migrating atom. The surface-to-subsurface activation energy is much higher for the {100} than for the other two orientations, but once again this is due to the particle shape, as discussed before in section 8.4.2. Figure 9.6Figure 9.7 and Figure 9.7 also show a downward sloping trend in the minima turning points as the foreign atom/vacancy pair approaches the surface, hinting at segregation of the Ag atom towards the surface in Cu crystals, more so for the {111} corner and {110} edge in the rhombicuboctahedron, than the solid {100} face. One can consider the "{111}" surfaces and "{110}" surfaces in this small cuboid particle to be dominated by edge effects.

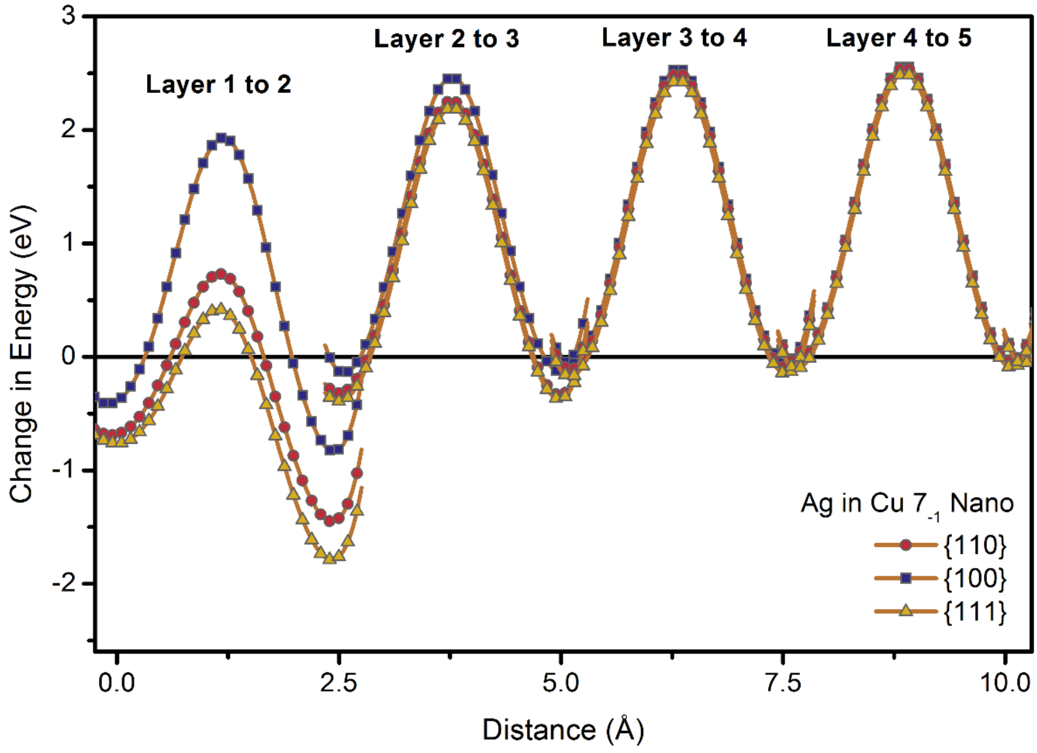


Figure 9.6 shows the change in crystal energy following the diffusion path of a migrating Ag adatom in the Cu imperfect cube.

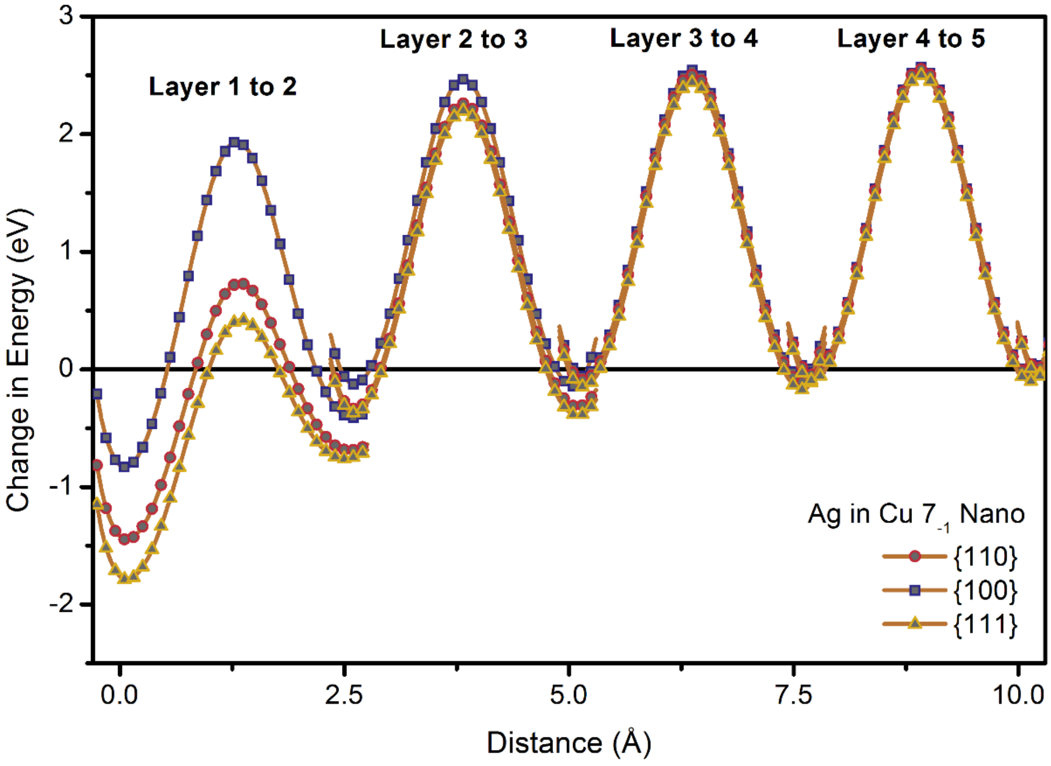


Figure 9.7 shows the change in crystal energy of the segregation of a point vacancy coupled to a migrating Ag adatom in the Cu imperfect cube.

#### 9.2.2.2 Cu Atom Migration in Ag Crystals

***Bulk Crystals***

An investigation of the energy profile measured along the diffusion path of a Cu in a Ag crystal yielded surprising results. The discontinuity caused by the presence of a point defect is much larger relative to the activation energy in the Ag crystal in the presence of a Cu atom in Figure 9.8, than it was in Figure 9.3. The steep incline of the surface-to-subsurface migration peak shows a remarkably strong drive for the vacancy to segregate to the surface, and the Cu atom to desegregate to the subsurface layer. As in Figure 9.3 and Figure 9.4, the drive to desegregate appears strongest in the (110) surface, followed by (100), and finally the (111) surface.

The lower activation energy for the (110) surface-to-subsurface migration path is because the first peak corresponds to migration of the Cu adatom from a crest position on the surface, to a trough position which is still exposed to the surface defect. The second peak is also lower, as this follows migration of the adatom from the trough to the subsurface layer. There is also a slight incline in the minima turning points approaching the surface, but increasing slightly, showing an inclination for the Cu adatom to desegregate into the bulk of the crystal away from the surface. As opposed to the Ag atom in the Cu lattice, the Cu atom can move through the Ag lattice with relative ease, shown by the smaller activation energy represented by the smaller amplitude of the peak, as the relatively small Cu atom can manoeuvre the transition in the more widely spaced Ag lattice.

***Nano-crystals***

Figure 9.10 and Figure 9.11 show the same energy profile of the Cu adatom, this time in the Ag $7_{-1}$ rhombicuboctahedron. In this case the surface segregation of the vacancy to the surface is even more aggressive, with absolutely no activation energy needed for the Cu adatom on the {111} corner to slip to the subsurface layer.

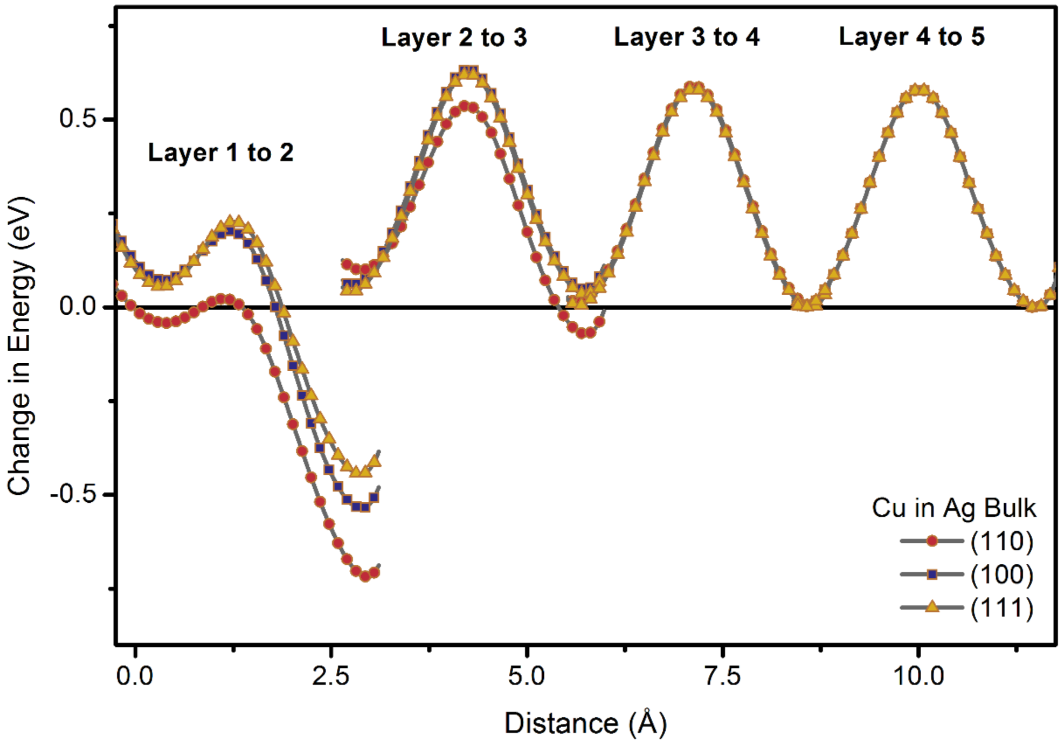


Figure 9.8 shows the change in crystal energy following the diffusion path of a migrating Cu adatom in the Ag bulk crystals.

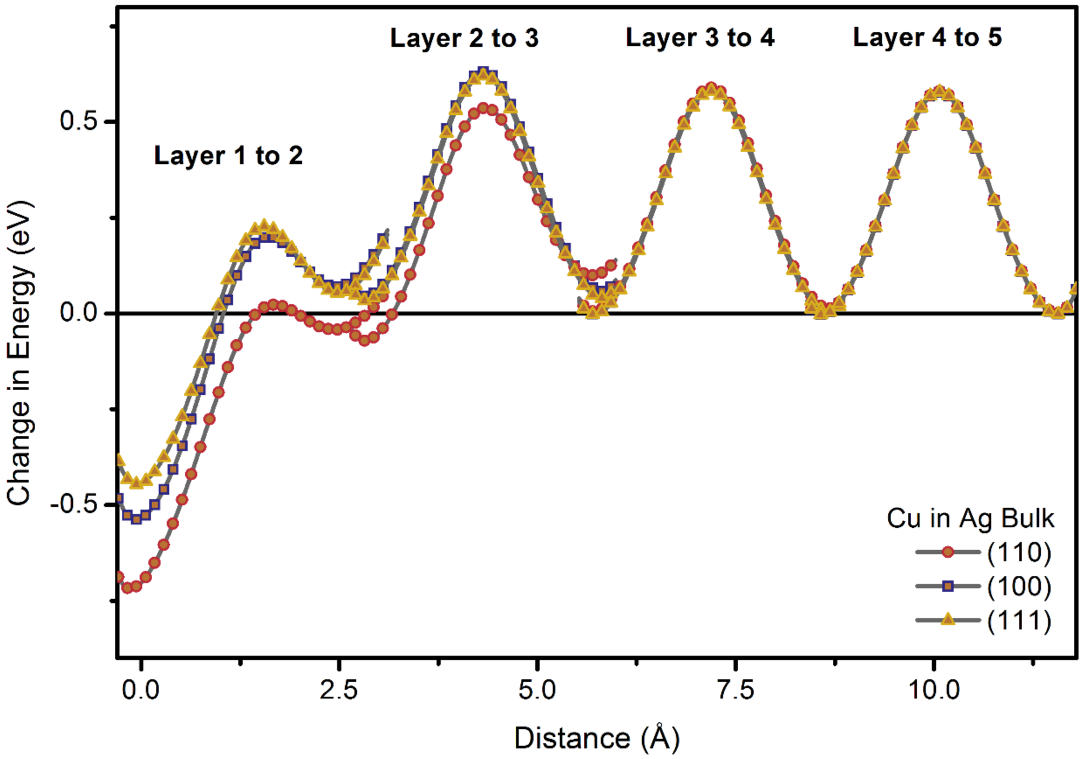


Figure 9.9 shows the change in crystal energy the segregation of a point vacancy coupled to a migrating Cu adatom in the Ag bulk crystals.

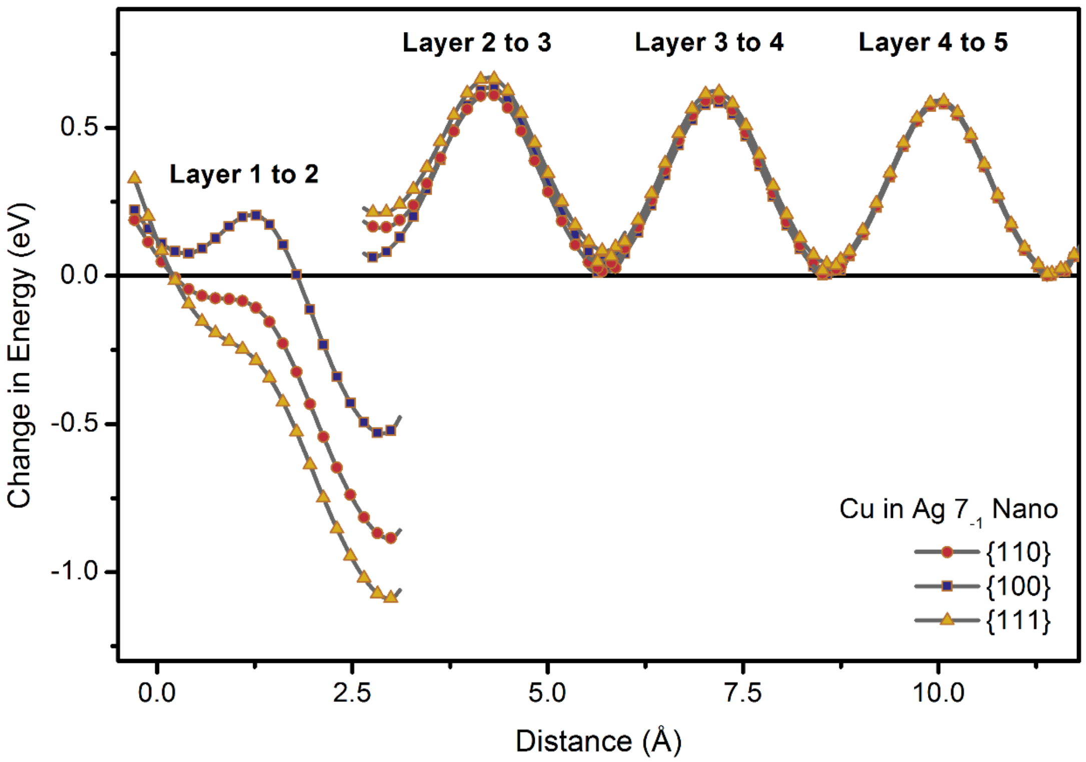


Figure 9.10 shows the change in crystal energy following the diffusion path of a migrating Cu adatom in the Ag imperfect cube.

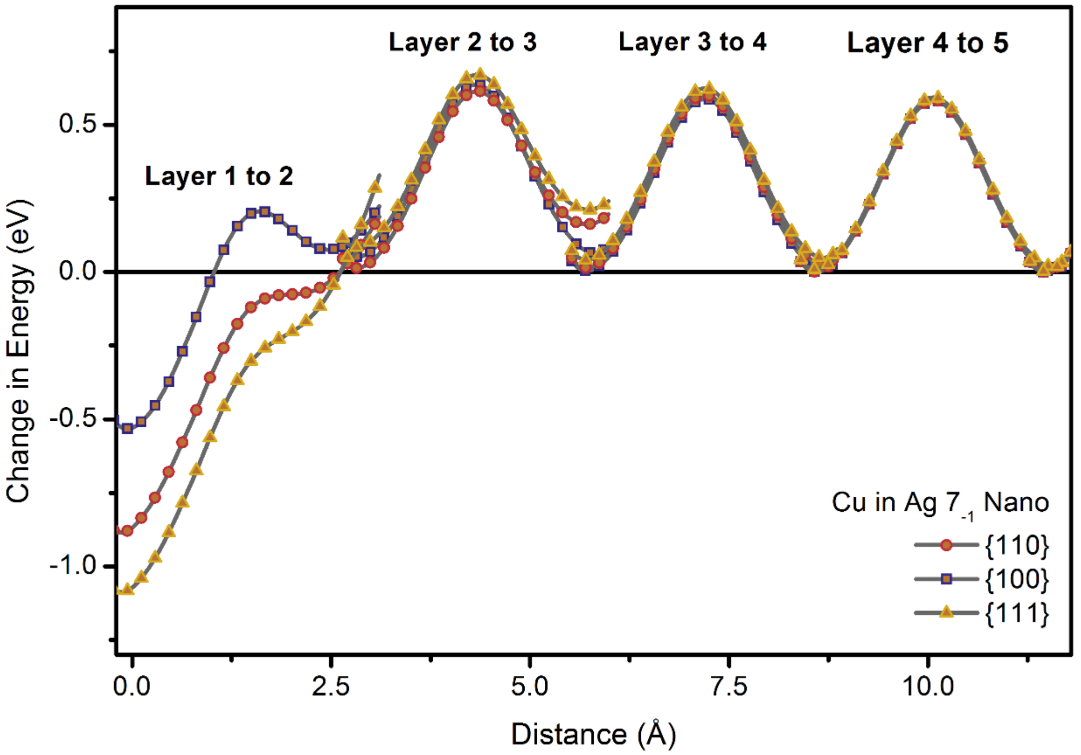


Figure 9.11 shows the change in crystal energy following the segregation of a point vacancy coupled to a migrating Cu adatom in the Ag imperfect cube.

A close second for strong vacancy segregation is the crest of the {110} surface. As in the bulk, the rhombicuboctahedron displays a slight upward slope of the minima turning points from bulk to surface, indicating that the Cu atom would preferentially desegregate into the bulk of the crystal.

### 9.2.3 Diffusion Energies

From the migration energy depth profiles a strong indication was observed for segregation of Ag and Cu: Ag to the surface, Cu to the bulk. As a final indicator, the difference in free energy, $\Delta G$, was calculated. The free energy represents the differences in the total potential energy of the crystal, and the change in free energy when a foreign atom is moved from within the crystal to its surface is an indication of whether that element is driven to segregate or desegregate. A negative $\Delta G$ value indicates segregation while a positive value points to a drive to desegregate. The values of $E_{extr}$, $E_{surf}$, $E_v$, $E_m$, $Q$ and $\Delta G$ are all summarized in Table 9.1. The activation energy $Q$ was compared with values obtained from radiotracer data [1], and the segregation energy $\Delta G$ compared with Molecular Dynamics calculations of surface dependent values from reference [2].

In Figure 9.12 is a summary of a $E_v$, $E_m$ and $Q$ for a single Ag atom in Cu bulk and nanocrystals, and in Figure 9.13 the same data for a Cu atom in Ag bulk and nanocrystals. Referring to Figure 8.13 and Figure 8.14, the migration energy of a Ag atom in Cu is much higher than for a Ag atom in a Ag crystal, or a Cu atom in a Cu crystal, which increases the diffusion activation energy $Q$ significantly as well. The Ag atom has greater difficulty migrating through the more tightly packed Cu lattice, which explains the increase in $E_m$. It is interesting to note that although this is a Cu crystal, if it is allowed to relax around the Ag atom, the adatom's associated $E_v$ energies change to a significant degree, as was seen in pure Ag crystals. This is as a result of the large change in $E_{extr}$ as seen in Figure 9.1.

In bulk this affects the activation energy $Q$. However, in the nanocrystal, the relaxation changes the $E_v$ and $E_m$ values proportionately so that $Q$ remains relatively unchanged. It may be that the smaller crystal deforms more thoroughly, which influences $E_m$ in the

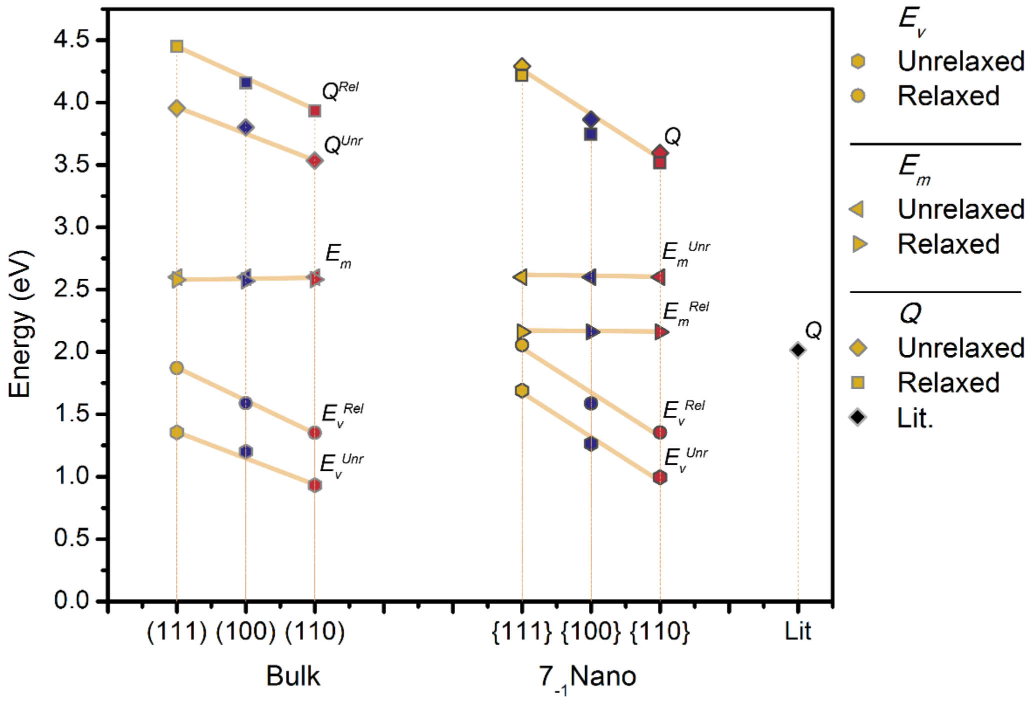


Figure 9.12. The vacancy formation energy, migration energy in the bulk and diffusion activation energy of the Ag adatom for the Cu bulk and nanocrystals. Literature values from reference [1].

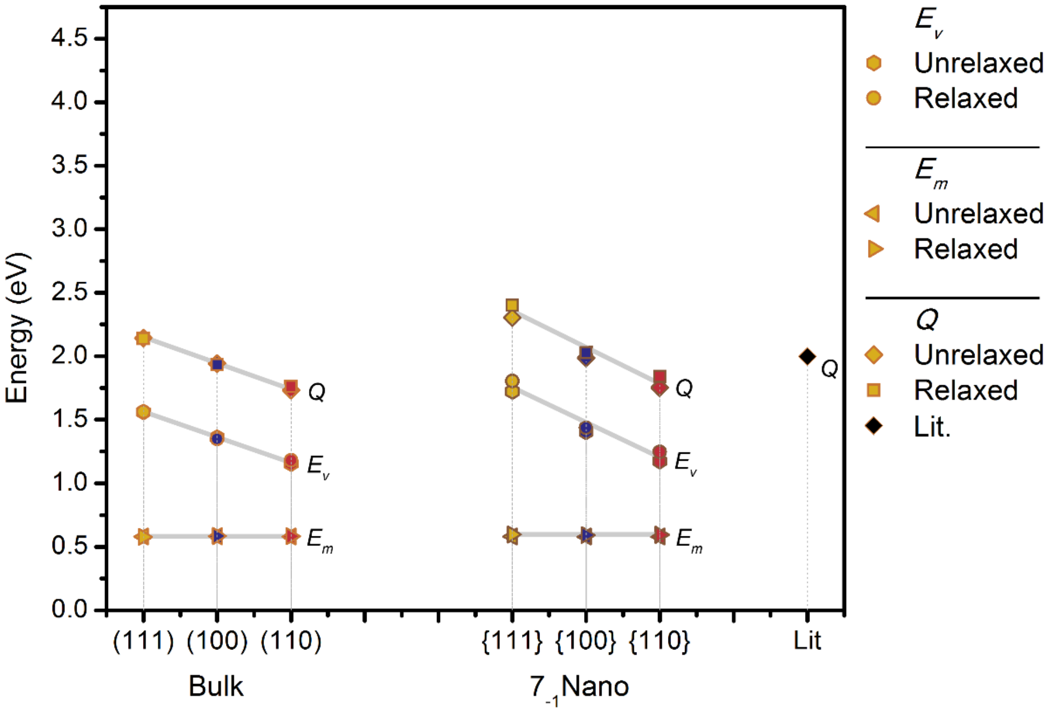


Figure 9.13. The vacancy formation energy, migration energy in the bulk and diffusion activation energy of the Cu adatom in the Ag crystals. Literature values from reference [1].

Table 9.1 A summary of the diffusion and segregation energy values of the mixtures of Ag and Cu crystals. All energies are given per atom.

| | | $E_{extr}$ (eV) | | $E_{surf}$ (eV) | | $E_v$ (eV) | | $E_m$ (eV) | | $Q$ (eV) | | $\Delta G$ (eV) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Unrelaxed** | **Relaxed** | **Lit.** | **Unrelaxed** | **Relaxed** | |
| **Ag atom in Cu crystal** | Bulk (111) | -4.12 | -4.63 | -2.76 | -2.76 | 1.36 | 1.87 | 2.60 | 2.58 | 3.96 | 4.45 | -0.25 [2] | -0.25 | -0.21 | |
| | Bulk (100) | -4.12 | -4.50 | -2.92 | -2.92 | 1.20 | 1.59 | 2.60 | 2.57 | 3.80 | 4.16 | -0.30 [2] | -0.31 | -0.27 | |
| | Bulk (110) | -4.12 | -4.52 | -3.18 | -3.17 | 0.93 | 1.35 | 2.60 | 2.58 | 3.53 | 3.93 | -0.31 [2] | -0.35 | -0.28 | (110) crest |
| | | | | | | | | | | | | | -0.06 | -0.12 | (110) trough |
| | Lit. | | | | | | | | | 2.02 [1] | | | | | |
| **Ag atom in Cu crystal** | Nano7-1 {111} | -4.12 | -4.41 | -2.43 | -2.36 | 1.69 | 2.06 | 2.60 | 2.16 | 4.29 | 4.22 | | -0.53 | -0.35 | |
| | Nano7-1 {100} | -4.12 | -4.41 | -2.85 | -2.83 | 1.26 | 1.59 | 2.60 | 2.16 | 3.87 | 3.75 | | -0.33 | -0.30 | |
| | Nano7-1 {110} | -4.12 | -4.41 | -3.12 | -3.06 | 1.00 | 1.36 | 2.60 | 2.16 | 3.60 | 3.52 | | -0.45 | -0.34 | {110} crest |
| | | | | | | | | | | | | | -0.26 | -0.17 | {110} trough |
| **Cu atom in Ag crystal** | Bulk (111) | -3.98 | -3.98 | -2.42 | -2.41 | 1.56 | 1.56 | 0.58 | 0.58 | 2.14 | 2.14 | | 0.09 | 0.06 | |
| | Bulk (100) | -3.98 | -3.96 | -2.62 | -2.61 | 1.36 | 1.35 | 0.58 | 0.58 | 1.94 | 1.93 | | 0.10 | 0.07 | |
| | Bulk (110) | -3.98 | -3.97 | -2.82 | -2.79 | 1.15 | 1.18 | 0.58 | 0.58 | 1.73 | 1.76 | | 0.09 | 0.05 | (110) crest |
| | | | | | | | | | | | | | 0.09 | 0.08 | (110) trough |
| | Lit. | | | | | | | | | 2.00 [1] | | | | | |
| **Cu atom in Ag crystal** | Nano7-1 {111} | -3.98 | -3.98 | -2.25 | -2.17 | 1.72 | 1.81 | 0.58 | 0.60 | 2.30 | 2.40 | | 0.15 | 0.05 | |
| | Nano7-1 {100} | -3.98 | -3.98 | -2.57 | -2.54 | 1.40 | 1.44 | 0.58 | 0.59 | 1.99 | 2.03 | | 0.11 | 0.08 | |
| | Nano7-1 {110} | -3.98 | -3.98 | -2.80 | -2.73 | 1.17 | 1.25 | 0.58 | 0.59 | 1.75 | 1.84 | | 0.12 | 0.06 | {110} crest |
| | | | | | | | | | | | | | 0.16 | 0.12 | {110} trough |

nanocrystal as well. The literature value of $Q$ of 2.02 eV/atom for Ag in Cu [1] does not compare well to the calculated values, which begin at 3.53 eV/atom, possibly related to the combination of a bad fit of QSC parameters to calculating $Q$, particularly $E_m$, as $E_v$ values correspond well. As the vacancies are not allowed to relax during migration to prevent the filling of open vacancies before the migration atom can migrate there, this freezing of the vacancy can artificially increase the migration energy as the nearest neighbours cannot move out of the way of the migrating atom. During Molecular Dynamics simulations where atoms relax and can move freely, this problem will not be a factor. An additional factor may be a bad fit of QSC parameters for modelling Ag.

In contrast to Figure 9.12, Figure 9.13 displays a low $E_m$ value for the Cu atom migrating through the Ag lattice, corresponding to the low peak amplitudes observed in Figure 9.10 and Figure 9.11. Thus, it results in a lower $Q$ value. Even though the Cu atom is embedded in a Ag crystal, the adatom's associated energies do not change significantly when it is relaxed, and thus there is not a great change in the $Q$ values from unrelaxed to relaxed crystals. The literature value also compares far better, as the QSC parameters are better able to calculate energies for Cu, and a lower $E_m$ value corresponds better to literature.

# 9.3 Segregation Energies

## 9.3.1 Segregation of Ag Atoms in Cu Crystals

***Bulk Crystals***

Having investigated fully all the diffusion activation energies, this section investigates the segregation energy of the foreign elements directly. By noting the difference in total free energy $\Delta G$, when the position of the foreign element in a surface is exchanged with that of a nearest neighbour deeper inside the crystal, a prediction can be made of the tendency of that element to segregate to the surface or the bulk. In Figure 9.14 and Figure 9.15 the left-most points (1 on the x-axis) correspond to the surface layer, the next point corresponds to the next nearest neighbour one step deeper, and so on.

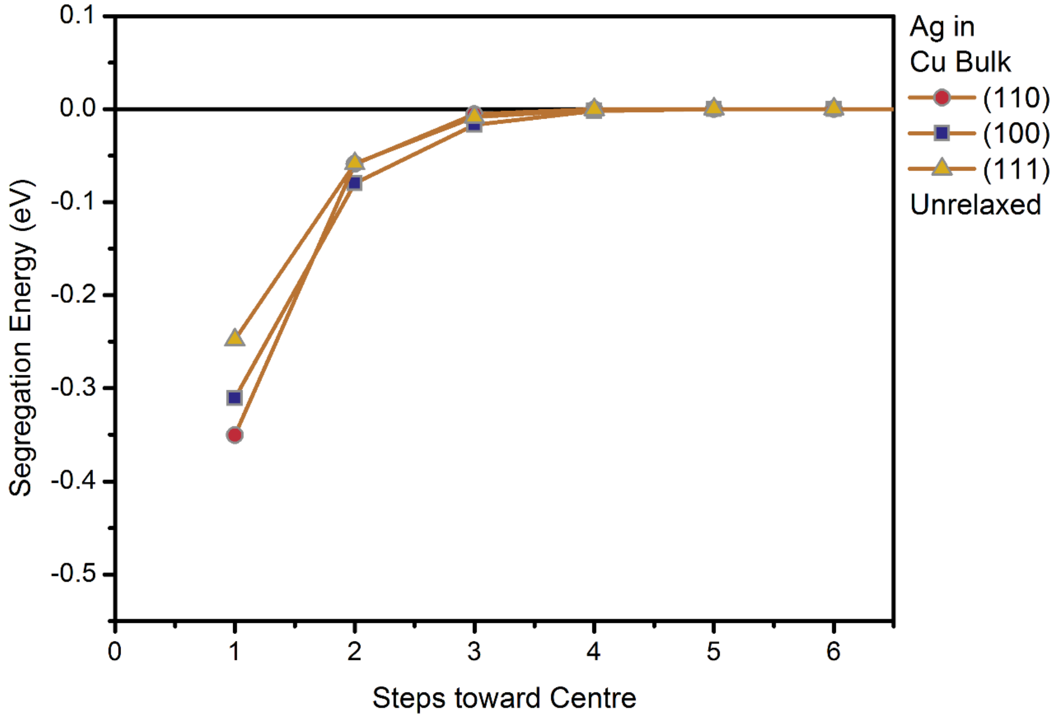


Figure 9.14. The segregation energy of the Ag adatom at each successive layer and into the bulk in unrelaxed Cu bulk crystals for each of the different surface orientations.

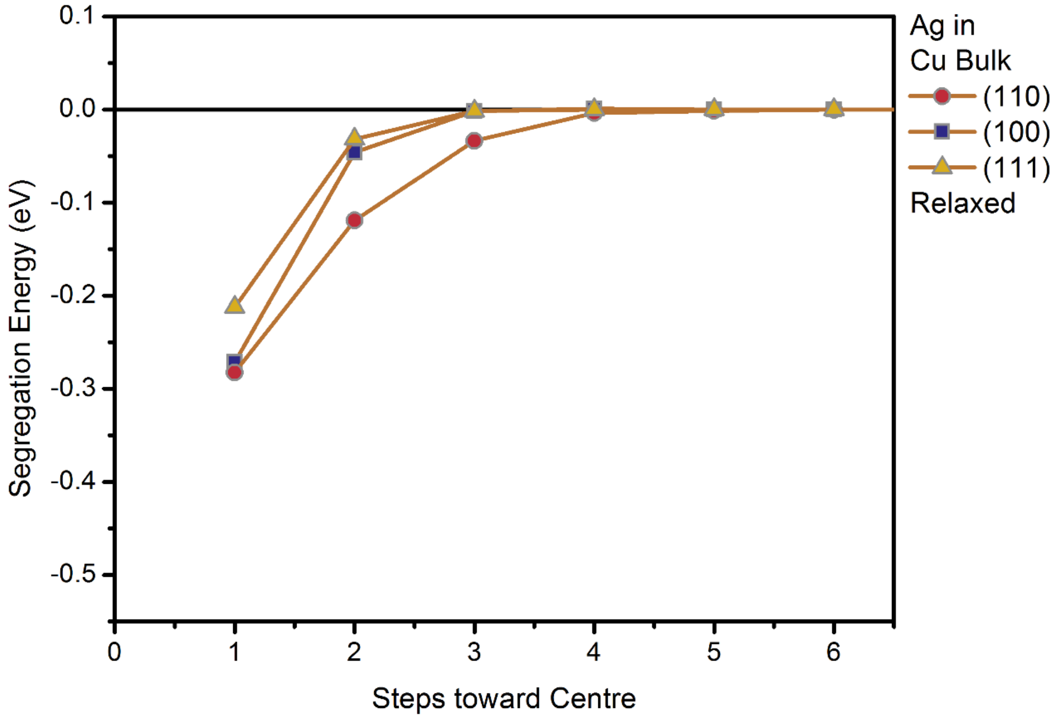


Figure 9.15. The segregation energy of the Ag adatom at each successive layer and into the bulk in relaxed Cu bulk crystals for each of the different surface orientations.

The change in free energy is determined by using the crystal state with the foreign element in the bulk as the default energy and finding the energy change when the foreign element is exchanged with another atom in a layer closer to the surface. Thus, the energy change in the surface layer is plotted at one, the change to the sub-surface layer is plotted to 2, etc. A negative energy indicates a drive to segregate out to the surface, as the overall energy of the crystal is more negative when the foreign element is on the surface, indicating a more strongly bonded, stable crystal.

In Figure 9.14 these changes in free energy were plotted for a Ag atom in Cu bulk crystals for different surface orientations. Thus far, measurements done on the surface have yielded different readings, with $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$, and the energy profiles from the migration paths of atoms to the surface showed a similar trend in bulk where edge effects did not interfere.
On the surface, the number of first nearest neighbours an atom has, has a significant influence on its energy. If a Ag atom is placed in the lattice position of a Cu atom, it will be closer to its nearest neighbours than is optimal for a Ag atom, and it disturbs first nearest neighbours significantly, which in turn influences the energy of the Ag atom. In the (110) surface an atom has seven nearest neighbours, in the (100) surface it has eight, and in the (111) surface it has nine. If it is assumed that the change in free energy is greatest when the foreign element goes from the full count of twelve nearest neighbours to the lowest number, a similar $\Delta G^{(111)} < \Delta G^{(100)} < \Delta G^{(110)}$ trend in the magnitude of the energy change in surface measurements would be expected. In the unrelaxed crystals, these expectations are borne out. As the Ag atom is stepped ever deeper into the unrelaxed Cu crystal, $\Delta G$ becomes less negative, showing a lesser drive to segregate to that layer. By the third layer, $\Delta G$ is almost negligibly small. Thus, in unrelaxed bulk Cu crystals, Ag will preferentially segregate out to the (110) surface, then the (100) surface, and least prefers to segregate to the tightly packed (111) surface.

The picture changes if, after the foreign atom is exchanged into each layer in the unrelaxed Cu crystal, it is allowed to relax for 4 000 steps. The crystal can adjust to the new position of the foreign element, to accommodate it and find a new arrangement which leads to an overall lower crystal energy. The resulting magnitude of the change in free energy shown in Figure 9.15 is slightly less, but the surface orientation dependence of $\Delta G$ changes. As the Cu atoms are allowed to relax, the first nearest neighbours are no longer uncomfortably close, which resulted in a disproportionate contribution to the bond interactions with the foreign atom in the unrelaxed crystal. As a result, the effect becomes less localized, and the second, third, and so on, nearest neighbours also contribute a larger proportion to the binding energies of the Ag adatom. With a larger volume of atoms interacting, the number of first nearest neighbours has a lesser effect, so the surface orientation dependence disappears, as can be seen in the (100) and (110) surface energies, where the difference between them become negligible. However, as the (111) surface is more tightly packed, it cannot relax as much, leading to a slight drop in segregation energy.

### *Nano-crystals*

The picture looks different in the unrelaxed $7_{-1}$ Cu rhombicuboctahedron in Figure 9.16, particularly with all the edge effects distorting the typical energies. In the "{111}" corner position, the Ag atom has a mere 5 nearest neighbours, atypical of bulk {111} surface atoms (refer to Figure 8.12 (a)). For this corner atom, the drive to segregate is even stronger. This indicates strongly that for the large Ag atom in an unrelaxed crystal, the drive for surface segregation is strongest towards surface positions which minimize the number of first nearest neighbours. In the case of the nanocrystal, this is the "{111}" corners with very few nearest neighbours, and the "{110}" edges with a few more nearest neighbours. The troughs of the {110} edges also have a low vacancy formation energy which further improves the chance of segregation to those sites.

This relationship of nearest neighbours and segregation energies once again becomes less apparent if the crystal can relax, at 1 on the x-axis in Figure 9.17. In this case the edge energies, with a more limited number of wider nearest neighbours, and more space for first nearest neighbours to relax, resemble each other, while the energy in the {100} face of the rhombicuboctahedron is slightly lower. The step-like character of the "{111}" atom is as a result of stepping into the crystal along a diagonal path, thus remaining closer to the surface than the {100} which steps into the bulk perpendicularly.

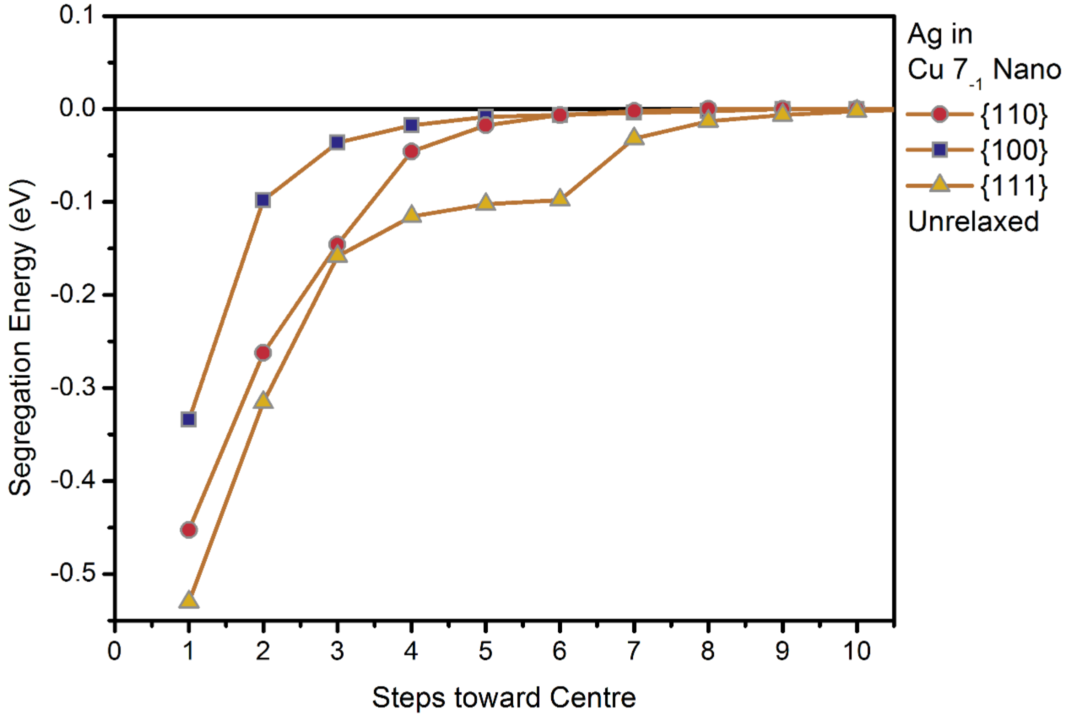


Figure 9.16. The segregation energy of the Ag adatom at each successive layer of the different surface orientations in the unrelaxed Cu imperfect cube.

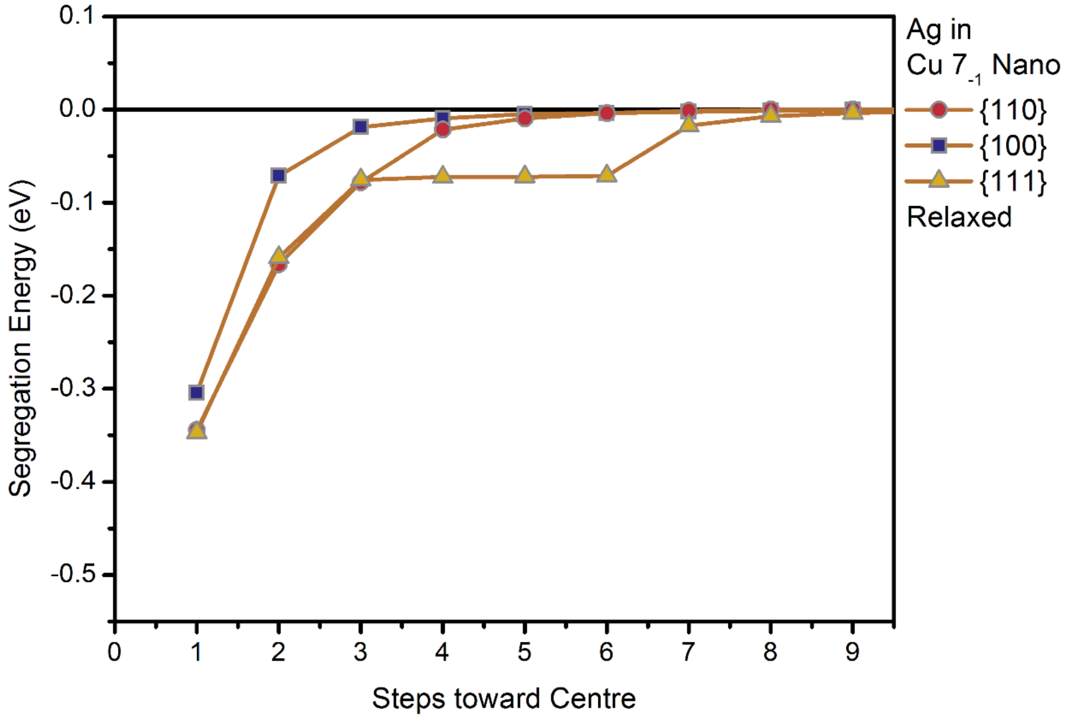


Figure 9.17. The segregation energy of the Ag adatom at each successive layer of the different surface orientations in the relaxed Cu imperfect cube.

## 9.3.2 Segregation of Cu Atoms in Ag Crystals

***Bulk Crystals***

In Figure 9.18 are the segregation energies of a Cu atom from the surface into the unrelaxed bulk of Ag bulk crystals. It is immediately apparent that the expected relationship of $\Delta G^{(111)} < \Delta G^{(100)} < \Delta G^{(110)}$ to first nearest neighbours is not present. Instead, as the Cu atom is smaller and thus further removed from the Ag nearest neighbours than it would be in a Cu crystal, that interaction is weakened. Rather, it is the sum of the first, second, third, and so on, nearest neighbours, overall, that influences the relationships between the surface segregation energies of the different orientations. As the effect is spread out over the wider volume, the small differences in number of first nearest neighbours become less influential, and the differences in surface segregation energy of the different orientations are negligible, including the trough of the (110) surface, shown as the second point from the surface on the (110) series. Overall the segregation energy is positive, suggesting that the Cu atom prefers to desegregate into the bulk of the crystal. The drive is not as strong, however, as it was for Ag to segregate out of a Cu crystal. The segregation energy is a mere 0.1 eV per atom. This positive segregation energy decreases rapidly by the third layer as the effect of the surface defect decreases on the segregating atom. By the fourth layer, there are no detectable influences from the free surface.

In Figure 9.19 the segregation energy relationships are not much changed other than a small decrease in magnitude. There is also a slight decrease in the segregation energy of the crest of the (110) surface. As in the previous case, allowing the Ag crystals to relax changes the bonds between the Cu adatom and its first nearest neighbours. In this case, the (110) surface on the crest position with the fewest first nearest neighbours can relax more closely around the Cu adatom without disturbing the crystal too greatly, allowing for a slight decrease in segregation energy to that position. However, the Cu atom would still desegregate into the bulk of the crystal from a (110) crest position. The (110) trough position, on the 2-position on the x-axis in Figure 9.19, has the highest segregation energy of 0.08 eV for a single atom. This is because a Cu atom in the trough position have (110) crest atoms as nearest neighbours, which are greatly disturbed when the crystal relaxes. As for the (111) and the (100) surfaces, the differences in segregation energies or a single atom are negligible, 0.06 eV and 0.07 eV respectively.

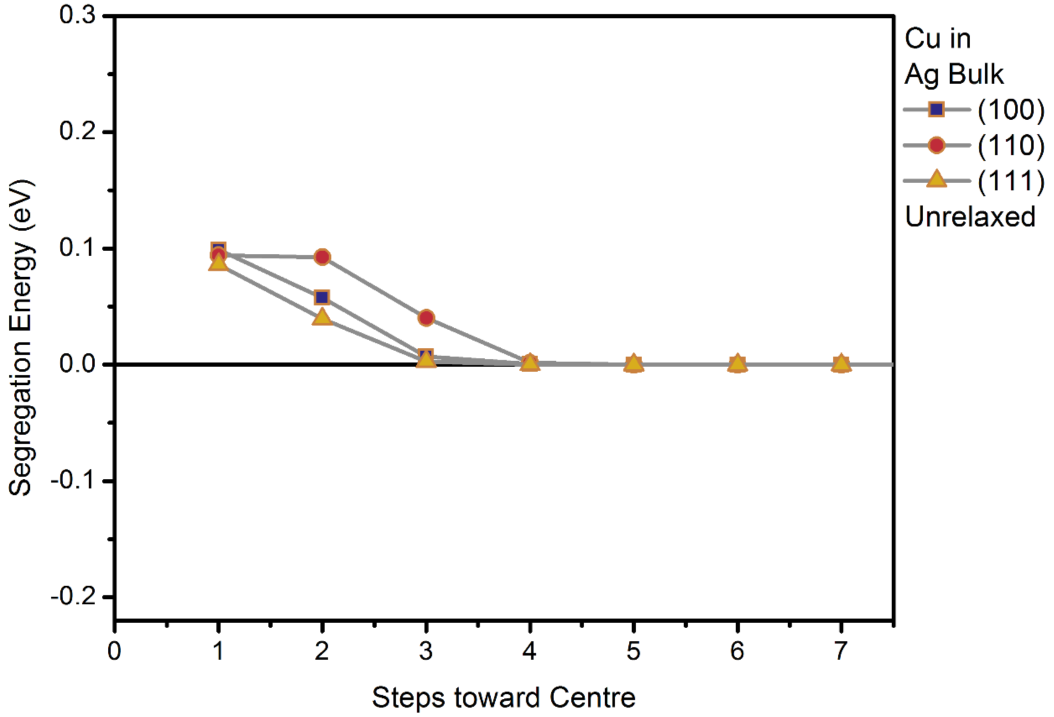


Figure 9.18. The segregation energy of the Cu adatom at each successive layer of the different surface orientations in unrelaxed Ag bulk crystals.

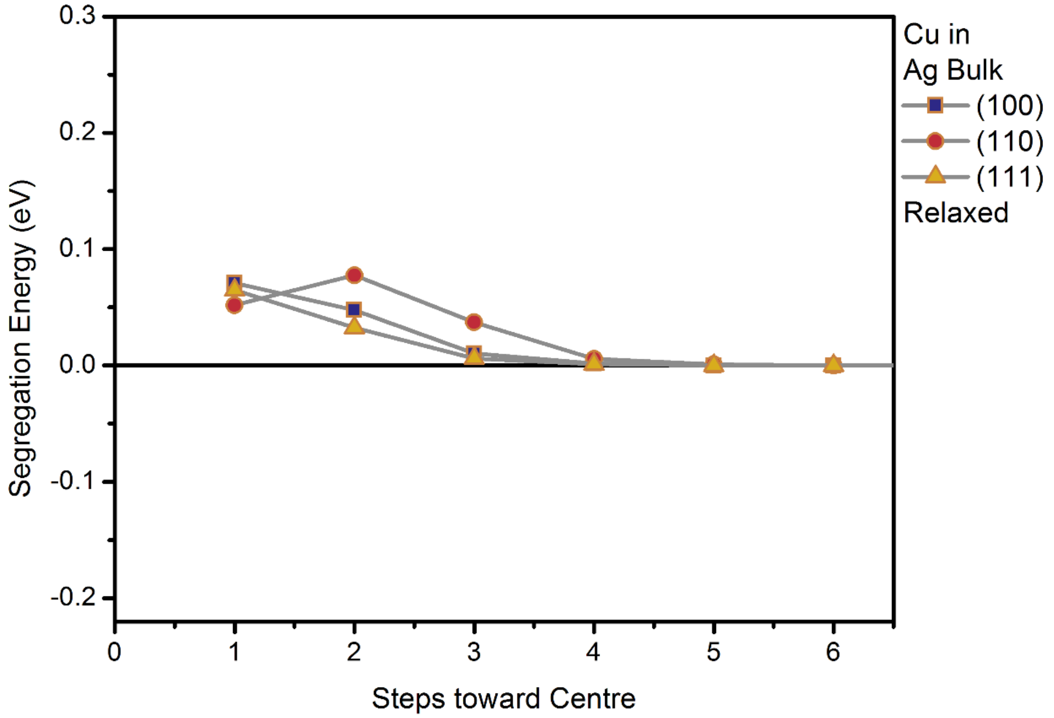


Figure 9.19 The segregation energy of the Cu adatom at each successive layer of the different surface orientations in relaxed Ag bulk crystals.

***Nano-crystals***

Figure 9.20 shows the segregation energy of a Cu atom in the unrelaxed Ag $7_{-1}$ rhombicuboctahedron. The differences in segregation energy seems to correspond with the number of first nearest neighbours, from five for the corner “{111}” position, to seven for the edge “{110}” position, to eight in the {100} face. However, as the “{111}” atom is on a corner, it would have the fewest second, third, fourth and so on, nearest neighbours as well. The {110} atom occurs on the edge of the crystal and has more extended nearest neighbours than the corner atom, but fewer than the {100} atom embedded in the middle of a face. The {110} trough segregation energy is slightly higher than the crest value, while the sub-surface energy of the “{111}” surface has the highest segregation energy. In both these cases, the Cu atom in a subsurface atom position has weak bonds with surface atoms that already have fewer nearest neighbours, significantly undermining the stability of the crystal.

Thus, these subsurface positions on the edges have the highest positive segregation energies, indicating the Cu atom least prefers to occupy subsurface positions in the corner and edge of the Ag $7_{-1}$ rhombicuboctahedron. The relaxed segregation energy shown in Figure 9.21 has shown a marked change, where the relationship has completely swung around, with the corner atom having the lowest surface segregation energy, followed by the {110} crest atom, with the {100} face atom having the largest segregation energy. The sub-surface segregation energy of the “{111}” corner atom is much higher than the other segregation energies. With the crystal allowed to relax, the Cu adatom can bind more closely with its first nearest neighbours when it is on an edge surface position. However, when it is in the subsurface position of those same edges, it significantly disturbs weakly bonded surface atoms around it, which leads to an increased segregation energy.

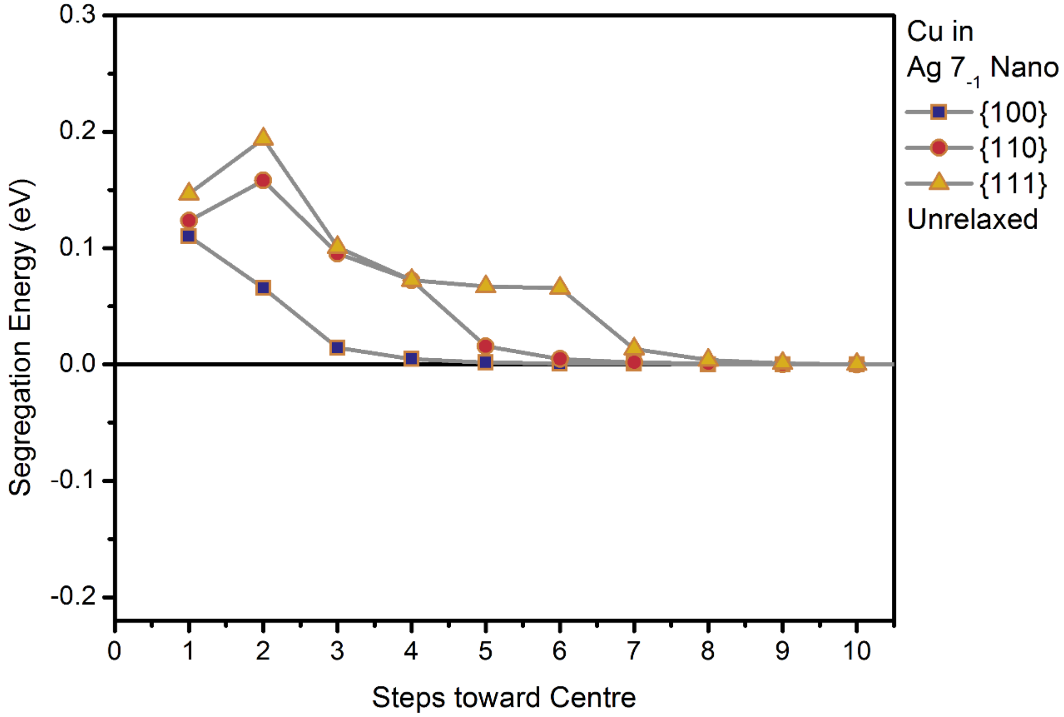


Figure 9.20. The segregation energy of the Cu adatom at each successive layer of the different surface orientations in the unrelaxed Ag imperfect cube.

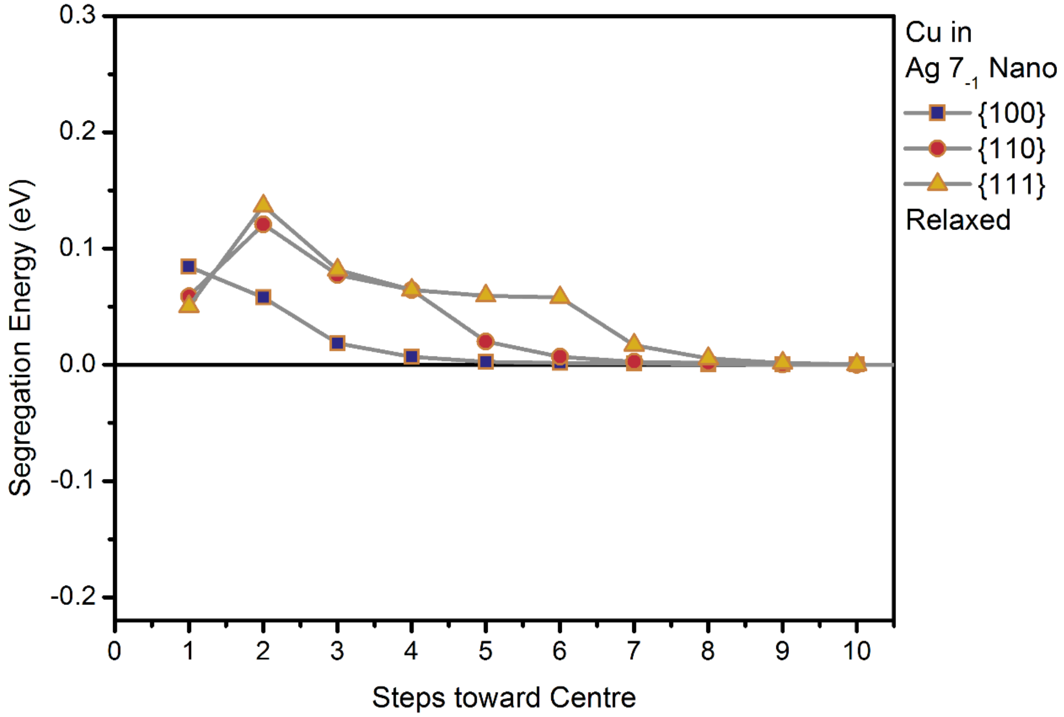


Figure 9.21. The segregation energy of the Cu adatom at each successive layer of the different surface orientations in the relaxed Ag imperfect cube.

### 9.3.3 Summary of Segregation Energies

Figure 9.22 and Figure 9.23 show a summary of all the segregation energies calculated for the different surfaces. In Figure 9.22 the bulk segregation energies are also compared to a previous Molecular Dynamics calculation of segregation energies of Ag in Cu crystals [2]. For better comparison, the values were converted to eV/atom and the use of negative and positive energies changed to the convention of negative energies driving surface segregation, whereas positive energies driving bulk desegregation. The literature values and the energies calculated in this study agree well. Though some surface orientation dependence is evident, there is only a slight difference in the (111) bulk surface for a Ag atom in a Cu crystal, where the large Ag atom distorts the tightly packed surface layer. The {110} crest for the Cu in a Ag crystal has a slightly lower desegregation energy as it allows the Cu atom to relax without disturbing the crystal structure too much. The greatest influence on segregation energy seems to be the shape of the crystal, as evidenced by the greater effect of edges on the segregation energies in the $7_{-1}$ rhombicuboctahedrons. This effect is much more magnified in the unrelaxed crystal, but may have a non-negligent impact on the preference for surface segregation to the corners and edges of nanocrystals.

### 9.3.4 Relaxation Simulation of Segregation in Ag/Cu Nano-crystal

Having confirmed the segregation energy is likely to drive segregation of Ag out onto the surface, and Cu into the bulk, a small $7_{-1}$ imperfect cube of 50 % Cu and 50 % Ag, mixed randomly, was simulated for 500 000 steps at 650 K to ensure that atom transport could take place in a relatively short time. Figure 9.24 (a-d) shows the heated nanoparticle as it loses its cubic shape. Figure 9.24 (d) and (e) appear to have mostly Ag on the surface, with Cu in the sub-surface layer.

A cross-section of the final step of the relaxation run in Figure 9.25 shows clear separation taking place between the Cu and Ag atoms, with the Ag atoms forming a thin shell on the outside of the particle. It is thus conclusively shown that the algorithm as modified can simulate segregation in Ag and Cu, with Ag atoms migrating to the surface.

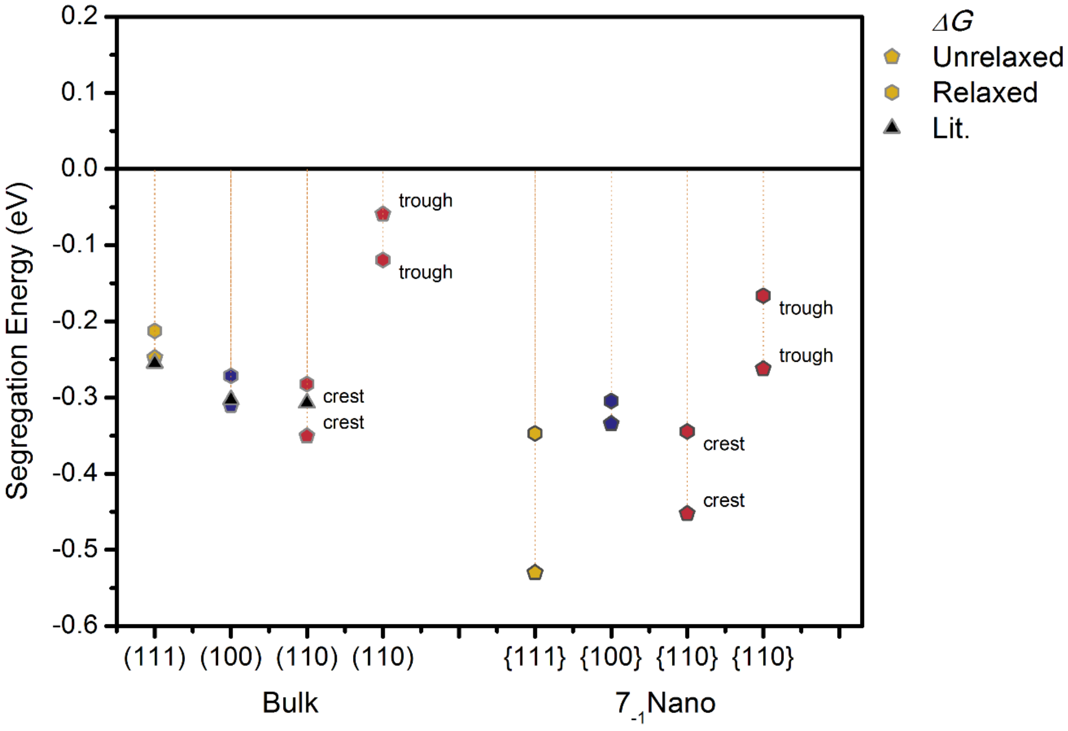


Figure 9.22. The segregation energy of the Ag adatom on the different surface orientations of the Cu bulk and nanocrystals. Literature values are from reference [2], though the convention for negative and positive energies are reversed for better comparison.

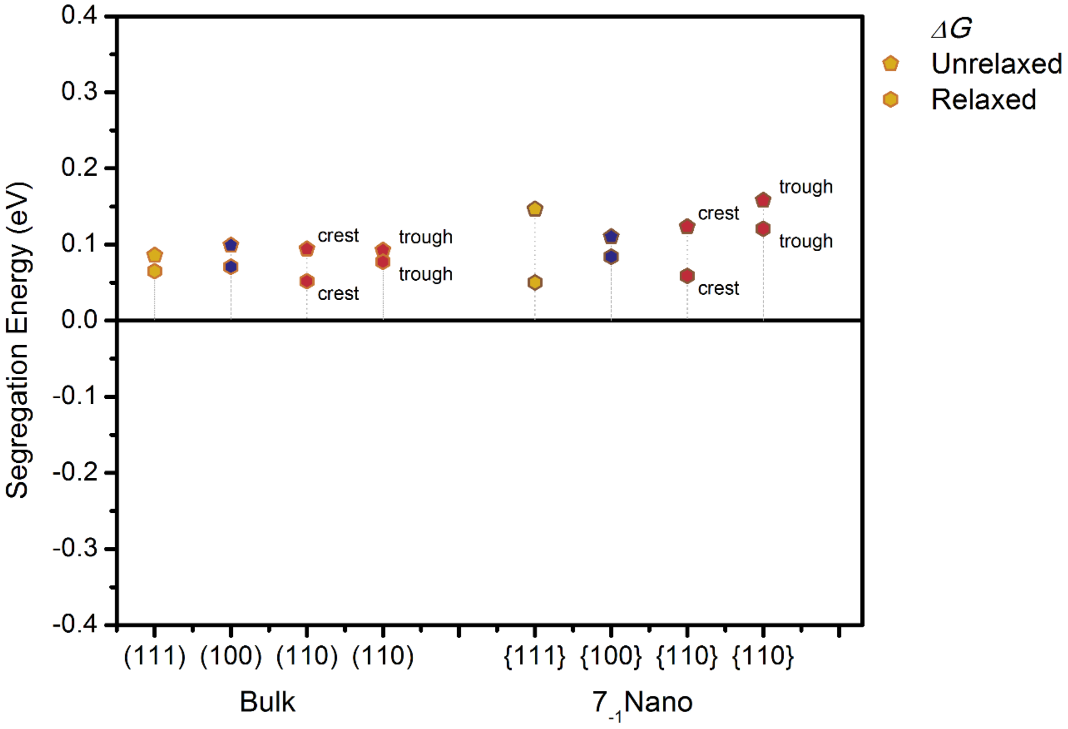


Figure 9.23. The segregation energy of the Cu adatom on the different surface orientations of the Ag bulk and nanocrystals.

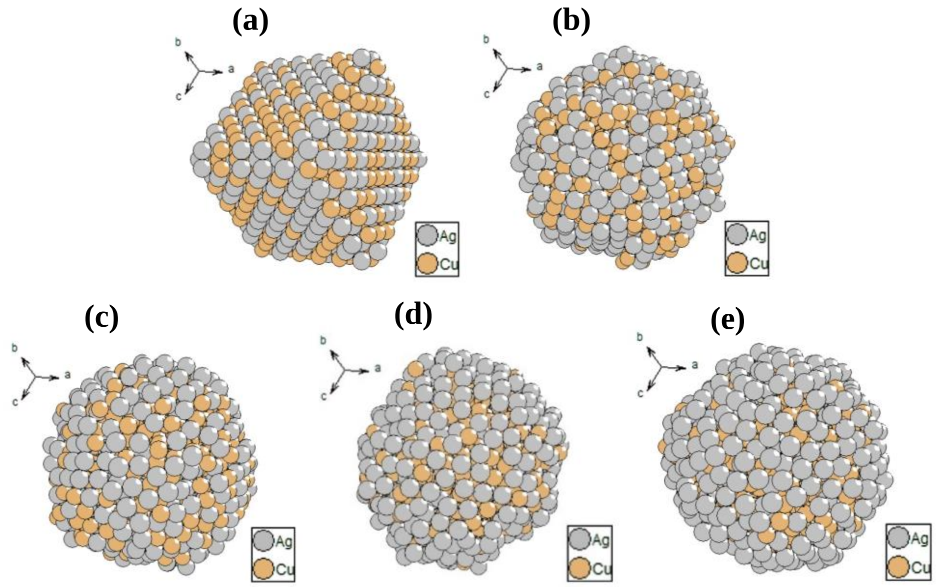


Figure 9.24 (a-e). A $7_{-1}$ imperfect cube of 50/50 Cu/Ag was left to run for 500 000 steps at 650 K. (a) through (e) are respectively 0 steps, 50 000 steps, 100 000 steps, 250 000 steps, and 500 000 steps.

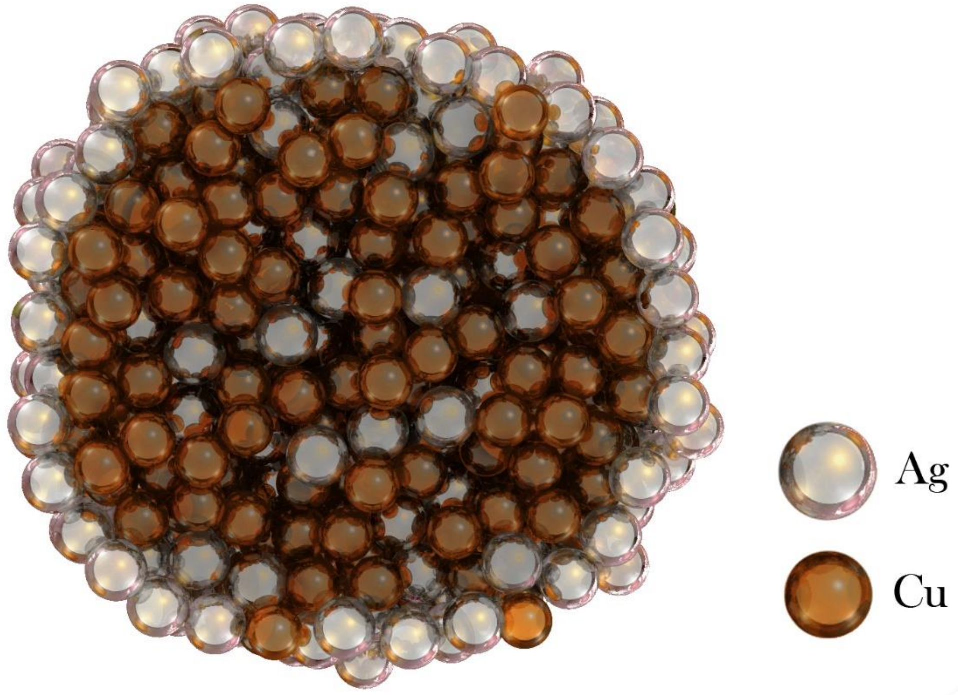


Figure 9.25. A cross-section through the nanoparticle at 500 000 steps. The central atoms are darkened slightly to help distinguish between bulk and surface atoms. The surface atoms are visibly enriched with Ag.

It would be interesting to see if future investigations of runs at lower temperatures for longer times would allow the cubic nanoparticle to retain its shape, and whether the face, edge and corners would segregate at different rates, as predicted by the energies in Figure 9.22.

## 9.4 Summary

Surface segregation of Ag in Cu was successfully simulated and the calculated segregation values of a Ag atom in Cu compared well to literature values.

The surface-orientation dependent relationships with diffusion energies were seen in the mixed metals. The migration energy profiles were interesting, as the minima showed a decreasing trend to the surface in the case of a Ag atom in a Cu crystal, and an increasing trend to the surface in the case of a Cu in a Ag crystal. These trends hinted that Ag would want to segregate out of a Cu crystal and Cu would want to desegregate into a Ag crystal. The migration energy involved a vacancy-adatom pair, and so to extract the pure segregation energy without the influence of a point defect, the change in free energy was measured from exchanging the position of the foreign element with atoms at different depths in the perfect crystal. The segregation energy of Ag in a Cu crystal was negative on the surface, indicating Ag wanted to segregate to the surface, while the Cu atom in Ag had a positive segregation energy, indicating it wanted to desegregate into the bulk of the Ag crystal. The relationship between the segregation energy and the surface-orientation was not quite as simple as the other diffusion-related energies, as the effect is not as local and not only first nearest neighbours, but also second, third etc. nearest neighbours have an influence on the segregation energy. As such, the differences in segregation energies were far more pronounced in the rhombicuboctahedron, where the surface atom's position on the corner, edge, or a face of the rhombicuboctahedron more significantly influenced the segregation energy.

A final time-relaxation run of a 50/50 Ag/Cu rhombicuboctahedron yielded a spherical particle with an outer shell of mostly Ag atoms, with the majority of Cu atoms in the bulk of the nanoparticle. This confirmed the ability of the Molecular Dynamics simulation of a Sutton-Chen potential using Quantum Sutton-Chen parameters to successfully model the segregation in the Ag and Cu system.

## 9.5 References

[1] R. C. Weast, M. J. Astle, eds., *CRC Handbook of Chemistry and Physics*, 63rd ed. (Boca Raton, FL: CRC Press, 1982).

[2] J. Y. Wang, J. du Plessis, J. J. Terblans and G. N. van Wyk, *Surface and Interface Analysis* **28** (1999): 73.

# Chapter 10 - Conclusion

The Sutton-Chen potential was used in Molecular Dynamics simulations to model the diffusion in face-centred cubic (FCC) crystals and develop a model that could successfully simulate the segregation in alloys. In depth analysis of both surface effects and temperature was also conducted.

The study used the Sutton-Chen potential with the original Sutton-Chen parameters to model pure crystals of Al, Ni, Cu, Pt and Ag using periodic boundaries to simulate bulk crystals. For these crystals, the cohesion energy per atom was calculated to validate the model. The cohesion energy ($E_{coh}$) is summarized in Table 10.1. The vacancy formation energy of monovacancies for each metal was also determined, for crystals with (110)-, (100)- and (111)-free surfaces. It was found that the surface binding energy ($E_{surf}$) of a single adatom on the free surfaces had the relationship $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$, and as the vacancy formation energy depends on the surface energy when creating a vacancy-adatom pair, the vacancy formation energy ($E_v$) has the relationship of $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$. $E_v$ at 0 K was also summarized in Table 10.1.

Table 10.1 summarizes the cohesion binding energies and vacancy formation energy of the (100) surface for 6 FCC metals using the SC potential. All energies are given per atom.

| FCC Metal | $E_{coh}$ (eV) | $E_v^{(110)}$ (eV) | $E_v^{(100)}$ (eV) | $E_v^{(111)}$ (eV) |
|---|---|---|---|---|
| **Al** | 3.23 | 0.59 | 0.64 | 0.78 |
| **Ni** | 4.24 | 1.38 | 1.68 | 1.91 |
| **Cu** | 3.34 | 1.08 | 1.33 | 1.51 |
| **Pd** | 3.81 | 1.49 | 1.80 | 2.09 |
| **Ag** | 2.82 | 1.20 | 1.47 | 1.65 |
| **Pt** | 5.75 | 1.41 | 1.59 | 1.96 |

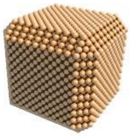
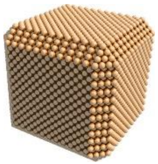
The effect of temperatures between 0 K and 1000 K on the vacancy formation energy was also measured. It was found that Al was not adequately modelled by the Sutton-Chen parameters, showing significant melting of the surface at 400 K, more than 500 K lower than the true melting temperature of Al. Cu also showed surface pre-melting at 900 K. Using temperature ranges that excluded premelting, it was found that with crystal expansion from increased temperature, and the change in surface binding energies resulting from fluctuations in surface structure, the vacancy formation energy decreased from 4 % to 17 % over the range of temperatures measured, up to 0.28 eV/at in the case of $\mathrm{Ag}^{(111)}$. The rates of change were also dependent on the orientation of the free surface. Thus, the activation energy for vacancy formation decreases with increased temperature, but also depends on surface orientation.

From characterizing the bulk materials, the study moved to characterizing nanomaterials, specifically Cu nanocubes, rhombicuboctahedrons and octahedrons. Particles ranged in size from 0.7 nm $3\times3\times3$ packed cubes to large 5 nm $15\times15\times15$ packed Cu cubes with {100} faces. Rhombicuboctahedrons are imperfect cubes with layers shaved off the edges and corners to expose {110} surfaces on the edges and {111} surfaces on the corners. These particles are classified, first by the size of the original cube in layers, and then by the number of layers shaved off the edges and corners. Thus, $15_{-3}$ denotes a $15\times15\times15$ packed cube with three layers shaved off the corners and edges. Octahedrons are two inverted pyramids, with {111} free surfaces. The focus was on the effect of size and shape on particle stability and reactivity. The rhombicuboctahedrons still showed surface orientation dependency in $E_{surf}$, as in the largest crystal $E_{surf}^{(111)} = 2.78$ eV, $E_{surf}^{(100)} = 2.98$ eV and $E_{surf}^{(110)} = 3.25$ eV.

$E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$ also applies to rhombicuboctahedrons, but edges between different surface orientations influence surface energies. Changing the shape by cutting away more layers of atoms from the edges and corners provide larger {110} and {111} surfaces and changed the reactivity of the particles. Comparing $E_{coh}$ and $E_v$ showed that larger particles are more stable, with a more negative $E_{coh}$, while particles with more exposed {110} edges are more reactive. The most stable particle shape for each particle size was chosen for temperature-dependence study, namely the $3_0$, $5_{-1}$, $7_{-1}$, $9_{-2}$, $11_{-2}$, $13_{-3}$ and $15_{-3}$ particles.

Using the average potential energy (PE) per atom which is equivalent to $E_{coh}$, the melting behaviour of these stable particles was studied. As with bulk materials, $E_v$ decreased an average of 0.13 eV as temperature increased. Fluctuations in PE/atom also clearly indicated the temperature range at which the particle melted, and this melting temperature changed with size. The $3_0$ and $5_{-1}$ particles were deemed too unstable as $3_0$ changed to hexagonal close-packed (HCP) at 50 K, and $5_{-1}$ changed to HCP at 450 K. The melting temperatures of the other particles are noted in Table 10.2.

Table 10.2. A summary of temperatures at which Cu rhombicuboctahedrons of various sizes changed state, either by rearranging into an HCP structure, or by melting, as simulated by the Sutton-Chen. Particles sizes are denoted as $x_{-i}$, which means a $x \times x \times x$-packed particle with $i$ layers cut away from the edges and corners.

| **Particle ( $x_{-i}$ )** | **Temperature (K)** |
|---|---|
| **$3_0$** | 50 (HCP) |
| **$5_{-1}$** | 450 (HCP) |
| **$7_{-1}$** | 750 (melt) |
| **$9_{-2}$** | 850 (melt) |
| **$11_{-2}$** | 850 (melt) |
| **$13_{-3}$**  | 900 (melt) |
| **$15_{-3}$**  | 900 (melt) |

In order to prevent the "flying ice-cube effect", where the particle as a whole has a high average velocity but atoms become closely packed and appear "frozen", it was necessary to include Dissipative Particle Dynamics in the Berendsen thermostat, which was used to regulate the temperature. As the $13_{-3}$ and $15_{-3}$ particles have diffusion energies closely resembling that found in bulk crystals, $7_{-1}$, $9_{-2}$ and $11_{-2}$ rhombicuboctahedrons were used to further the study of diffusion and segregation in nanoparticles.

Having chosen nanoparticle shapes and sizes for comparing to bulk materials, the study focussed on Ag and Cu which are known to segregate. The Sutton-Chen parameters were revisited and specifically optimized for calculating cohesion energy $E_{coh}$, vacancy formation energy $E_v$ and diffusion activation energy ($Q$) in these two metals, and the Quantum Sutton-Chen (QSC) parameters were chosen to give the closest results when compared to literature. These produce a different $E_v$ value to the values from the original parameters. The new values for relaxed bulk and nano-crystal energies are summarized in Table 10.3. The same relationships where $E_{surf}^{(111)} < E_{surf}^{(100)} < E_{surf}^{(110)}$ and $E_v^{(111)} > E_v^{(100)} > E_v^{(110)}$, were found with the new QSC parameters in both pure Ag and Cu. As diffusion activation energy $Q$ depends on $E_v$, it was found that $Q^{(110)} < Q^{(100)} < Q^{(111)}$.

Table 10.3. The vacancy formation energy and diffusion activation energy for Cu and Ag bulk- and $7_{-1}$ nano-crystals. All energies are given per atom.

| Crystal | Free Surface | $E_v$ (eV) | | | | $Q$ (eV) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | ***Bulk*** | **$7_{-1}$** | **$9_{-2}$** | **$11_{-2}$** | ***Bulk*** | **$7_{-1}$** | **$9_{-2}$** | **$11_{-2}$** |
| **Ag** | **{110}** | 1.24 | 1.26 | 1.25 | 1.27 | 2.57 | 2.59 | 2.58 | 2.60 |
| | **{100}** | 1.46 | 1.49 | 1.49 | 1.50 | 2.79 | 2.82 | 2.82 | 2.83 |
| | **{111}** | 1.66 | 1.81 | 1.76 | 1.79 | 2.99 | 3.14 | 3.09 | 3.12 |
| **Cu** | **{110}** | 1.34 | 1.54 | 1.51 | 1.51 | 2.52 | 2.72 | 2.69 | 2.68 |
| | **{100}** | 1.59 | 1.75 | 1.77 | 1.75 | 2.77 | 2.93 | 2.94 | 2.92 |
| | **{110}** | 1.75 | 2.23 | 2.06 | 2.07 | 2.93 | 3.41 | 3.23 | 3.24 |

The energies for the larger nanocrystals approached those of the bulk. However, the diffusion-related energies of the $7_{-1}$ crystal diverged from bulk values significantly, as the $7_{-1}$ particle has edge effects which influence surface energies of its {110} and {111} edges and corners. Thus, for the last part of the study, the $7_{-1}$ rhombicuboctahedron was used to contrast with bulk energy values.

The final part of the study examined the Ag/Cu alloy, using the mixed Sutton-Chen potential with slight modifications, which ensure the conservation of energy and momentum. The diffusion activation energy for a Cu impurity in Ag crystals and Ag impurity in Cu crystals is summarized in Table 10.4. The activation energies are larger for Ag in Cu than the values for the pure metals in Table 10.3, and the energies are lower for Cu in Ag. The rhombicuboctahedron values for {110} and {111} surfaces are more a result of edge-effects than the typical free surface found in bulk crystals.

Table 10.4. The diffusion activation and segregation energies for impurities in Ag and Cu are summarized for bulk and nanocrystals. All energies are given per atom.

| **Adatom in Crystal** | **Free Surface** | ***Q* (eV)** | | **Δ*G* (eV)** | |
|---|---|---|---|---|---|
| | | ***Bulk*** | **$7_{-1}$** | ***Bulk*** | **$7_{-1}$** |
| **Ag atom in Cu crystal** | **{110}** | 3.93 | 3.52 | -0.28 | -0.34 |
| | **{100}** | 4.16 | 3.75 | -0.27 | -0.30 |
| | **{111}** | 4.45 | 4.22 | -0.21 | -0.35 |
| **Cu atom in Ag crystal** | **{110}** | 1.76 | 1.84 | 0.05 | 0.06 |
| | **{100}** | 1.93 | 2.03 | 0.07 | 0.08 |
| | **{110}** | 2.14 | 2.40 | 0.06 | 0.05 |

In the calculation of the diffusion activation energy, the minima of the migration energy profile indicated that Ag would segregate to the surface, and Cu into the bulk. The segregation energy of a Ag atom in a Cu crystal and for a Cu atom in a Ag crystal was calculated and the results are also summarized in Table 10.4. The negative segregation energy for a Ag atom in a Cu crystal indicates a drive for Ag to diffuse to the surface. The slight positive energy of the Cu atom in a Ag crystal indicates a weak drive for Cu atoms to desegregate into the bulk of the Ag crystal. After the crystals are allowed to relax, the relationship $\Delta G^{(111)} < \Delta G^{(100)} < \Delta G^{(110)}$ indicating surface orientation dependence is not seen. There is no strong correlation between surface orientation and segregation energy, beyond a slight lesser preference of Ag atoms for the Cu{111} surface. The shape of the rhombicuboctahedron did have a larger effect on the segregation energy, particularly in the case of the Ag atom in a Cu crystal. Here slight differences in energy are less a result of the surface orientation, which determines the number of first nearest neighbours of an atom, and more due to the foreign atom's position on a corner or edge, as opposed to a face. This position on an edge or corner has fewer first, second, third etc. nearest neighbours, and as segregation affects a larger volume of atoms than just immediate nearest neighbours, these positions have a greater impact on segregation energy values.

In the final part of this study, a time-lapse run of a 50/50 Ag/Cu $7_{-1}$ rhombicuboctahedron was run for 500 000 integration steps at 650 K. The result showed conclusively that the modified mixed Sutton-Chen potential successfully simulated surface segregation, by forming a thin shell of Ag on the surface of the nanoparticle.

A future relaxation run at a lower temperature could test the shape dependence of the segregation energy in a rhombicuboctahedron predicted by the segregation energy in Table 10.4. The effect of temperature on segregation could also be studied. Further improvements in calibrating the Sutton-Chen parameters would improve the calculated results, and the model could be a useful tool to study other segregating alloys, impurities and dopants.

# Appendix A - Computer Code

This appendix contains the computer code written during the course of this study. The code was written in Visual Basic. The following code includes the main code for the nearest neighbour list set up, migration energy measurements, integration step, Berendsen thermostat, Dissipative Particle Dynamics, Sutton-Chen potential calculations, Velocity Verlet algorithm, periodic boundaries, surface binding energy measurement and Multi-Section Method, described in Chapters 3 and 4. Comments are given in green to clarify points on the code.

## Nearest Neighbours List

```
Public Sub nearestNeighbours(ByRef g_cut_off_radius As Double, ByRef g_dbl_periodic_boundary_x As Double, ByRef g_dbl_periodic_boundary_y As Double, _
                                 ByRef bln_bulk As Boolean, ByRef dbl_atom_radius As Double)
        'This is a simple function to find which atoms are close enough to each other to react. The distances between atoms
        'and themselves are not calculated. Those that are close enough to interact are written into a nearest neighbour
        'list. The nearest neighbour's index is written into the next available space in an array that lists the nearest neighbours
        'for each atom. It is also stored wether or not the nearest neighbour is the same element as the selected atom.
        'The total number of nearest neihgbours for each atom is stored in the 0 array position.

        'Note g_int16_nearest_neighbour_information(A, 0, 0) is where the total number of nearest neighbours
        'for atom A is stored in the array. The number_of_nn array keeps track of how many neighbours each atom has and refreshes each
        'time nn are recalculated. The index for the first nearest neighbour is stored in g_int16_nearest_neighbour_information(A, 1, 0),
        'the second in g_int16_nearest_neighbour_information(A, 2, 0) and so on.

        'g_int_element is either 0 or 1. The element index can range from 0 to 3, depending on wether the two nearest neighbours
        'selected are both el1, both el2, or el1-2, or el2-1. As g_int_element = 0 for el1 and 1 for el2, and same_element = 0
        'and different_element = 1, then the element index  in g_int_nearest_neighbour_information should work as follows:
        'el1-el1 = 0
        'el1-el2 = 1
        'el2-el2 = 2
        'el2-el1 = 3

        'Lastly a pointer value is stored that associates the two nearest neighbours with each other.
        Const same_element As Integer = 0  'Both selected atom and selected nearest neighbour are of same element
        Const different_element As Integer = 1  'Selected atom and selected nearest neighbour are of differnt elements

        Dim number_of_nn(g_int_total_system_atoms - 1) As Integer
        Dim inter_neighbour_distance As Double


        For i As Integer = 0 To g_int_total_system_atoms - 1
```

```
            For j As Integer = 0 To i - 1

                Dim j_pointer As Integer    'points to where j is in i's list of nn
                Dim i_pointer As Integer    'points to where i is in j's list of nn
                'This provides a temporary pointer in case it's a nearest neighbour

                j_pointer = number_of_nn(i) + 1
                i_pointer = number_of_nn(j) + 1

                interAtomicDisplacement(i, j, i_pointer, j_pointer, bln_bulk, g_cut_off_radius, g_dbl_periodic_boundary_x, g_dbl_periodic_boundary_y)
                inter_neighbour_distance = interAtomicDistance(i, j_pointer)

                If inter_neighbour_distance < g_cut_off_radius Then

                    number_of_nn(i) = j_pointer
                    number_of_nn(j) = i_pointer

                    g_int16_nearest_neighbour_information(i, j_pointer, index) = j           'the i_pointer'th nearest neighbour of i is j.
                    g_int16_nearest_neighbour_information(j, i_pointer, index) = i

                    g_int16_nearest_neighbour_information(i, j_pointer, pointer) = i_pointer      'To find i in the nearest neighbour list of j, look at the j_pointer'th n
earest neighbour of j
                    g_int16_nearest_neighbour_information(j, i_pointer, pointer) = j_pointer

                    'save inter_neighbour_distance for use in potential energy calculation for both i and j
                    g_dbl_interatomic_distance(i, j_pointer) = inter_neighbour_distance
                    g_dbl_interatomic_distance(j, i_pointer) = inter_neighbour_distance

                    'save the interaction of elements if bimetallic
                    If g_int_number_of_elements = 2 Then
                        If g_byt_element(i) = g_byt_element(j) Then
                            g_int16_nearest_neighbour_information(i, j_pointer, element) = same_element + 2 * g_byt_element(i)
                            g_int16_nearest_neighbour_information(j, i_pointer, element) = same_element + 2 * g_byt_element(j)
                        Else
                            g_int16_nearest_neighbour_information(i, j_pointer, element) = different_element + 2 * g_byt_element(i)
                            g_int16_nearest_neighbour_information(j, i_pointer, element) = different_element + 2 * g_byt_element(j)
                        End If
                    Else
                        g_int16_nearest_neighbour_information(i, j_pointer, element) = 0
                        g_int16_nearest_neighbour_information(j, i_pointer, element) = 0
                    End If

                End If

            Next j

        Next i

        For k As Integer = 0 To g_int_total_system_atoms - 1
            g_int16_nearest_neighbour_information(k, 0, index) = number_of_nn(k)
        Next k


    End Sub
```

## Migration Enegy Measurement

```
 Public Sub RunA()
        'This sub measures the migration energy of an adatom as it is stepped towards the position of its nearest neighbour
        Dim number_of_steps As Integer
        Dim dbl_result_potential_energy As Double
        Dim dbl_initial_potential_energy As Double
        Dim dbl_zero_potential_energy As Double
        Dim dbl_transpose_potential_energy As Double = 0
        Dim dbl_migration_distance As Double
        Dim dbl_total_migration_dist As Double
        Dim bln_centre As Boolean
        Dim bln_end As Boolean

        Dim lst_all_data As New List(Of String)
        Dim lst_migration_atoms As New List(Of Integer)

        Dim str_element(g_int_total_system_atoms - 1) As String

        ReDim g_sgl_fixed_atom_positions(0, 3)
        ReDim g_sgl_fixed_atom_velocities(0, 3)
        Dim selected_atom_positions(g_int_total_system_atoms - 1, 2) As Double
        Dim selected_atom_velocities(g_int_total_system_atoms - 1, 2) As Double

        int_iteration_steps = 0

        For i As Integer = 0 To g_int_total_system_atoms - 1
            str_element(i) = str_element1
            g_byt_element(i) = 0
        Next i

        bln_centre = False
        bln_end = False

        'find the centre atom
        Dim dbl_centre(2) As Double
        Dim point(2) As Double
        Dim distance As Double
        Dim new_distance As Double
        Dim centre_atom As Integer = 0

        findCentreAtom(dbl_centre, point, distance, new_distance, centre_atom, g_dbl_atom_position, g_int_total_system_atoms)

        'The start atom, its nearest neighbours and their nearest neighbours are loaded as selected atoms
        int_selected_atoms = 1
        ReDim g_int_selected_atoms(g_int_total_system_atoms - 1)
        Dim int_start_atom As Integer
        '***
        'SECOND START ATOM
        '=================
        Dim int_start_atom2 As Integer
        Dim bln_second_start_at As Boolean = True
```

```
Dim int_atoms_down As Integer = 0       'how far are we from the surface? to specifiy a second surface atom to step to for 110


int_start_atom = int_start_migration_atom
lst_migration_atoms.Add(int_start_atom)
Try
    If int_start_migration_atom2 <> 0 Then
        int_start_atom2 = int_start_migration_atom2
    Else
        bln_second_start_at = False
    End If
Catch ex As Exception
    bln_second_start_at = False
End Try

'FOR EACH START ATOM:
'====================
'Until the centre of mass is reached:
Do Until bln_end = True

    Dim int_nearest_neighbour As Integer
    Dim int_total_nn As Integer
    Dim dbl_original_start_postion(2) As Double
    Dim dbl_vacancy_position(2) As Double
    Dim dbl_migration_position(2) As Double
    Dim dbl_step_vector(2) As Double

    '*******
    g_int_total_selected_atoms = g_int16_nearest_neighbour_information(int_start_atom, 0, 0)

    selectedMigrationAtoms(int_start_atom, g_int_selected_atoms, g_int_total_selected_atoms)

    For k As Integer = 0 To 2
        dbl_original_start_postion(k) = g_dbl_atom_position(int_start_atom, k)
    Next k

    int_total_nn = g_int16_nearest_neighbour_information(int_start_atom, 0, index)


    '===========================================================
    'discriminate against nearest neighbours that are too far removed from the migration atom
    'set first nearest neighbours as separate array so next nearest neighbours don't confuse matters
    Dim lst_vacancy_atoms As New List(Of Integer)
    Dim int_tot_vac_atoms As Integer
    Dim vacancy_atoms_nn(g_int_max_neighbours - 1) As Integer

    For nn As Integer = 1 To int_total_nn

        int_nearest_neighbour = g_int16_nearest_neighbour_information(int_start_atom, nn, index)

        For k As Integer = 0 To 2
            dbl_vacancy_position(k) = g_dbl_atom_position(int_nearest_neighbour, k)
        Next k

        dbl_total_migration_dist = FindDistanceBtnPts(dbl_vacancy_position, dbl_original_start_postion)
        If dbl_total_migration_dist < dbl_atom_radius * 2.3 Then
            lst_vacancy_atoms.Add(g_int16_nearest_neighbour_information(int_start_atom, nn, index))
        End If
    Next nn

    int_tot_vac_atoms = lst_vacancy_atoms.Count

    Dim vacancy_atoms(int_tot_vac_atoms - 1) As Integer
    For i As Integer = 0 To int_tot_vac_atoms - 1
        vacancy_atoms(i) = lst_vacancy_atoms(i)
    Next i
    '===========================================================

    'reset the start atom element
    If int_fix_start_atom_element = 1 Then
        g_byt_element(int_start_atom) = 1
        str_element(int_start_atom) = str_element2
    End If

    'Refresh system term calculation after changing the elements of the crystal

    updateDistance(g_int_total_system_atoms)
    If g_int_number_of_elements = 2 Then
        calculateLocalNnConc(dbl_atom_radius)
    End If
    nmTerm(g_int_total_system_atoms)

    'writePathandEnvironment(int_start_atom)

    'find the migration atom nearest neighbours

    'save atoms as relaxed
    'Dim save As Integer
    For atom As Integer = 0 To g_int_total_system_atoms - 1

        For k As Integer = 0 To 2
            selected_atom_positions(atom, k) = g_dbl_atom_position(atom, k)
            selected_atom_velocities(atom, k) = g_dbl_atom_velocity(atom, k)
        Next k
    Next atom

    'FIND MIGRATION ATOM BINDING ENERGY
    '==============================
    'save the first binding energy calculation to later transpose the migration energy profile with
    'Find the crystal energy to use to zero measurements
    updateDistance(g_int_total_system_atoms)
    If g_int_number_of_elements = 2 Then
        calculateLocalNnConc(dbl_atom_radius)
    End If
    nmTerm(g_int_total_system_atoms)
    dbl_zero_potential_energy = SystemEnergyCalculation()

    If int_start_atom = 263 Then
        int_start_atom = int_start_atom
```

```
End If
If int_start_atom = 223 Then
    int_start_atom = int_start_atom
End If

'==============================

int_atoms_down = int_atoms_down + 1

'FOR EACH NEAREST NEIGHBOUR:
'===========================
'CALCULATE AND SET PATHS AND NN, CREATE VACANCY
For vac As Integer = 1 To int_tot_vac_atoms

    int_nearest_neighbour = vacancy_atoms(vac - 1)

    'calculate step vector
    For k As Integer = 0 To 2
        dbl_vacancy_position(k) = g_dbl_atom_position(int_nearest_neighbour, k)
    Next k

    setStepVector(dbl_vacancy_position, dbl_original_start_postion, dbl_step_vector)

    'nearest neighbours of vacancy atom
    Dim total_selected_vac As Integer
    selectedMigrationAtoms(int_nearest_neighbour, vacancy_atoms_nn, total_selected_vac)


    '****
    Dim total_selected_nn As Integer

    total_selected_nn = g_int_total_selected_atoms

    '=======================================
    'combine selected atoms and vacancy nn
    For j As Integer = 0 To total_selected_vac - 1
        Dim bln_repeat As Boolean = False
        'check the nearest neighbour hasn't already been selected
        For i As Integer = 0 To g_int_total_selected_atoms - 1
            If g_int_selected_atoms(i) = vacancy_atoms_nn(j) Then
                bln_repeat = True
            End If
        Next i
        'if it's not a repeat, add it to the selected atoms
        If bln_repeat = False Then
            g_int_selected_atoms(total_selected_nn) = vacancy_atoms_nn(j)
            total_selected_nn = total_selected_nn + 1
        End If
    Next j
    '=======================================

    'make sure all nearest neighbours are nearest neighbours of the start atom
    int_added_atom = int_start_atom
    g_int_total_selected_atoms = total_selected_nn
    setSelectAtomNearestNeighbours()
```

```
'remove nn from position to create vacancy
g_dbl_atom_position(int_nearest_neighbour, 2) = g_dbl_atom_position(int_nearest_neighbour, 2) + 1000000

If int_nearest_neighbour = 263 Then
    int_start_atom = int_start_atom
End If
If int_nearest_neighbour = 223 Then
    int_start_atom = int_start_atom
End If

'System energy calculation is saved as a reference energy
updateDistance(g_int_total_system_atoms)
If g_int_number_of_elements = 2 Then
    calculateLocalNnConc(dbl_atom_radius)
End If
nmTerm(g_int_total_system_atoms)
dbl_initial_potential_energy = SystemEnergyCalculation()

'STEP LENGTH
'===========
'calculate an atom radius and add 0.25 atom radius bracket to the measurement length
Dim step_length As Double

step_length = Math.Sqrt(dbl_step_vector(0) ^ 2 + dbl_step_vector(1) ^ 2 + dbl_step_vector(2) ^ 2)

number_of_steps = Math.Floor((dbl_atom_radius / 2.5) / step_length)

'step atom position
For k As Integer = 0 To 2
    g_dbl_atom_position(int_start_atom, k) = g_dbl_atom_position(int_start_atom, k) - (number_of_steps * dbl_step_vector(k))
Next k
'===========

Dim lst_data As New List(Of String)
Dim partial_energy_change(g_int_total_system_atoms - 1) As Double
Dim previous_partial_energy(g_int_total_system_atoms - 1) As Double

'STEP THE MIGRATION ATOM FROM ORIGINAL POSITION TO VACANCY
'=========================================================
'MEASURE ENERGY CHANGE AND SAVE PERTINENT INFORMATION IN LISTS
'for each step:

For steps As Integer = 0 To ((2 * number_of_steps) + 49)


    'calculate system energy
    updateDistance(g_int_total_system_atoms)
    If g_int_number_of_elements = 2 Then
        calculateLocalNnConc(dbl_atom_radius)
    End If
    nmTerm(g_int_total_system_atoms)
    dbl_result_potential_energy = SystemEnergyCalculation()
```

```
'Now, to give the binding energy of the migration atom:
'======================================================
'The bracket calculates the change in crystal energy during atom migration,
dbl_transpose_potential_energy = (dbl_result_potential_energy - dbl_initial_potential_energy) _
    + (dbl_zero_potential_energy - dbl_initial_potential_energy)

'calculate the relative separation between migration atom and vacancy
For k As Integer = 0 To 2
    dbl_migration_position(k) = g_dbl_atom_position(int_start_atom, k)
Next k

dbl_migration_distance = Math.Round(FindDistanceBtnPts(dbl_original_start_postion, dbl_migration_position), 3)
If steps < number_of_steps Then
    dbl_migration_distance = -dbl_migration_distance
End If
If steps = number_of_steps - 1 Then
    dbl_migration_distance = dbl_migration_distance
End If
If steps > (24 + number_of_steps) Then
    dbl_migration_distance = dbl_migration_distance
End If
If steps > (48 + number_of_steps) Then
    dbl_migration_distance = dbl_migration_distance
End If

Dim pos(2) As Double

'step the migration atom
For k As Integer = 0 To 2
    pos(k) = dbl_migration_position(k)
    g_dbl_atom_position(int_start_atom, k) = g_dbl_atom_position(int_start_atom, k) + dbl_step_vector(k)
Next k

'===============
'save the number of selected and relaxed atoms, the migration atom # and vacancy atom #
'the relative separation, position and the difference in energies
lst_all_data.Add(Convert.ToString(int_start_atom & "," & int_nearest_neighbour & "," & dbl_migration_distance _
                                         & "," & pos(0) & "," & pos(1) & "," & pos(2) _
                                          & "," & dbl_transpose_potential_energy & "," & g_int_total_selected_atoms _
                                           & "," & dbl_zero_potential_energy & "," & dbl_initial_potential_energy _
                                           & "," & dbl_result_potential_energy))
'*dbl_result_potential_energy is the total energy of the crystal with a vacancy after the migration atom was
' moved
'*dbl_initial_potential_energy is the total energy of the crystal with a vacancy before the migration atom
' was moved
'*dbl_zero_potential_energy is the total energy of the perfect crystal
'*dbl_transpose_potential_energy is the total change in crystal energy from the perfect crystal to the current
' vacancy/migration atom position


'===============
'write xyz file
lst_data.Add(Convert.ToString(pos(0) & "," & pos(1) & "," & pos(2)))
```

```
    Next steps
    '=========================================================

    writePathandEnvironment(int_start_atom, int_nearest_neighbour, number_of_steps, dbl_original_start_postion, dbl_vacancy_position, lst_data)

    '=====================================
    'RESET SYSTEM FOR NEXT NN
    'return nearest neighbour to original position
    g_dbl_atom_position(int_nearest_neighbour, 2) = g_dbl_atom_position(int_nearest_neighbour, 2) - 1000000

    'return migration atom to original position
    For k As Integer = 0 To 2
        g_dbl_atom_position(int_start_atom, k) = dbl_original_start_postion(k)
    Next k

    'reset selected atoms to only start atom nn
    nearestNeighbours(g_cut_off_radius, g_dbl_periodic_boundary_x, g_dbl_periodic_boundary_y, bln_bulk, dbl_atom_radius)
    selectedMigrationAtoms(int_start_atom, g_int_selected_atoms, g_int_total_selected_atoms)

    'writePathandEnvironment(int_start_atom)

    'reset atoms
    For atom As Integer = 0 To g_int_total_system_atoms - 1
        'save = g_int_selected_atoms(atom)
        For k As Integer = 0 To 2
            g_dbl_atom_position(atom, k) = selected_atom_positions(atom, k)
            g_dbl_atom_velocity(atom, k) = selected_atom_velocities(atom, k)
        Next k
    Next atom
    '=====================================

Next vac
'===========================

'Allow the program to pause if needed
Do Until bln_Pause = False
    Sleep(100)
    System.Windows.Forms.Application.DoEvents()
Loop

'=================================================
'RESET SYSTEM FOR NEXT MIGRATION ATOM
'if relax_nearest_neighbours is selected, reload the original atom positions before starting
If int_relax_nn = 1 Then
    loadAtoms()
End If

'reset the start atom element
If int_fix_start_atom_element = 1 Then
    g_byt_element(int_start_atom) = 0
    str_element(int_start_atom) = str_element1
End If

'Next: choose a nearest neighbour closest to the centre of mass as the start atom
```

```
    Dim next_start_atom As Integer

    distance = 10 ^ 6
    new_distance = 10 ^ 6

    For j As Integer = 0 To int_tot_vac_atoms - 1

        next_start_atom = vacancy_atoms(j)

        For k As Integer = 0 To 2
            point(k) = g_dbl_atom_position(next_start_atom, k)
        Next k

        distance = FindDistanceBtnPts(point, dbl_centre)
        If distance < new_distance Then
            int_start_atom = next_start_atom
            new_distance = distance
        End If
    Next j

    'set second start atom if preselected
    If bln_second_start_at = True Then
        If int_atoms_down = 1 Then
            int_start_atom = int_start_atom2
        End If
    End If

    'Stop when the centre is passed
    If int_start_atom = centre_atom Then
        bln_centre = True
    End If
    If bln_centre = True Then
        If int_start_atom <> centre_atom Then
            bln_end = True
        End If
    End If

    'if start atom element is fixed, toggle atom elements
    If int_fix_start_atom_element = 1 Then
        g_byt_element(int_start_atom) = 1
    End If

    'store the migration atom in a list
    lst_migration_atoms.Add(int_start_atom)

    'recalculate the nearest neighbours for a new migration atom
    nearestNeighbours(g_cut_off_radius, g_dbl_periodic_boundary_x, g_dbl_periodic_boundary_y, bln_bulk, dbl_atom_radius)

'=================================================

Loop
'====================
int_iteration_steps = int_iteration_steps - 50
Dim iso_image_file As New ImageFileHandler()
iso_image_file.SaveISOImages(str_save_file & "\")
```

```
        'WRITE ALL MIGRATION ENERGIES TO FILE
        '=======================================
        'write all data profiles from all nn
        Dim str_all_data As String
        str_all_data = str_load_file & "\all_data_profiles.txt"

        WriteValuesToText(str_all_data, "start_atom,vacancy_nn,migration_distance,posx,posy,posz,result_potential_energy,total_selected_atoms,total_crystal_energy,nn_vac_e
nergy,migration_vac_energy")

        For Each data_line As String In lst_all_data
            WriteValuesToText(str_all_data, data_line)
        Next data_line

        'write out the migration atoms in xyz
        writeMigrationAtoms(lst_migration_atoms)

        'CREATE A DEPTH AGAINST MIGRATION ENERGY PROFILE
        '===============================================
        'load saved file and extract depth profile from surface to centre using index #'s saved
        Dim lst_profile As New List(Of String)
        createMigrationDepthProfile(lst_all_data, lst_migration_atoms, number_of_steps, lst_profile)

        'calculate the segregation and activation energy values to file
        Dim lst_energy_values As New List(Of String)
        calculateMigrationEnergies(lst_profile, lst_energy_values)
        '===============================================

        'WRITE MIGRATION ENERGY PROFILES TO FILE
        '=======================================

        'write continuous depth migration energy profile
        Dim str_profile As String
        str_profile = str_load_file & "\depth_profile.txt"

        For Each data_line As String In lst_profile
            WriteValuesToText(str_profile, data_line)
        Next data_line


        System.IO.File.Copy(str_load_file & "depth_profile.txt", str_load_file & "depth_profile.csv", True)

        'write a file with segregation and activation energies
        Dim str_energy_value As String
        str_energy_value = str_load_file & "\migration_energies.txt"

        For Each data_line As String In lst_energy_values
            WriteValuesToText(str_energy_value, data_line)
        Next data_line
        '=======================================

    End Sub
```

## Time Lapse Relaxation Run

```
Private Sub RunB()

        'The start button runs the main part of this program, which is designed to relax already existing structures.
        'Starting conditions are loaded that contain original positions, velocities and energies. From there an outer loop is
        'started which calls a function to determine the nearest neighbour of each atom in the system. Once this is established,
        'an inner loop runs the integration steps. This inner loop selects the atoms for which the calculations need to be done,
        'does the necessary calculations by calling the relevant modules and functions. The calculated values are returned,
        'stored and used in subsequent calcualtions. The inner loop runs a predetermined number of times, before returning to
        'the outer loop to recalculate the nearest neighbours. The outer loop continues either for a predetermined number of
        'times or until a user interrupt event.


        kineticEnergyAndTemperatureControl()

        If g_int_full_save_after = 0 Then
            g_int_full_save_after = int_user_set_iterations
        End If

        If g_int_image_save_after = 0 Then
            g_int_image_save_after = int_user_set_iterations
        End If

        int_iteration_steps = int_start_step

        time.Restart()

        Do Until int_iteration_steps = int_user_set_iterations Or bln_Stop = True

            int_iteration_steps = int_iteration_steps + 1


            nearestNeighbours(g_cut_off_radius, g_dbl_periodic_boundary_x, g_dbl_periodic_boundary_y, bln_bulk, dbl_atom_radius)

            integrationStep()

            If bln_fixed_base = True Then
                fixBottomLayerPositions()
            End If

            If bln_fixed_atoms = True Then
                fixAtomPosVel()
            End If


            kineticEnergyAndTemperatureControl()

            g_dbl_total_potential_energy = SystemEnergyCalculation()


            'changes the frequency of full saves during the middle of a simulation
            If int_iteration_steps Mod g_int_full_save_after = 0 Then
                If g_int_full_save_to = 0 Then
                    g_int_full_save_to = int_user_set_iterations
                End If
                g_int_full_save = g_int_full_save_to
            End If

            'changes the frequency of image saves during the middle of a simulation
            If int_iteration_steps Mod g_int_image_save_after = 0 Then
                If g_int_image_save_to = 0 Then
                    g_int_image_save_to = int_user_set_iterations
                End If
                g_int_image_save = g_int_image_save_to
            End If

            If int_iteration_steps Mod g_int_full_save = 0 Then
                saveFullResults()
                writePotentialToText()
            End If

            If int_iteration_steps Mod g_int_image_save = 0 Then
                saveImage()
                outputProgress()
            End If


            'Allow execution of windows program if needed
            System.Windows.Forms.Application.DoEvents()

            'Allow the program to pause if needed
            Do Until bln_Pause = False
                Sleep(100)
                System.Windows.Forms.Application.DoEvents()
            Loop

        Loop

        saveFullResults()

        'Convert image files into ISO
        Dim iso_image_file As New ImageFileHandler()
        iso_image_file.SaveISOImages(str_save_file & "\")

        Exit Sub

    End Sub
```

## Integration Step

```
Private Sub integrationStep()

        'This is a sub-routine which runs the integration step. The sub calls a function which randomly selects atoms from the system for which
        'to calculate new data. It then calls the module to do the Sutton Chen calculations for those selected atoms and returns a value which is
        'returned to the main run sub. Either all atoms are selected and moved, or only one is moved at a time, or a few are chosen and moved.
        'After selecting which atoms are being worked with, the sub selects and atom and checks wether the acceleration for that selected atom has
        'previously been calculated. If it has, the acceleration is copied over, if not, it is calculated. With the current acceleration calculated,
        'the sub then calls the calculation that determines the new position of the atom selected. Thereafter new interatomic distances between the
        'selected atom and its nearest neighbours are determined, and a new acceleration is then calculated for the newly determined position.
        'Finally, a new velocity is determined for the selected atom. Once the sub has followed these processes for each selected atom, it exits
        'the sub.

        Dim int_selected_atom As Integer

        '##CALCULATE POSITION##
        '======================
        For i As Integer = 0 To (g_int_total_selected_atoms - 1)

            If g_int_total_selected_atoms = g_int_total_system_atoms Then
                int_selected_atom = i
            Else
                int_selected_atom = g_int_selected_atoms(i)
            End If

            newPositionCalculation(int_selected_atom, g_dbl_time_step, g_dbl_atom_position, g_dbl_atom_velocity, g_dbl_atom_acceleration)

        Next i

        '##CALCULATE VELOCITY 1/2##
        '==================================
        For i As Integer = 0 To (g_int_total_selected_atoms - 1)

            If g_int_total_selected_atoms = g_int_total_system_atoms Then
                int_selected_atom = i
            Else
                int_selected_atom = g_int_selected_atoms(i)
            End If


            newVelocityCalculation(int_selected_atom, g_dbl_time_step, g_dbl_atom_velocity, g_dbl_atom_acceleration)

            'Allow the program to pause if needed
            Do Until bln_Pause = False
                Sleep(100)
                System.Windows.Forms.Application.DoEvents()
            Loop

        Next i


        '##CALCULATE ACCELERATION##
        '===================================
        'Updates the interatomic distances
        updateDistance()
```

```
    'Calculate the density-like term and "n" and "m"-terms.

    nmTerm(g_int_total_system_atoms)

    'calculate new accelerations
    For i As Integer = 0 To (g_int_total_selected_atoms - 1)


        If g_int_total_selected_atoms = g_int_total_system_atoms Then
            int_selected_atom = i
        Else
            int_selected_atom = g_int_selected_atoms(i)
        End If

        'set concentration dependent SC values if alloy
        If g_int_number_of_elements = 2 Then
            setLocalNnSCconstants(int_selected_atom)
        End If

        CalculateAcceleration(int_selected_atom)

        'Allow the program to pause if needed
        Do Until bln_Pause = False
            Sleep(100)
            System.Windows.Forms.Application.DoEvents()
        Loop

    Next i

    '##CALCULATE VELOCITY 2/2##
    '==================================
    For i As Integer = 0 To (g_int_total_selected_atoms - 1)

        If g_int_total_selected_atoms = g_int_total_system_atoms Then
            int_selected_atom = i
        Else
            int_selected_atom = g_int_selected_atoms(i)
        End If

        newVelocityCalculation(int_selected_atom, g_dbl_time_step, g_dbl_atom_velocity, g_dbl_atom_acceleration)

        'Allow the program to pause if needed
        Do Until bln_Pause = False
            Sleep(100)
            System.Windows.Forms.Application.DoEvents()
        Loop

    Next i


End Sub
```

## Temperature Control

```
  Public Sub kineticEnergyAndTemperatureControl()
        'This sub checks wether the system temperature is at the desired temperature. It first calculates the kinetic energy, then the temperature
        'and then checks that system temperature against the desired temperature. If the temperature needs adjusting, the sub calls the
        'Thermostat module, before checking the temperature again.
        'To determine which thermostat needs to be used, the user-set constant is checked: 0 for velocity scaling, 1 for the Berendsen thermostat and 2 for the NoseHoover.
        '(3 for Langevin?)
        Dim scaling_factor As Double

        calculateSystemTemperature()

        'BerendsenTemperatureControl()

        If int_thermostat = 0 Then

            'simple velocity scaling
            scaling_factor = VelocityScalingFactor(g_dbl_desired_temperature, g_dbl_system_temperature)
            VelocityScaling(scaling_factor, g_int_total_system_atoms, g_dbl_atom_velocity)

        ElseIf int_thermostat = 1 Then

            'calculate dissipative particle dynamics noise using current velocities
            Dim noise_factor(g_int_total_system_atoms - 1, 2) As Double
            If bln_run_a = False Then
                If g_dbl_desired_temperature <> 0 Then
                    DissipativeParticleDynamicsScaling(g_int_total_system_atoms, g_cut_off_radius, g_dbl_time_step, g_dbl_desired_temperature, g_sgl_Mass, g_byt_element, _
                                                       g_int16_nearest_neighbour_information, g_dbl_interatomic_distance, g_dbl_atom_velocity, noise_factor)
                End If
            End If

            'Double check the temperature
            calculateSystemTemperature()

            'velocity scaling with Berendsen scaling factor
            scaling_factor = BerendsenScalingFactor(g_dbl_desired_temperature, g_dbl_system_temperature, g_dbl_thermostat_constant, g_dbl_time_step)
            VelocityScaling(scaling_factor, g_int_total_system_atoms, g_dbl_atom_velocity)

            'add calculated noise factors
            If bln_run_a = False Then
                If g_dbl_desired_temperature <> 0 Then
                    AddDPDTerms(noise_factor, g_dbl_atom_velocity, g_int_total_system_atoms)
                End If
            End If

        End If

            VelocitySquaredCalculation(g_dbl_velocity_squared, g_dbl_atom_velocity, g_int_total_system_atoms)
            g_dbl_total_kinetic_energy = KineticEnergyFromVelocitySquared(g_dbl_velocity_squared, g_sgl_Mass, g_byt_element, g_int_total_system_atoms, bln_fixed_base, _
                                                                          g_int_start_fixed_atoms, g_int_end_fixed_atoms)
            g_dbl_system_temperature = TemperatureFromKineticEnergy(g_dbl_total_kinetic_energy, g_int_total_system_atoms)

    End Sub
```

## Berendsen Thermostat

```
Public Function BerendsenScalingFactor(ByVal desired_temperature As Double, ByVal system_temperature As Double, ByVal Berendsen_constant As Double, _
                                           ByVal dt As Double) As Double
        'This function calculates a factor for velocity scaling that corrects the system temperature to the desired temperature by weak coupling.

        Dim scaling_factor As Double

        If system_temperature <> 0 Then
            If desired_temperature <> 0 Then
                scaling_factor = 1 + (dt / Berendsen_constant) * ((desired_temperature / system_temperature) - 1)
            Else
                scaling_factor = 0
            End If
        Else
            scaling_factor = 0
        End If

        scaling_factor = Math.Sqrt(Math.Abs(scaling_factor))

        Return scaling_factor

    End Function
```

## Dissipative Particle Dynamics

```
Public Sub DissipativeParticleDynamicsScaling(ByVal total_system_atoms As Integer, ByVal r_cut_off As Double, ByVal time_step As Double, ByVal Temperature_ref As Double, _
                                                  ByVal mass() As Single, ByVal element() As Byte, ByVal neighbour_info(,,) As Integer, ByVal interatom_dist(,) As Double,
_
                                                  ByVal velocity(,) As Double, ByRef noise_factors(,) As Double)
        'This sub adds noise and friction to the berendsen scaling factor.
        Dim random_number As Double

        'The normal distribution is used to select a scalar value for the velocity
        randomGen = New Troschuetz.Random.MT19937Generator()
        normalDistribution = New Troschuetz.Random.NormalDistribution(randomGen)
        normalDistribution.Sigma = 1        'Sigma of normal distribution
        normalDistribution.Mu = 1           'The mean value

        random_number = normalDistribution.NextDouble()

        Dim r_interatom As Double

        For i As Integer = 0 To total_system_atoms - 1

            'For each atom choose its nearest neighbour according to distance weighted probability
            'get random number and multiply with r_cut_off/2.6
            Dim random_dist As Double
            random_number = normalDistribution.NextDouble()
            random_dist = (r_cut_off / 2.6) * random_number

            'compare the random distance with the interatomic distances of the nearest neighbours and find the one closest to it
            Dim nn As Integer
            Dim search_nn As Integer
            Dim max_nn As Integer
            Dim min_dist As Double

            nn = 1
            search_nn = 1
            max_nn = neighbour_info(i, 0, 0)
            If max_nn < 1 Then
                GoTo skipline
            End If
            min_dist = Math.Abs(random_dist - interatom_dist(i, search_nn))

            Do Until search_nn = max_nn
                search_nn = search_nn + 1
                Dim new_dist As Double
                new_dist = Math.Abs(random_dist - interatom_dist(i, search_nn))
                min_dist = Math.Min(min_dist, new_dist)

                If min_dist = new_dist Then
                    nn = search_nn
                End If

            Loop

skipline:

            'Find interatomic distance
            r_interatom = interatom_dist(i, nn)

            'find damping factor
            Dim damping_factor As Double
            damping_factor = dpdDampingFactor(r_cut_off, r_interatom, time_step)

            'find velocity noise factor
            Dim j As Integer
            Dim vel_noise_factor As Double
            Dim weighted_mass As Double

            j = neighbour_info(i, nn, 0)
            weighted_mass = (mass(element(i)) * mass(element(j))) / (mass(element(i)) + mass(element(j)))
            vel_noise_factor = Math.Sqrt((damping_factor * (2 - damping_factor) * k_boltzmann * Temperature_ref) / weighted_mass)

            'find the components to the relative velocity vector
            Dim relative_vel(2) As Double
            Dim random_vector(2) As Double

            relative_vel(0) = velocity(i, 0) - velocity(j, 0)
            random_vector(0) = normalDistribution.NextDouble()


            relative_vel(1) = velocity(i, 1) - velocity(j, 1)
            random_vector(1) = normalDistribution.NextDouble()

            relative_vel(2) = velocity(i, 2) - velocity(j, 2)
            random_vector(2) = normalDistribution.NextDouble()
            'determine the direction of the noise vector
            Dim unit_vector(2) As Double
            Dim unit_vector_r As Double

            unit_vector_r = Math.Sqrt(relative_vel(0) ^ 2 + relative_vel(1) ^ 2 + relative_vel(2) ^ 2)
            If unit_vector_r = 0 Then
                unit_vector_r = 1
            End If
            For k As Integer = 0 To 2
                unit_vector(k) = relative_vel(k) / unit_vector_r
            Next

            'find the change in velocity due to noise and friction
            Dim velocity_change(2) As Double
            For k As Integer = 0 To 2
                velocity_change(k) = ((-damping_factor) * (relative_vel(k)) + vel_noise_factor * random_vector(k)) * unit_vector(k)
            Next k

            'distribute weighted relative velocity
            Dim mass_factor_i As Double
            Dim mass_factor_j As Double

            mass_factor_i = weighted_mass / mass(element(i))
            mass_factor_j = weighted_mass / mass(element(j))
            For k As Integer = 0 To 2
                noise_factors(i, k) = noise_factors(i, k) + (mass_factor_i * velocity_change(k))
                noise_factors(j, k) = noise_factors(j, k) - (mass_factor_j * velocity_change(k))
            Next k

        Next i

    End Sub
```

## Dissipative Particle Dynamics Damping Factor

```
  Public Function dpdDampingFactor(ByVal r_cut_off As Double, ByVal r_interatom As Double, ByVal time_step As Double)
        'This function calculates the factor used to determine the dpd noise

        Dim damping_factor As Double

        If r_interatom < r_cut_off Then
            damping_factor = Math.Abs((r_cut_off - r_interatom))
            damping_factor = (-damping_factor) * time_step * 0.0001
        Else
            damping_factor = 0
        End If

        damping_factor = 1 - Math.Exp(damping_factor)

        Return damping_factor

    End Function
```

## Sutton-Chen Module

```
Module SuttonChenCalculations

    'Variables
    Dim Density As Double
    Dim Energy As Double
    Dim acceleration As Double
    Dim nearest_neighbour As Integer

    'Locally used arrays with stored terms for re-use
    Public g_cut_off_radius As Single
    Public partial_energy() As Double

    'Arrays holding SC constants
    Public g_dbl_SC_constants(,) As Double

 'This module determines the acceleration and energy of the atoms specified using Sutton Chen potential calculations.
    'This module needs an input of the Sutton Chen constants for the relevant materials, the lattice parameters
    'and the interatomic distances, as well as the atoms for which to do calculations and the nearest
    'neighbours for the selected atom. The module first uses the interatomic distances and the lattice parameter
    'along with the SC constant m to calculate the density like terms pi in a separate function. This
    'term is then used with the atom, its nearest neighbours and the SC constants supplied to then calculate the forces.
    'First the individual forces of each nearest neighbour influencing the selected atom is calculated,
    'and then all the forces are summed to find the total force acting on the selected atom. Finally
    'the force is used to calculate the acceleration.

    'Constants
    Public Const m As Integer = 0
    Public Const n As Integer = 1
    Public Const a As Integer = 2
    Public Const e As Integer = 3
    Public Const c As Integer = 4
    Const correction_factor As Single = 9648.5
    Const same_element As Integer = 0  'Both selected atom and selected nearest neighbour are of same element
    Const different_element As Integer = 1  'Selected atom and selected nearest neighbour are of differnt elements
```

## “nm” Term Calculation

```
 Public Sub nmTermCalculation(ByVal D As Integer, Optional ByVal run_B As Boolean = False) ', ByRef g_int16_nearest_neighbour_information As Array, ByRef g_dbl_nn_SC_terms
 As Array, _
        'ByRef g_dbl_interatomic_distance As Array)
        'This function calculates the m-term for the atom number that is passed in, with all its nearest neighbours.
        'Note the upper limit is g_int_nearest_neighbour_list(A, 0, 0), which is where the total number of nearest neighbours
        'for atom A is tored in the array. The index for the first nearest neighbour is stored in g_int16_nearest_neighbour_information(A, 1, 0)
        'and so on.

        'In order to know which Sutton Chen parameters to use, the selected atom - nearest neighbour el - el needs
        'to be known. These are stored in the nearest_neighbour_element(i, j, #) array, where the elements of i and j (1 or 0) is stored in
        'g_int16_nearest_neighbour_information(i, 0, element):

        'el1-el1 = 0
        'el1-el2 = 1
        'el2-el2 = 2
        'el2-el1 = 3
        Dim mterm As Double
        Dim nterm As Double
        Dim sc_element As Integer
        Dim nearest_neighbour As Integer
        Dim nn_pointer As Integer

        For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)
            nearest_neighbour = g_int16_nearest_neighbour_information(D, nn, index)

            If run_B = True Then
                If nearest_neighbour > D Then
                    GoTo SkipRepition
                End If
            End If

            nn_pointer = g_int16_nearest_neighbour_information(D, nn, pointer)

            sc_element = g_int16_nearest_neighbour_information(D, nn, element)

            mterm = (g_dbl_SC_constants(a, sc_element) / g_dbl_interatomic_distance(D, nn)) ^ g_dbl_SC_constants(m, sc_element)

            g_dbl_nn_SC_terms(D, nn, 0) = mterm
            If run_B = True Then
                g_dbl_nn_SC_terms(nearest_neighbour, nn_pointer, 0) = mterm
            End If

            nterm = (g_dbl_SC_constants(a, sc_element) / g_dbl_interatomic_distance(D, nn)) ^ g_dbl_SC_constants(n, sc_element)

            g_dbl_nn_SC_terms(D, nn, 1) = nterm
            If run_B = True Then
                g_dbl_nn_SC_terms(nearest_neighbour, nn_pointer, 1) = nterm
            End If

SkipRepition:

        Next nn

    End Sub
```

## Density Term Calculation

```
 Public Sub ElectronDensityTermCalculation(ByVal D As Integer) ', ByRef g_int16_nearest_neighbour_information As Array, ByRef g_dbl_nn_SC_terms As Array, _
        'ByRef g_sgl_interatomic_distance As Array)
        'This function calculates the density term for the atom number that is passed in, with all its nearest neighbours.

        Density = 0

        For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)
            Density = Density + g_dbl_nn_SC_terms(D, nn, 0)
        Next nn

        g_dbl_nn_SC_terms(D, 0, 0) = Math.Sqrt(Density)

    End Sub
```

## Force and Acceleration Calculation

```
Public Sub CalculateAcceleration(ByVal D As Integer) 'ByRef g_dbl_atom_position As Array, ByRef g_int16_nearest_neighbour_information _
        'As Array, ByRef g_dbl_interatomic_distance As Array, ByRef g_dbl_nn_SC_terms As Array, ByRef g_dbl_atom_acceleration As Array)
        'This sub calculates the acceleration for the selected atom D. It first calls a density term calculation for the selected atom,
        'then runs through the list of nearest neighbours, does a density term calculation for each and sums the individual force's components.
        'Finally it calculates the acceleration from the total force. k is the xyz component.

        'g_int_element is either 0 or 1. The element index can range from 0 to 3, depending on wether the two nearest neighbours
        'selected are both el1, both el2, or el1-2, or el2-1. As g_int_element = 0 for el1 and 1 for el2, and same_element = 0
        'and different_element = 1, then the element index  in g_int_nearest_neighbour_information should work as follows:
        'el1-el1 = 0
        'el1-el2 = 1
        'el2-el2 = 2
        'el2-el1 = 3

        'The g_dbl_nn_SC_terms array works with a similar system to that of the g_int_nearest_neighbour_information array: the first index indicates
        'the atom with which you are working, while the second index indicates what the number of nearest neighbour you are working with is. This g_dbl_nn_SC_terms
        'array stores (a/r)^m values between selected_atom and nearest_neighbour and the square root of the sum of these values for each selected_atom at
        'g_dbl_nn_SC_terms(selected_atom, 0).

        'It's also important to have the global array g_dbl_interatomic_displacement with the following structure: 3-d array with the first dimension having the primary
        'index of the atom for which the displacement is used; the second dimension stores the # of nearest neighbour that is displaced from the primary atom; the third
        'dimension of the array stores the x, y, and z components of the displacement in positions 0, 1, and 2 respectively.

        Dim Force() As Double = {0, 0, 0}
        Dim sc_element As Integer
        Dim nearest_neighbour As Integer
        Dim partial_force_constant As Double

        For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)

            nearest_neighbour = g_int16_nearest_neighbour_information(D, nn, index)
            sc_element = g_int16_nearest_neighbour_information(D, nn, element)

            partial_force_constant = (-g_dbl_SC_constants(e, sc_element) / (g_dbl_interatomic_distance(D, nn) * g_dbl_interatomic_distance(D, nn))) _
                * ((g_dbl_SC_constants(n, sc_element) * g_dbl_nn_SC_terms(D, nn, 1)) _
                   + ((-((g_dbl_SC_constants(m, sc_element) * g_dbl_SC_constants(c, sc_element)) / 2)) * (g_dbl_nn_SC_terms(D, nn, 0))) _
                   * ((1 / g_dbl_nn_SC_terms(D, 0, 0)) + (1 / g_dbl_nn_SC_terms(nearest_neighbour, 0, 0))))

            Force(0) = Force(0) - partial_force_constant * g_dbl_interatomic_displacement(D, nn, 0)
            Force(1) = Force(1) - partial_force_constant * g_dbl_interatomic_displacement(D, nn, 1)
            Force(2) = Force(2) - partial_force_constant * g_dbl_interatomic_displacement(D, nn, 2)

        Next nn

        g_dbl_atom_acceleration(D, 0) = correction_factor * Force(0) / g_sgl_Mass(g_byt_element(D))
        g_dbl_atom_acceleration(D, 1) = correction_factor * Force(1) / g_sgl_Mass(g_byt_element(D))
        g_dbl_atom_acceleration(D, 2) = correction_factor * Force(2) / g_sgl_Mass(g_byt_element(D))

    End Sub
```

## System Energy Calculation

```
 Public Function SystemEnergyCalculation(Optional ByRef dbl_partial_energy() As Double = Nothing) 'ByVal Max As Integer) ', ByRef g_int16_nearest_neighbour_information As
Array, ByRef g_dbl_interatomic_distance As Array, _
        'ByRef g_dbl_nn_SC_terms As Array)     ', Optional ByVal atom_number As Integer = 0)  '*for MC

        '    'This function runs through all the atoms in the system and calculates the partial energies, sums them and returns the total energy.
        '    'Later it can be adjusted to only work out the partial energies of a selected few atoms who's positions and velocities have changed,
        '    'before summing all partial energies.

        '    'Currently the MC alternative which involves calculating the nearest and next nearest neighbours of only a limited
        '    'amount of atoms has been conscripted for use in the potential measurement RunC sub. For this purpose the index of the
        '    'added atom which is varied is passed as "index", and only those terms are calculated.

        Energy = 0

        For a As Integer = 0 To g_int_total_system_atoms - 1
            If a = 101 Then
                a = 101
            End If
            partial_energy(a) = NewEnergyTermCalculation(a) ', g_int16_nearest_neighbour_information, g_dbl_interatomic_distance, g_dbl_nn_SC_terms)
        Next a

        For i As Integer = 0 To g_int_total_system_atoms - 1
            Energy = Energy + partial_energy(i)
        Next i

        Return Energy

    End Function
```

## Energy Term Calculation

```
 Public Function NewEnergyTermCalculation(ByVal D As Integer) ', ByRef g_int16_nearest_neighbour_information As Array, _
        'ByRef g_dbl_interatomic_distance As Array, ByRef g_dbl_nn_SC_terms As Array) As Double
        'This function calculates the energy for each individual atom. It can be optimized.
        'Note the upper limit is g_int_nearest_neighbour_list(D, 0), which is where the total number of nearest neighbours
        'for atom D is stored in the array. The index for the first nearest neighbour is stored in g_int_nearest_neighbour_list(D, 1)
        'and so on.

        'In order to know which Sutton Chen parameters to use, the selected atom - nearest neighbour el - el needs
        'to be known. These are stored in the nearest_neighbour_element(i, j) array, where the element (1 or 0) os stored in
        'nearest_neighbour_element(i, 0). nearest_neighbour_element(0, j) hold the information for the selected atom D passed into
        'the function, with nearest_neighbour_element(0, 0) storing the element of atom D and nearest_neighbour_element(0, 1-12)
        'storing the relationship between atom D and its nearest neighbours 1 - 12. Similarly nearest_neighbour_element(1-12, j) holds
        'the information of the nearest neighbours 1-12 of atom D.


        Dim Energy_term As Double
        Dim nn_summation As Double
        Dim sc_element As Integer
        Dim nearest_neighbour As Integer

        nn_summation = 0
        'set concentration dependent SC values if alloy
        If g_int_number_of_elements = 2 Then
            SimulationProgram.setLocalNnSCconstants(D)
        End If

        For nn As Integer = 1 To g_int16_nearest_neighbour_information(D, 0, index)
            nearest_neighbour = g_int16_nearest_neighbour_information(D, nn, index)
            sc_element = g_int16_nearest_neighbour_information(D, nn, element)

            nn_summation = nn_summation + g_dbl_nn_SC_terms(D, nn, 1)

        Next nn

        If g_int_number_of_elements = 2 Then
            If g_int16_nearest_neighbour_information(D, 0, conc) = 0 Then
                sc_element = 2
            ElseIf g_int16_nearest_neighbour_information(D, 0, conc) = 10 Then
                sc_element = 0
            Else
                sc_element = 1
            End If
        Else
            sc_element = 0
        End If

        Energy_term = g_dbl_SC_constants(e, sc_element) * ((nn_summation / 2) - g_dbl_SC_constants(c, sc_element) * g_dbl_nn_SC_terms(D, 0, 0))

        Return Energy_term

    End Function

End Module
```

## Velocity Verlet Module

```
Module VerletAlgorithmCalculation


    'This module needs access to four arrays; atom_position, which is 2-dimensional and contains the index of the atom in the first dimension
    'and the x,y,z-component of the position in the second dimension; a similarly structured atom_velocity array, and two 1-dimensional arrays;
    'sgl_acceleration for acceleration after time step dt, which contain only the x,y,z-components of the
    'accelrations for the currently selected atom i.
```

## Position Calculation

```
    Public Sub newPositionCalculation(ByVal i As Integer, ByVal dt As Double, ByRef g_dbl_atom_position As Array, ByRef g_dbl_atom_velocity As Array, ByRef g_sgl_accelerat
ion As Array)
        'This module will be used to calculate the position of the selected atom i at a time t+dt from values of current
        'position, current velocity and current acceleration supplied.

        g_dbl_atom_position(i, 0) = g_dbl_atom_position(i, 0) + dt * (g_dbl_atom_velocity(i, 0)) + ((1 / 2) * (dt * dt) * g_sgl_acceleration(i, 0))
        g_dbl_atom_position(i, 1) = g_dbl_atom_position(i, 1) + dt * (g_dbl_atom_velocity(i, 1)) + ((1 / 2) * (dt * dt) * g_sgl_acceleration(i, 1))
        g_dbl_atom_position(i, 2) = g_dbl_atom_position(i, 2) + dt * (g_dbl_atom_velocity(i, 2)) + ((1 / 2) * (dt * dt) * g_sgl_acceleration(i, 2))

    End Sub
```

## Velocity Calculation

```
    Public Sub newVelocityCalculation(ByVal i As Integer, ByVal dt As Double, ByRef g_dbl_atom_velocity As Array, ByRef g_dbl_acceleration As Array)
        'This module will be used to calculate the velocity of the selected atom i at a time t+dt from values of current
        'velocity, current acceleration and the acceleration at time t+dt supplied.

        g_dbl_atom_velocity(i, 0) = g_dbl_atom_velocity(i, 0) + (1 / 2) * g_dbl_acceleration(i, 0) * dt
        g_dbl_atom_velocity(i, 1) = g_dbl_atom_velocity(i, 1) + (1 / 2) * g_dbl_acceleration(i, 1) * dt
        g_dbl_atom_velocity(i, 2) = g_dbl_atom_velocity(i, 2) + (1 / 2) * g_dbl_acceleration(i, 2) * dt

    End Sub

End Module
```

## Periodic Boundary Module

```
Module PeriodicBoundaries

    Public Sub CopyCentralBlock(ByRef positions(,) As Double, ByRef velocities(,) As Double, ByVal total_central_atoms As Integer, ByVal _
                                       box_length_x As Double, ByVal box_length_y As Double)
        'This sub copies the positions and velocities of the central block of a bulk crystal to surrounding blocks. The atoms in the
        'system runs from index 0 to total_central_atoms x 9 - 1. The central block atoms which are used in calculations are from index
        '0 to total_central_atoms - 1. The next atom is in block A, through to H, each containing the same number of atoms as the central
        'block.

        Dim int_start As Integer
        Dim int_end As Integer
        Dim j As Integer

        'BLOCK A
        '=======
        '(x)( )( )
        '( )(o)( )
        '( )( )( )

        int_start = total_central_atoms
        int_end = 2 * total_central_atoms - 1
        j = 0

        For i As Integer = int_start To int_end

            positions(i, 0) = positions(j, 0) - box_length_x
            positions(i, 1) = positions(j, 1) + box_length_y
            positions(i, 2) = positions(j, 2)

            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k
            j = j + 1

        Next i


        'BLOCK B
        '=======
        '( )(x)( )
        '( )(o)( )
        '( )( )( )

        int_start = 2 * total_central_atoms
        int_end = 3 * total_central_atoms - 1
        j = 0

        For i As Integer = int_start To int_end

            positions(i, 0) = positions(j, 0)
            positions(i, 1) = positions(j, 1) + box_length_y
            positions(i, 2) = positions(j, 2)

            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k

            j = j + 1

        Next i

        'BLOCK C
        '=======
        '( )( )(x)
        '( )(o)( )
        '( )( )( )

        int_start = 3 * total_central_atoms
        int_end = 4 * total_central_atoms - 1
        j = 0

        For i As Integer = int_start To int_end

            positions(i, 0) = positions(j, 0) + box_length_x
            positions(i, 1) = positions(j, 1) + box_length_y
            positions(i, 2) = positions(j, 2)

            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k

            j = j + 1

        Next i

        'BLOCK D
        '=======
        '( )( )( )
        '( )(o)(x)
        '( )( )( )

        int_start = 4 * total_central_atoms
        int_end = 5 * total_central_atoms - 1
        j = 0

        For i As Integer = int_start To int_end

            positions(i, 0) = positions(j, 0) + box_length_x
            positions(i, 1) = positions(j, 1)
            positions(i, 2) = positions(j, 2)

            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k

            j = j + 1

        Next i

        'BLOCK E
        '=======
        '( )( )( )
        '( )(o)( )
        '( )( )(x)

        int_start = 5 * total_central_atoms
        int_end = 6 * total_central_atoms - 1
        j = 0

        For i As Integer = int_start To int_end

            positions(i, 0) = positions(j, 0) + box_length_x
            positions(i, 1) = positions(j, 1) - box_length_y
            positions(i, 2) = positions(j, 2)

            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k

            j = j + 1

        Next i

        'BLOCK F
        '=======
        '( )( )( )
        '( )(o)( )
        '( )(x)( )

        int_start = 6 * total_central_atoms
        int_end = 7 * total_central_atoms - 1
        j = 0
```

```
For i As Integer = int_start To int_end

    positions(i, 0) = positions(j, 0)
    positions(i, 1) = positions(j, 1) - box_length_y
    positions(i, 2) = positions(j, 2)

    For k As Integer = 0 To 2
        velocities(i, k) = velocities(j, k)
    Next k

    j = j + 1

Next i

'BLOCK G
'=======
'( )( )( )
'( )(o)( )
'(x)( )( )

int_start = 7 * total_central_atoms
int_end = 8 * total_central_atoms - 1
j = 0

For i As Integer = int_start To int_end

    positions(i, 0) = positions(j, 0) - box_length_x
    positions(i, 1) = positions(j, 1) - box_length_y
    positions(i, 2) = positions(j, 2)

    For k As Integer = 0 To 2
        velocities(i, k) = velocities(j, k)
    Next k

    j = j + 1

Next i

'BLOCK H
'=======
'( )( )( )
'(x)(o)( )
'( )( )( )

int_start = 8 * total_central_atoms
int_end = 9 * total_central_atoms - 1
j = 0

For i As Integer = int_start To int_end

    positions(i, 0) = positions(j, 0) - box_length_x
    positions(i, 1) = positions(j, 1)
    positions(i, 2) = positions(j, 2)
```

```
            For k As Integer = 0 To 2
                velocities(i, k) = velocities(j, k)
            Next k

            j = j + 1

        Next i

    End Sub

End Module
```

## Surface Binding Energy Measurement

```
 Public Sub RunC()
        'This sub measures the surface potential energy at different points by moving an atom closer to the surface  across the surface
        'and calculating the potential.

        'to allow flexibility in x,y,z directions:
        Dim int_loop_count As Integer = 0
        Dim int_dirctn_1 As Integer
        Dim int_dirctn_2 As Integer
        Dim int_dirctn_3 As Integer
        Dim dbl_min_1 As Double
        Dim dbl_max_1 As Double
        Dim dbl_step_1 As Double
        Dim dbl_min_2 As Double
        Dim dbl_max_2 As Double
        Dim dbl_step_2 As Double
        Dim dbl_min_3 As Double
        Dim dbl_max_3 As Double
        Dim bln_bottom_surface As Boolean
        Dim bln_no_result As Boolean = False

        Dim str_dirctn_suffix As String = ""

        'To only save the minimum energy for each point
        Dim multirange_min As Double
        Dim multirange_max As Double
        Dim multirange_step As Double
        Dim dbl_exit_for As Double
        Dim dbl_result_point As Double
        Dim dbl_result_potential_energy As Double
        Dim dbl_initial_potential_energy As Double
        Dim bln_full_range As Boolean = False


        str_save_file = str_save_file & "\results\"

        If (Not System.IO.Directory.Exists(str_save_file)) Then
            System.IO.Directory.CreateDirectory(str_save_file)
        End If
```

```
'in case of drifting particles
If bln_psm_auto_drift = True Then
    autoDriftTracking(bln_bottom_surface, _
                      dbl_min_x, dbl_min_y, dbl_min_z, _
                      dbl_max_x, dbl_max_y, dbl_max_z)
End If

Do

    Dim savePotential As New SettingHandler
    savePotential.Clear()

    'check if PSM is 1D or 3D and set directions accordingly
    Range1D2Dor3D(bln_full_range, int_dirctn_1, dbl_min_1, dbl_max_1, dbl_step_1, int_dirctn_2, dbl_min_2, dbl_max_2, dbl_step_2, _
                  int_dirctn_3, dbl_min_3, dbl_max_3, bln_bottom_surface, int_loop_count, str_dirctn_suffix)

    g_dbl_atom_position(int_added_atom, int_dirctn_3) = g_dbl_atom_position(int_added_atom, int_dirctn_3) + 1000

    updateDistance(g_int_total_system_atoms)

    nearestNeighbours(g_cut_off_radius, g_dbl_periodic_boundary_x, g_dbl_periodic_boundary_y, bln_bulk, dbl_atom_radius * 1.2)
    If g_int_number_of_elements = 2 Then
        calculateLocalNnConc(dbl_atom_radius)
    End If
    nmTerm(g_int_total_system_atoms)

    g_dbl_total_potential_energy = SystemEnergyCalculation()

    dbl_initial_potential_energy = g_dbl_total_potential_energy

    'Make sure the range is not larger than the distance between atoms
    refineMultisectionRange(multirange_max, multirange_min, multirange_step, int_dirctn_3, bln_bottom_surface)

    g_dbl_atom_position(int_added_atom, int_dirctn_3) = multirange_min

    'if only selected atoms are being used, all selected atoms are used in calculations
    If int_selected_atoms = 1 Then
        setSelectAtomNearestNeighbours()
    Else
        g_int_total_selected_atoms = g_int_max_neighbours
        ReDim Preserve g_int_selected_atoms(g_int_total_selected_atoms - 1)
    End If

    int_iteration_steps = int_start_step

    time.Restart()

    '##Find minimum in range##
    '========================
    For vary_1 As Double = dbl_min_1 To dbl_max_1 Step dbl_step_1

        g_dbl_atom_position(int_added_atom, int_dirctn_1) = vary_1

        For vary_2 As Double = dbl_min_2 To dbl_max_2 Step dbl_step_2
```

```
g_dbl_atom_position(int_added_atom, int_dirctn_2) = vary_2

dbl_result_potential_energy = 0
dbl_exit_for = 0

'##Use  multisection method to find the minimum##
'================================================

int_iteration_steps = int_iteration_steps + 1

'If dbl_min_3 <> dbl_max_3 Then

dbl_max_3 = multirange_max
dbl_min_3 = multirange_min

If int_selected_atoms = 0 Then
    updateAdatomNearestNeighbours(int_dirctn_3, dbl_min_3, multirange_step, bln_bottom_surface)
End If

multiSection(dbl_max_3, dbl_min_3, dbl_result_point, dbl_result_potential_energy, dbl_initial_potential_energy, int_dirctn_3, bln_bottom_surface)


dbl_result_potential_energy = dbl_result_potential_energy - dbl_initial_potential_energy

'##Write final result##
'======================
'only write the result to file if a minimum potential was found
If bln_no_result = False Then
    Dim potential_result(3) As String
    potential_result(int_dirctn_1) = Convert.ToString(vary_1)
    potential_result(int_dirctn_2) = Convert.ToString(vary_2)
    potential_result(int_dirctn_3) = Convert.ToString(dbl_result_point)
    potential_result(3) = Convert.ToString(dbl_result_potential_energy)

    savePotential.AddLine(potential_result)
End If

outputProgress()

If dbl_step_y = 0 Then
    Exit For
End If

'Allow the program to pause if needed
Do Until bln_Pause = False
    Sleep(100)
    System.Windows.Forms.Application.DoEvents()
Loop

If bln_Stop = True Then
    savePotential.Save(str_save_file)
    System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults.csv", True)
    If bln_3D = True Then
        System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "simsettings" & str_dirctn_suffix & ".txt", True)
```

```
                            System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults" & str_dirctn_suffix & ".csv", True)
                        End If
                        Exit Sub
                    End If

                Next vary_2

                If dbl_step_x = 0 Then
                    Exit For
                End If

                savePotential.Save(str_save_file)
                System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults.csv", True)
                If bln_3D = True Then
                    System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "simsettings" & str_dirctn_suffix & ".txt", True)
                    System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults" & str_dirctn_suffix & ".csv", True)
                End If

                'Allow execution of windows program if needed
                System.Windows.Forms.Application.DoEvents()

                'Allow the program to pause if needed
                Do Until bln_Pause = False
                    Sleep(100)
                    System.Windows.Forms.Application.DoEvents()
                Loop

            Next vary_1

            savePotential.Save(str_save_file)
            System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults.csv", True)
            If bln_3D = True Then
                System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "simsettings" & str_dirctn_suffix & ".txt", True)
                System.IO.File.Copy(str_save_file & "simsettings.txt", str_save_file & "psmresults" & str_dirctn_suffix & ".csv", True)
            End If

        Loop Until bln_full_range = True

    End Sub
```

## Multi-Section Method

```
 Public Sub multiSection(ByVal high_pt As Double, ByVal low_pt As Double, ByRef result_pt As Double, ByRef result_potential As Double, _
                            ByVal start_potential As Double, ByVal dirctn As Integer, ByVal opp_side As Boolean) ', ByVal str_filename As String)
        'This function quickly finds the point closest to the minimum surface potential energy in the z-direction for a particular x,y point

        Dim minimum_pt As Double
        Dim min_val As Double = 0
        Dim min_val2 As Double = 0
        Dim high_val As Double
        Dim low_val As Double
        Dim mid_intvl As Double
        Dim mid1_val As Double
        Dim mid1_pt As Double
```

```
Dim mid2_val As Double
Dim mid2_pt As Double
Dim mid3_val As Double
Dim mid3_pt As Double
Dim choose1_pt As Double
Dim choose1_val As Double
Dim choose2_pt As Double
Dim choose2_val As Double
Dim i As Integer
Dim neg_pot As Boolean = False
Dim not_too_far As Boolean = False
Dim continue_on As Boolean = False
Dim step_intvl As Double

'check against a faraway point
Dim start_pt As Double
If opp_side = True Then
    start_pt = low_pt - 100
Else
    start_pt = high_pt + 100
End If
findSurfacePotentialForPt(start_pt, dirctn)
start_potential = g_dbl_total_potential_energy

step_intvl = high_pt - low_pt

Dim count_up As Integer = 0

'find the potential energies for the lowest and highest points in the range
Do Until (continue_on = True)

    If count_up Mod 10 = 0 Then
        low_pt = low_pt + step_intvl * 0.5
    End If

    count_up = count_up + 1


    findSurfacePotentialForPt(high_pt, dirctn)
    high_val = g_dbl_total_potential_energy

    findSurfacePotentialForPt(low_pt, dirctn)
    low_val = g_dbl_total_potential_energy

    'do a check to see that the adatom is not too far from the surface
    If high_val >= start_potential Then
        If low_val >= start_potential Then
            not_too_far = False
        Else
            not_too_far = True
        End If
    Else
        If low_val >= start_potential Then
            not_too_far = False
        Else
```

```
            not_too_far = True
        End If
    End If

    'if the adatom is too far away, move it closer a step
    If not_too_far = False Then
        step_intvl = high_pt - low_pt
        If opp_side = True Then
            high_pt = high_pt + (step_intvl)
            low_pt = low_pt + (step_intvl)
        Else
            high_pt = high_pt - (step_intvl)
            low_pt = low_pt - (step_intvl)
        End If
    End If

    'do a check to see that the adatom is not too close to another atom
    If high_val > 0 Then
        neg_pot = False
    ElseIf low_val > 0 Then
        neg_pot = False
    Else
        neg_pot = True
    End If

    'if the adatom is too close to an atom, move it away a step
    If neg_pot = False Then
        If opp_side = True Then
            high_pt = high_pt - 1
            low_pt = low_pt - 1
        Else
            high_pt = high_pt + 1
            low_pt = low_pt + 1
        End If
    End If

    'can we continue?
    If neg_pot = False Then
        continue_on = False
    ElseIf not_too_far = False Then
        continue_on = False
    Else
        continue_on = True
    End If

    'in case this loop turns into a "step-forward-step-backward-cha-cha-cha", at least the loop is not infinite
    If count_up > 20 Then
        Exit Do
    End If
    'if this ever become problematic, maybe rethink this structure?

Loop

i = 0
```

```
'Loop through a procedure that looks at the potential mid-way between the two range extremes
Do

    i = i + 1

    mid_intvl = high_pt - low_pt
    mid1_pt = high_pt + mid_intvl
    mid2_pt = low_pt - mid_intvl
    mid3_pt = low_pt + (mid_intvl / 2)

    findSurfacePotentialForPt(mid1_pt, dirctn)
    mid1_val = g_dbl_total_potential_energy

    findSurfacePotentialForPt(mid2_pt, dirctn)
    mid2_val = g_dbl_total_potential_energy

    findSurfacePotentialForPt(mid3_pt, dirctn)
    mid3_val = g_dbl_total_potential_energy

    'Find the minimum value
    min_val = Math.Min(mid1_val, min_val)
    min_val = Math.Min(mid2_val, min_val)
    min_val = Math.Min(mid3_val, min_val)
    min_val = Math.Min(low_val, min_val)
    min_val = Math.Min(high_val, min_val)

    ' outputData(0, 0)

    If min_val = mid1_val Then
        choose1_pt = mid1_pt
        choose1_val = mid1_val
        minimum_pt = mid1_pt

        'Find the next smallest value
        min_val2 = Math.Min(mid2_val, min_val2)
        min_val2 = Math.Min(mid3_val, min_val2)
        min_val2 = Math.Min(low_val, min_val2)
        min_val2 = Math.Min(high_val, min_val2)

    ElseIf min_val = mid2_val Then
        choose1_pt = mid2_pt
        choose1_val = mid2_val
        minimum_pt = mid2_pt

        'Find the next smallest value
        min_val2 = Math.Min(mid1_val, min_val2)
        min_val2 = Math.Min(mid3_val, min_val2)
        min_val2 = Math.Min(low_val, min_val2)
        min_val2 = Math.Min(high_val, min_val2)

    ElseIf min_val = mid3_val Then
        choose1_pt = mid3_pt
        choose1_val = mid3_val
        minimum_pt = mid3_pt
        'Find the next smallest value
        min_val2 = Math.Min(mid1_val, min_val2)
        min_val2 = Math.Min(mid2_val, min_val2)
        min_val2 = Math.Min(low_val, min_val2)
        min_val2 = Math.Min(high_val, min_val2)

    ElseIf min_val = low_val Then
        choose1_pt = low_pt
        choose1_val = low_val
        minimum_pt = low_pt

        'Find the next smallest value
        min_val2 = Math.Min(mid1_val, min_val2)
        min_val2 = Math.Min(mid2_val, min_val2)
        min_val2 = Math.Min(mid3_val, min_val2)
        min_val2 = Math.Min(high_val, min_val2)

    ElseIf min_val = high_val Then
        choose1_pt = high_pt
        choose1_val = high_val
        minimum_pt = high_pt

        'Find the next smallest value
        min_val2 = Math.Min(mid1_val, min_val2)
        min_val2 = Math.Min(mid2_val, min_val2)
        min_val2 = Math.Min(mid3_val, min_val2)
        min_val2 = Math.Min(low_val, min_val2)

    End If

    'Set the second smallest value
    If min_val2 = mid1_val Then
        choose2_pt = mid1_pt
        choose2_val = mid1_val
    ElseIf min_val2 = mid2_val Then
        choose2_pt = mid2_pt
        choose2_val = mid2_val
    ElseIf min_val2 = mid3_val Then
        choose2_pt = mid3_pt
        choose2_val = mid3_val
    ElseIf min_val2 = low_val Then
        choose2_pt = low_pt
        choose2_val = low_val
    ElseIf min_val2 = high_val Then
        choose2_pt = high_pt
        choose2_val = high_val
    End If


    If Math.Abs(choose1_val - choose2_val) < 0.00001 Then
        If Math.Abs(choose1_pt - choose2_pt) < 0.1 Then
            result_pt = choose1_pt
            result_potential = choose1_val
            Exit Sub
        End If
```

```
        End If

        'Limit the range depending on where the values of the potential are the smallest
        If choose1_pt > choose2_pt Then

            high_pt = choose1_pt
            high_val = choose1_val

            low_pt = choose2_pt
            low_val = choose2_val

        Else

            low_pt = choose1_pt
            low_val = choose1_val

            high_pt = choose2_pt
            high_val = choose2_val

        End If

        If i >= 50 Then
            Exit Do
        End If

    Loop

    result_pt = minimum_pt
    result_potential = min_val

End Sub
```

# Appendix B - Conferences and publications

## Conferences

C. van der Walt, H.C. Swart and J.J. Terblans, *Calculated vacancy formation energy in Cu nanoparticles*, NanoAfrica 2012: The 4th International Conference on Nanoscience and Nanotechnology, University of the Free State, Bloemfontein, South Africa, 1-4 April 2012: Poster presentation.

C. van der Walt, H.C. Swart and J.J. Terblans, *The effect of temperature on the calculated bulk vacancy formation energy in Al and Cu,* in Proceedings of SAIP2012: the 57th Annual Conference of the South African Institute of Physics, edited by Johan Janse van Rensburg (2014), pp. 245 - 251. ISBN: 978-1-77592-070-0. Available online at http://events.saip.org.za

*The calculated vacancy formation energies of Al, Ni, Cu, Pd, Ag, and Pt,* C. van der Walt, J. J. Terblans, H. C. Swart, SAIP2013: the 58th Annual Conference of the South African Institute of Physics, University of Zululand, South Africa, 8-12 July 2013: Oral presentation.

## Publications

C. van der Walt, J. J. Terblans and H. C. Swart, *Comput. Mater. Sci.* **83** (2014): 70–77.

C. van der Walt, J. J. Terblans and H. C. Swart. (2016). "Calculated nanocube vacancy formation energy and cohesion energy at 0 K," manuscript submitted for publication.

C. van der Walt, J. J. Terblans and H. C. Swart. (2016). "Temperature and surface orientation dependant calculated vacancy formation energy for Cu nanocubes," manuscript submitted for publication.

C. van der Walt, J. J. Terblans and H. C. Swart. (2016). "A study of diffusion, atom migration and segregation in Cu and Ag alloy bulk- and nanocrystals," manuscript submitted for publication.