Mechanical Characterization via Full Atomistic Simulation: Applications to Nanocrystallized Ice

A thesis presented

By

Arvand M.H. Navabi

to

The Department of Civil and Environmental Engineering

in partial fulfillment of the requirements
for the degree of

Master of Science

in the field of

Civil Engineering

Northeastern University
Boston, Massachusetts
August, 2016

Submitted to
Prof. Steven W. Cranford
Acknowledgements

I acknowledge generous support from my thesis advisor Dr. Steven W. Cranford whose encouragement and availability was crucial to this thesis and also my parents for allowing me to realize my own potential.

The simulations were made possible by LAMMPS open source program. Visualization has been carried out using the VMD visualization package.
Abstract

This work employs molecular dynamic (MD) approaches to characterize the mechanical properties of nanocrystalline materials via a full atomistic simulation using the ab initio derived ReaxFF potential. Herein, we demonstrate methods to efficiently simulate key mechanical properties (ultimate strength, stiffness, etc.) in a timely and computationally inexpensive manner. As an illustrative example, the work implements the described methodology to perform full atomistic simulation on ice as a material platform, which — due to its complex behavior and phase transitions upon pressure, heat exchange, energy transfer etc. — has long been avoided or it has been unsuccessful to ascertain its mechanical properties from a molecular perspective.

This study will in detail explain full atomistic MD methods and the particularas required to correctly simulate crystalline material systems. Tools such as the ReaxFF potential and open-source software package LAMMPS will be described alongside their fundamental theories and suggested input methods to simulate further materials, encompassing both periodic and finite crystalline models.
# Table of Contents

Acknowledgements .......................................................................................................................... 3

Abstract ............................................................................................................................................. 4

1 Introduction ..................................................................................................................................... 8

1.1 Theory and experimentation interplay ..................................................................................... 10
1.2 Overview of reactive molecular dynamics (RMD) ................................................................. 11
1.3 Approach .................................................................................................................................... 12
1.4 Thesis outline ............................................................................................................................... 13
1.5 Ice properties ............................................................................................................................... 13
    1.5.1 Mechanical behavior of ice ................................................................................................. 15
    1.5.2 Elastic modulus of ice ........................................................................................................... 15
    1.5.3 Ice tensile and compressive strength .................................................................................. 16
    1.5.4 Temperature effects ............................................................................................................. 16
    1.5.5 Strain rate effects ................................................................................................................ 17
1.6 Small-Scale vs. Large-Scale Failure ......................................................................................... 17

2 Methodology and modeling tools ................................................................................................... 18

2.1 Molecular dynamics approach ................................................................................................. 18
    2.1.1 Solving the Schrodinger’s equation .................................................................................... 20
    2.1.2 Born-Oppenheimer (BO) approximation ............................................................................ 20
    2.1.3 Approximations for the electron repulsion ........................................................................ 21
    2.1.4 Force field functional form ................................................................................................. 23
    2.1.5 Force field parameterization .............................................................................................. 24
    2.1.6 Force field variates ............................................................................................................ 25
2.2 ReaxFF reactive force field .......................................................................................................... 26
    2.2.1 Bonded interactions .......................................................................................................... 28
    2.2.2 Non-bonded interactions .................................................................................................... 32
2.3 LAMMPS ..................................................................................................................................... 33

3 LAMMPS command operations .................................................................................................... 34
3.1 Introduction to LAMMPS commands ................................................................. 34
  3.1.1 LAMMPS input script .................................................................................. 34
  3.1.2 Input script structure .................................................................................. 35
3.2 Input script and commands ............................................................................. 37
  3.2.1 Command categories .................................................................................. 38
  3.2.2 Initialization: atom_style, boundary, units ............................................. 38
  3.2.3 Atom definition: read_data, region .......................................................... 42
  3.2.4 Force fields: pair_style, Pair_coeff ......................................................... 44
  3.2.5 Settings: group, neighbor, timestep, velocity .......................................... 45
  3.2.6 Fixes: Fix, unfix ........................................................................................ 49
  3.2.7 Computes: compute .................................................................................. 50
  3.2.8 Output: dump, dump_modify, thermo ..................................................... 50
  3.2.9 Actions: run, minimize ............................................................................. 52
  3.2.10 Miscellaneous: variable .......................................................................... 54
4 Periodic modeling of materials ........................................................................ 55
  4.1 Input script part A: initialization ................................................................. 55
  4.2 Input script part B: equilibration ................................................................. 59
  4.3 Input script part C: computations ............................................................... 62
5 Finite modeling of materials ............................................................................... 66
  5.1 Input script part A: initialization ................................................................. 67
  5.2 Input script part B: equilibration ................................................................. 73
  5.3 Input script part C: computations ............................................................... 74
6 Molecular dynamics case study: mechanical characterization of crystalized ice crystallized

  6.1 Tensile and compressive strength of ice ..................................................... 79
  6.2 Ice elastic modulus ..................................................................................... 83
  6.3 Strain rate effects on ice mechanical behavior .......................................... 85
7 Conclusion ........................................................................................................ 87

Appendix .............................................................................................................. 88
Input script for periodic tension models: ..........................................................89
Input script for periodic compression models: .....................................................92
Input script for finite tension models: .................................................................94
Input script for finite compression models: ..........................................................96
Table of figures ........................................................................................................100
References ..................................................................................................................101
1 Introduction

While the inventions and technological advancements of World War Two can never be justified against the appalling loss of life, World War Two has entirely influenced today’s technological developments. Although the ultimate goal of most research was to overcome opponents, many inventions also proved to be able to improve human life and knowledge of their environment. During the war, G. Pyke proposed that an iceberg be used as an aircraft carrier in the Atlantic [1]. Although the proposed was never built, pykrete\(^1\) was invented as a result of G. Pyke’s effort in building the aircraft carrier. Pykrete has some interesting properties including its tensile strength in comparison to concrete and its high crushing strength over ice. And the new era of studies on ice as a material had begun after this discovery. Since then research on ice has had many developments, classical thermodynamics and kinetics of ice nucleation and growth has been studied. Depending on the thermal-mechanical history of ice, the structure and crystal formation has been explored. The mechanical behavior of ice from fracture propagation in the crystal structure to creep of glaciers and to the fracturing of Arctic ice has been analyzed and research has been published.

The importance of studies on ice spans from precipitation, rainmaking, polar marine transportation, oil exploration, and icing. It is a key factor in global climate and an important figure in degradation of porous materials and cold concrete. Because of the importance of ice applications, investigations on large-scale ice have long been performed, resulting in a relatively large amount of information on the mechanical behavior of ice. The states of molecular structure, its elasticity-plasticity, and

---

\(^{1}\) A frozen composite material made of approximately 14 percent different forms of wood pulp and 86 percent ice.
ice shear, tensile and compressive strength under wide varieties of temperatures have been studied alongside grain size, volume, and sample’s stress history effects on its mechanical properties.

It has only been recently that high-speed computations and powerful parallel computers are at every scientist’s hands and advancements in computer simulation methods have progressed at a significant pace. Exploiting computational power, full atomistic molecular dynamics simulation enable us to establish relations between models of condensed matter on scales from sub-Ångstroms (dealing with electrons and relations between molecules) up to mesoscopic and macroscopic scales that are used in everyday applications. Molecular dynamics simulation method (MD) was introduced in the 1950s, a short time after the Monte Carlo (MC) method. Although computation was very slow and storage space was significantly scarce, important observations were enabled at the nanoscale, including, for example, that “hard spheres crystallize at a density long before close packing has been achieved and that dynamic correlations in fluids exhibit long time tails” [2]. These were the beginnings of MD simulation achievements, and many developments are yet to come that will provide many broad applications. Applications are in many diverse fields like solid state physics to fluids and biological systems, the ultimate goal is to successfully fill the gap between small scales of electronic structure calculations to the mesoscopic scales that include coarse-grained techniques, particle dynamics, and multiscale collision dynamics.

The overall goal of MD approaches is to traditionally bridge the gap between the molecular-scale mechanical behavior of a material (here, ice) and its larger-scale mechanical behavior, i.e., from nano-to-macro. In order to adequately perform this goal, detailed knowledge of nano-scale mechanical behavior of ice is crucial. This study intends to discover further information about ice mechanical behavior under compressive and tensile stresses by full atomistic molecular dynamic simulations approaches.
1.1 Theory and experimentation interplay

There is an interesting reciprocal interaction between experimentalist and modeler: The first step is always taken by the experimentalist who determines properties and finds interesting observations while their experimental instrumentation of their measurements are in need of a more atomic resolution explanation. This is where modelers can use simulations to determine atomistic scale processes to provide further insight of the properties previously determined by experimentations or even to predict previously unknown or unclear phenomena. The experimentalist then evaluates the simulation approximations made by theory analysis and provides corroboration to them, and the loop goes on helping our knowledge on the environment that surrounds us (Figures 1 and 2).
1.2 Overview of reactive molecular dynamics (RMD)

*Ab initio* methods are computational chemistry approaches that are based on quantum chemistry (QC), these procedures alongside density function theory (DFT) and *semi*-empirical calculations have grown into a powerful tool to gain information on microscopic chemical reactions. Quantum chemistry methods have enabled scientists to investigate reactive processes and minimum-energy structures of compounds [3] [4] [5]. Nevertheless, there are significant limitations in the practice of QC methods while providing valuable theoretical guidance at electronic level simulations. Disadvantages include but are not limited to the intense computational processing for simulations, often restricting the number of atoms that can be treated due to the time and processing consumption. As a result, hybrid quantum mechanics/molecular mechanics (QM/MM) can be put into use, in which the reactive region is marked by QM while MM describes the non-reactive regions that are based on empirical potential energy functions, that is, *force fields*.

In molecular modeling, force field denotes the functional form and parameter sets used to calculate a system’s potential energy (the system can consist of atoms or coarse-grained particles). In the force field, energy function parameters are derived from both experimentations and quantum mechanics (therefore, such force fields are commonly called *empirical* potential energy functions). Hybrid QM/MM methods lack efficiency in computation and disregard the time evolution due to the effort involved in solving the QM algorithms. With the aforementioned methodologies’ deficiencies, researchers developed the molecular dynamics (MD) approach, based on derived force fields and solving Newton’s motion equation.

In MD, $\vec{F}$ is the acting force on particles computed from equation (1):

$$\vec{F} = -\vec{\nabla}V$$  \hspace{1cm} (1)
With $\nabla V$ representing the gradient of the potential energy function $V$. MD takes account of particle dynamics and the reactive processes defined in some cases by force field, making it an efficient computation method in investigating the time-evolution of large systems under well-defined thermodynamics constraints [6]. Basically, the potential energy should be determined by the use of many available approximations, employing the approximated potential energy into Equation (1), and solve it for the force ($\vec{F}$). When the force of the subject is determined, the Newton’s law of motion comes into play and results in the dynamic process of the wanted particle.

1.3 Approach

Full atomistic-scale MD simulations have been performed in this study utilizing the recent advancements of reactive force fields by deploying Reaxff, aiming to bridge sophisticated computational models to experimental findings of materials. For the sake of proper analysis and the benefit of comparison, the study has conducted a periodic modeling in addition to a finite modeling of materials via MD and LAMMPS simulation software.

The end goal of this study is to explore the molecular mechanic behavior of material under stress and learn the failure method of the material. The study will provide a comprehensive method for executing such applications on any choice of material, facilitating a generalized structure of simulation with the LAMMPS program, and thoroughly explains the necessary inputs and outputs to the simulation.

Eventually the study will apply its developed simulation on ice, explains its findings on the mechanical behavior, and provides comparison of its simulation approaches in the context of ice.
1.4 Thesis outline

In Chapter 2 the methodology is introduced, explaining the Schrodinger’s equation and the ways to solve it followed by an introduction to ReaxFF. Chapter 3 will explain the needed commands to properly simulate a model. Chapter 4 and 5 will enable the reader to understand and perform periodic and finite simulations for any material. Chapter 6 offers a case study in which ice is simulated and its mechanical properties are characterized followed by the last Chapter 7 that will conclude this study.

1.5 Ice properties

Frozen water — or ice — has 12 different crystal structures and an additional 2 amorphous states [7]. Ice structures formed in low pressure (ordinary condition) is termed ice I, which divides into two main, closely related sub-categories, Hexagonal ice Ih, which its crystal symmetry is in the shape of snowflakes and is formed by freezing water. The other form is ice, Ic, is obtained by deposition of vapors at temperatures as low as -130º C. by depositing vapor at very low temperatures and by compressing ice Ic ice type at liquid nitrogen temperature amorphous ice is formed. In addition to these elemental phases, there are clathrate hydrates that are crystallized compounds composed of massive H₂O and trapped Xe, Ar, or CH₄ within its structure. Since this research focuses on “ordinary formed ice”, ice Ih will be discussed in more detail in comparison to ice Ic.
Ice in ordinary condition, or ice Ih, is the naturally formed ice in common terrestrial form shown in Figure 3. Each H₂O molecule has four nearest neighbors formed near the vertices of a regular tetrahedron centered about the molecule of interest. Every oxygen atom is bonded to two hydrogen atoms with a strong covalent bond, in contrast to the weak hydrogen bond attaching the molecules. The sequence of molecules upon the perpendicular plane to the c-axis is …ABBAABBA…; the lattice parameters close to the melting point are \( a = 0.4523 \text{ nm} \) and \( c = 0.7367 \text{ nm} \). The temperature independent ratio of \( a/c \) is 1.628 which is close to the ideal ratio of 1.633. The packing factor of Ic ice is less than 0.34 that in result makes ice less dense than water.
In contrast, H₂O molecules in ice Ic form a cubic structure having a stacking sequence of …AABBCCAABBCC… alongside the {111} plane; the lattice parameter in cubic ice is 0.635 nm at -130º C in both types of ice, [7]. The ground state will indicate a large minimum energy state of 3.41 J/mol which stands within the third thermodynamics law, provided with infinite orientation steps.

1.5.1 Mechanical behavior of ice

The purpose of this section is to review and summarize the existing information on the mechanical properties of ice which is essential to the interpretation of subsequent results of this study obtained from the implementation of full atomistic simulation and molecular analysis of ice. Comparison of the aforementioned methods of mechanical properties determination is critical in comprehending the outcome of the atomistic analysis. For the sake of proper distinction between microscopic masses of ice in an atomistic scale with large masses of macroscopic ice, the terms atomistic and massive ice have been used accordingly throughout this study.

1.5.2 Elastic modulus of ice

Polycrystalline ice has been tested by subjecting 0.5 m diameter ice plates to biaxial bending. The results of the measurements have concluded that at -10º C the Young’s modules ranged from 9.7 to 11.2 GPa and the Poisson’s ratio was determined to be 0.29 to 0.32 [8].
1.5.3 Ice tensile and compressive strength

The strength of a massive sample of ice has not been measured by many investigators [7], over the temperature range of -10º to -20º C the tensile strength varies from 0.7 to 3.1 MPa and the compressive strength of ice varies from 5 to 25 MPa. Both with the increase of strain rate and decrease of temperature, the compressive strength of ice increases, whereas the tensile strength is relatively insensitive to these parameters. In general, an increase in grain size causes decrease in the tensile strength of ice.

In the following, the effects of temperature and strain rate have been discussed in more detail. Note that in this atomistic scale analysis, mechanical behaviors relating to ice grain size will not be accounted due to the nature of the defined circumstances of the simulation which will be thoroughly explained through Chapter 6.

1.5.4 Temperature effects

With the decrease (↓) of temperature both compressive and tensile strength of ice increase (↑). Although change in compressive strength is more prominent than in tensile strength when the sample is subjected to temperature variation. The compressive strength of ice increases by a factor of 4 from 0º to 40º C. whereas, this factor is only 1.3 over the same temperature span for tension [9]. Another study suggests that grain boundary sliding phenomena and ice dislocation dictate the temperature dependence of compressive strength in ice. And the limited dependency of strain to temperature change is related to the localization of stress-accommodating mechanisms at the tips of tensile flaws [7]. Hence, it is justifiable that in atomistic scale investigation, temperature increase fails to exhibit such dramatic variations in the strength due to the elimination of ice grains.
1.5.5 Strain rate effects

While strain rate shows effective on the compressive strength of ice, tensile strength is insensitive to strain rate over the examined range. Tensile stress-strain curves of massive ice experiments show ductile behavior at low strain rates and brittle behavior when subject to intermediate and high strain rates. Compressive stress-strain curves illustrate ductile behavior under low to intermediate strain rate while it shows brittle behavior under high strain rates. These strain rates stand in accordance with the grain boundary sliding deformation mechanism and dislocations taking place during the creeping of ice [7] [10] [11].

1.6 Small-Scale vs. Large-Scale Failure

Schulson suggests that the brittle compressive failure is independent of both the scale and material and believes kilometer-wide wing-like cracks in the Arctic sea-ice covers to be similar to millimeter-sized cracks in the laboratory. Moreover, lab tests and field observations show that fracturing and fragmentation in ice follow fractal organization [12]. In large or small scale size the fundamental physics may not change but the terminal compressive failure stresses will probably be lower in larger scales. This notion is also backed by Sanderson’s findings that large fractures fail at lower stresses than small ones [13]. Also, Dempsey has discussed that failure under tension is in fact related to size and that, in part, is the reason of large scale structures feature larger flaws [14].
2 Methodology and modeling tools

2.1 Molecular dynamics approach

Molecular dynamics is a simulation tool used to study physical movements and interactions of atoms and molecules. In many versions of molecular dynamics methods, the second Newton’s law is applied to trajectories of atoms and molecules which then solved numerically.

Combination of quantum mechanics (QM) and statistical mechanics (SM) are the main physical theory tools used in molecular dynamics simulations. Quantum mechanics is based on Schrodinger’s Equation (Equation (2)), which relates to inter-atom interactions because of its capability in describing the electrons in a molecule:

\[ \hat{H}\Psi = E\Psi \]  

Unfortunately, Schrodinger’s Equation cannot be solved analytically for systems containing more than one electron (e.g., only hydrogen!) thus, in order to solve this issue, approximations have been used to solve it for more than one electron (via so-called functionals).

Needing more information on very wide ensembles of states, caused statistical mechanics (SM) to come to play. Equation (3) is a partition function that provides probabilities of all the different states that a system may have.

\[ Z = \sum_k e^{\frac{E_k}{k_BT}} \]  

Equation (3) is a high-dimensional integral that cannot be evaluated analytically and is only possible to solve by sampling which is derived from computer simulations.
Back to Schrodinger’s Equation; all the quantum ground states are based on solving the time-independent (stationary system) of Equation (2). Where parameter $\hat{H}$ is the Hamiltonian operator which characterizes the total energy ($E$) of any given wave function and takes different forms depending on the situation. Parameter $\Psi$ is the wave function containing all the electron coordinates of the quantum system. Parameter $E$ is the total energy of state $\Psi$.

So in other words, the Hamiltonian operator ($\hat{H}$) when applied to the wave function ($\Psi$), will be proportional to the same wave function ($\Psi$) multiplied by the total energy of the system ($E$).

The various components of the Hamiltonian operator represent the energy terms that are the kinetic energy ($\hat{T}$) plus the potential energy ($\hat{V}$) of the electrons, leading to **Equation (4)**:

$$\hat{H}\Psi = E\Psi$$

$$(\hat{T} + \hat{V})\Psi = E\Psi$$ (4)

The potential energy includes interactions of the nuclei with each other termed as nuclear repulsion, interactions of the nuclei with the electrons termed electron-nuclear attraction and the interactions of the electrons with each other termed as the electron-electron repulsion. **Equation (5)** exhibits the energy terms within the kinetic and the potential energy, a more detailed version of the Schrodinger’s Equation. Note that the potential energy depends on the spatial arrangement of the particles.

$$\hat{H}\Psi = E\Psi$$ (2)

$$(\hat{T} + \hat{V})\Psi = E\Psi$$ (4)
2.1.1 Solving the Schrodinger’s equation

Schrodinger’s Equation is predominantly impossible to solve for any but the most trivial structures, approximations are used to solve the above equation, one of which that is the main focus of this study is the force fields (made possible by the Born-Oppenheimer approximation, see below), explained in detail in the following section. Before that, the time laps of electronic and nuclear terms in Equation (2) must be corrected as the following:

\[ -\sum_i \frac{v_i^2}{2} + \sum_{i \neq j} \frac{Z_i Z_j}{|R_i - R_j|} - \sum_i \frac{Z_i}{|R_i - \hat{r}_i|} + \sum_{i \neq j} \frac{1}{|r_i - r_j|} \Psi(r_i; R_i) = E(R_i) \Psi(r_i; R_i) \] (5)

2.1.2 Born-Oppenheimer (BO) approximation

As Equation (5) suggests, the Schrodinger Equation consists of both the nuclear and the electron motions. In order to solve the equation, the Born-Oppenheimer (BO) approximation is introduced which assumes that the motion of atomic nuclei and the electrons in a molecule are distinguishable, allowing the wave function (\( \Psi \)) of the molecule to be broken into two terms, electronic and nuclear components, or Equation (6):

\[ \Psi_{total} = \psi_{electronic} \times \psi_{nuclear} \] (6)
This approximation enables solving the molecular wave-functions for larger molecules in two less complicated steps, first the Schrodinger’s equation is solved depending only on $\psi_{\text{electronic}}$ while fixing the nuclei to a certain setting, often the equilibrium state of the nuclei. Then, the second step is to solve for the Schrodinger’s equation containing only the nuclei term $\psi_{\text{nuclear}}$.

2.1.3 Approximations for the electron repulsion

The electron repulsion term in the Schrodinger’s Equation (5) cannot be solved by the usual known methods used in differential equations and has to be approximated with the most precise possible approach. From the early days of quantum mechanics researchers have suggested different approaches in order to solve the aforementioned equation for treating many-electron systems. Below, three approximations with somewhat the same final result are briefly introduced, approach 3 is the main method used in this study and will be explained thoroughly along with its benefits.

**Approach 1**: employing approximate wave-function forms (quantum chemistry). The Hartee-Fock (HF) method, approximates the wave-function and the energy of a many-body stationary system. The HF method solves the individual energy of each electron by introducing a modified version of the Schrodinger Equation, resulting in an approximate solution to the equation.

**Approach 2**: employing electron density functional theory (DFT) which uses electron density functionals to determine the properties of a many-electron system. The ground-state electron density, contains the same information as a ground-state wave-function, hence electron density can provide the same information regarding the state of a system as a wave-function.
**Approach 3:** approximation of Schrödinger’s Equation with force fields, involves empirical functions derived from experimental work and quantum mechanical calculations. Force fields do not treat the electron degrees of freedom explicitly, and only provide a sense of atomic coordinates through describing the potential energy. Because the potential energy’s dependence on the atom positions, atomic coordinates can also be calculated from the potential energy.

If solving the electron degrees of freedom, the potential energy of the system is determined by using a wave-function and if the potential energy is determined by molecular dynamics, the process is done by an empirical function termed force field.

![Figure 4. Electron repulsion approximations. Time efficiency versus number of atoms that can be computed](image-url)
2.1.4 Force field functional form

Force fields fundamentally define the total potential energy into two components, the bonded term for defining the interactions of atoms that are linked by covalent bonds. On the other hand, the non-bonded term defines the long-range electrostatic and van der Waals forces. The general form of force fields is described in Equation 7, and specific decompositions of force field terms may be subjected to change depending on the type of the force field.

\[ E_{\text{total}} = E_{\text{bonded}} + E_{\text{non-bonded}} \]  \hspace{1cm} (7)

The term \( E_{\text{bonded}} \) is regarded as the covalent contributor, whereas the term \( E_{\text{non-bonded}} \) is regarded as the non-covalent contributor to the total energy.

The components of the non-bonded and the bonded potential energies are as described in Equation 8 and 9:

\[ E_{\text{bonded}} = E_{\text{bond}} + E_{\text{angle}} + E_{\text{dihedral}} \]  \hspace{1cm} (8)

\[ E_{\text{non-bonded}} = E_{\text{electrostatic}} + E_{\text{van der Waals}} \]  \hspace{1cm} (9)

Bond and angle energies are incapable of accounting for bond breaking this is due to the fact that \( E_{\text{non-bonded}} \) and \( E_{\text{bonded}} \) are commonly modeled by quadratic energy functions, which makes their use limited to specific applications. The non-bonded term is more computationally expensive which perhaps can be reduced to only justifying pairwise energies, in accordance to the use of the force field. The electrostatic term is computed with the Coulomb’s law and the van der Waals term is traditionally computed using the Lennard-Jones potential.
Note that for high stretch covalent bonds, a more computationally expensive solution can be used, called the Morse potential. The functional form of the dihedral energy significantly varies in accordance to the application of the force field.

Additionally, terms are added to reinforce the accuracy of the force field, such as improper torsional terms in order to enforce the planarity of aromatic rings, also terms may be added to account for more conjugated systems.

2.1.5 Force field parameterization

From the previous section, it is apparent that force fields define a functional form of the potential energies in an attempt to solving for the total potential energy. But the functional forms of the potential energies perform alongside other empirical terms called the parameter sets. A set of unique parameters are defined for different types of atoms, chemical bonds, dihedral angles and so on. A typical parameter set, associates values for atomic mass, van der Waals radius, partial charge for each atom, equilibrium values of bond lengths. Bond angles and dihedral angles for pairs, triplets, and quadruplets of bonded atoms, and values corresponding to the effective spring constant for each potential.

It is important to define parameter sets and functional forms to be self-consistent. In other words, an interatomic potential function’s parameter must never be used in conjunction with another potential function.
2.1.6 Force field variates

Force fields vary by their application, and are designed differently in order to serve their purposes. Some are designed to perform analysis on hydrocarbons and other small organic molecules. Some study the structures and vibration of molecules or crystals. A force field called ECEPP was specifically designed to model proteins. Moreover, AMBER and many others are mainly developed for molecular dynamics of macromolecules. ReaxFF is one of the many force fields designed to model its own specific purpose. ReaxFF, in contrast to many conventional force fields is able to model chemical reactions, this improvement is made possible with the ability of the ReaxFF force field to avoid explicit bonds in favor of bond orders. Bond order is another empirical interatomic potential that is used in many molecular mechanics force fields which can describe several different bonding states of an atom with a single type of parameter, and thus the capability of describing chemical reactions.

In the following, some forms of force fields are briefly mentioned in term of their category.

2.1.6.1 Classical force fields:

Classical force fields include AMBER which is widely used to simulate proteins and DNA. Chemistry at HARvard Molecular Mechanics or CHARMM is used for small molecules and macromolecules. ECEPP designed for polypeptide molecules and many more classical force fields.
2.1.6.2 Polarizable force fields:

An important part of electrostatic modeling is the ability to correctly consider the polarizability of a system, polarizable force fields fall into traps when used in macroscopic models, but there is no such traps when employed for simulating microscopic models.

One of the most advanced force fields in MD simulation is the X-Pol, capable of simulating solvated protein in water consisting of 15000 atoms with periodic boundary conditions in 3200 steps and in 24 hours, running on one processor [15].

2.1.6.3 Reactive force fields:

ReaxFF is the atomistic-scale computational method of choice to the date in which this study is performed. ReaxFF is fast and employs bond order potentials that will enable the chemical reaction analysis of a system. EVB (empirical valence bond) is also a reliable choice of simulation applicable to condensed phases and enzymes. Finally for simulating water and acids the ReaxFF is used.

There are certainly many other force fields and force field categories that fall short in the context of this study, herein, they are not mentioned.

2.2 ReaxFF reactive force field

ReaxFF is a powerful molecular dynamics simulation tool that employs empirical interatomic potentials and requires significantly less computations in comparison to classical principals, and enables the simulation to describe dynamic processes over longer time-steps, it implicitly solves for the occurring chemical reactions with using bond-order formalisms. This section is intended
to explain the methodology of ReaxFF since it is the method used in this study in order to analyze the nano-scale mechanical behavior of materials.

As mentioned above, ReaxFF uses the bond-order formalism to describe the interactions of atoms with each other [16]. ReaxFF similar to empirical interatomic approximations called force fields, divides the system energy into various partial energies. As mentioned in Eq. 7, the total energy is divided into two individual expressions that are associated with the bonded and non-bonded potential energies. In the case of ReaxFF, the components of Eq. 7 is as the following (Eq. 10 and 11):

\[ E_{total} = E_{bonded} + E_{non-bonded} \]  \hspace{1cm} (7)

\[ E_{bonded} = E_{bond} + E_{over} + E_{under} + E_{lp} + E_{val} + E_{tor} \]  \hspace{1cm} (10)

\[ E_{non-bonded} = E_{van
der Waals} + E_{Coulomb} \]  \hspace{1cm} (11)

Such that:

\[ E_{system} = E_{bond} + E_{over} + E_{under} + E_{lp} + E_{val} + E_{tor} + E_{van
der Waals} + E_{Coulomb} \]  \hspace{1cm} (12)

Where the partial contributors to the total energy of the system are the bonded terms: bond, over coordination penalty, under coordination stability, lone pair, valence angle and torsion. The non-bonding terms are Coulomb and van der Waals energies [17] [18].
2.2.1 Bonded interactions

2.2.1.1 Calculating bond order

The initial positioning of the atoms within the system are recorded, then, the first step is to establish the bond orders between all the atoms. As a fundamental assumption the ReaxFF determines the bond order \( BO'_{ij} \) between a pair of atoms directly from the interatomic distance \( r_{ij} \) of the pair. **Eq. 13** is an example of the bond order for carbon-carbon bond. This study does not cover any \( C-C \) relationships, however the above example is, namely, in order to better comprehend the function of the bond order calculation in ReaxFF, thus it can be different for other materials [16].

\[
BO'_{ij} = \exp \left[ p_{bo,1} \left( \frac{r_{ij}}{r_0} \right)^{p_{bo,2}} \right] + \exp \left[ p_{bo,3} \left( \frac{r_{ij}}{r_0} \right)^{p_{bo,4}} \right] + \exp \left[ p_{bo,5} \left( \frac{r_{ij}}{r_0} \right)^{p_{bo,6}} \right]
\]

(13)

The first term presenting the single bond, the second term describing the double bond (pi bond), and the last term explaining the triple bond or the double pi bond. Basically, **Eq. 13** has retrieves the atom pair distances and outputs the associated bond order of the two atoms. Each bonding term, \( p \), and bonding equilibrium distance, \( r_0 \), have been set to values derived from quantum mechanical approaches for the atoms in which their distances from one another are at \( r_{ij} \). Shown in **Figure 4**, the bond order of the carbon-carbon is determined by the interatomic distances.
2.2.1.2 Bond order adjustment for over-coordination (over-coordination penalty energy)

The bond order formalism described in the previous section, allows unrealistic interactions between the long-range second nearest neighbors of the subjected atom. This is due to the capability of the bond order calculation that enables long range bond modeling. This type of bonding in bond order force fields causes unrealistic behavior when intact molecules are simulated [16]. For example, when a carbon atom is involved in a bond order of 4 or more, realistically, these long range interactions caused by the bond order calculation should be corrected and reduced. Although when a carbon atom has less than 4 bonds, the bond order formalism’s inclusion of long range interactions are in fact correct and result in a more accurate modeling. For the full detailed equation regarding the over-coordination penalty energy refer to [16].
Figure 6. (a) Effect of the bond order correction on C-C and C-H bond orders. (b) Effects of shortening of the C-C bond to 1.0 Å on the relaxed C-H bond as calculated DFT on the left and ReaxFF on the right [16].

2.2.1.3 Angle and torsion interactions

The angle and torsion descriptions in ReaxFF are bond dependent parameters, whereas in non-reactive force fields, the angle and torsion definitions are regardless of the bond strength or weakness. In ReaxFF the separated atom from its initial molecule, due to the torsion and angle specifications, absorbs some of the breaking force into its bonds and therefore, less force is exerted into the rest of the remaining molecule, enabling the process to better resemble the reality.

As the following equation (Eq. 14) suggests, the angle energy is related to both bond orders ($BO_1, BO_2$) that form the three atoms, with an angle of $\phi$. The angular parameter set $\lambda$ is to achieve an agreement with the quantum values. The depth and width of the angular potential is accounted for with the use of $k_a$ and $k_b$. The angle at equilibrium is the $\phi_0$. 

\[\Sigma BO_{C}=7.05\]
\[\Sigma BO_{H}=1.37\]
\[\Sigma BO_{C}=4.58\]
\[\Sigma BO_{H}=0.97\]
\[ E_{angle} = [1 - \exp(\lambda BO_1^3)].[1 - \exp(\lambda BO_2^3)].\{k_a - k_b . \exp(-k_b . (\phi - \phi_0)²)\} \tag{14} \]

The general formation of the angle and torsion functions are as below:

\[ E_{angle} = f(\phi_{ijk}, BO_{ij}, BO_{jl}) \tag{15} \]

\[ E_{torsion} = f(\phi_{ijkl}, BO_{ij}, BO_{jk}, BO_{kj}) \tag{16} \]

The angle and torsion energy terms must \( \rightarrow 0 \) if \( BO \rightarrow 0 \) [19].

2.2.1.4 Lone pair electrons, conjugation

The creation or reaction of lone-pair electrons should be assigned as an energy term, the \( E_{lone-pair} \) corresponds to an energy penalty for having too many lone pairs on an atom. Conjugated systems have added stabilization. The \( E_{conj} \) has maximum contribution when successive bonds have a bond-order values of 1.5.

2.2.1.5 Hydrogen bonding

Hydrogen bonding is a highly significant part of the total energy calculations, especially when ReaxFF is used for the \( H_2O \) structure of ice and determining its behavior in fracture propagation and failure due to stress. Hydrogen bonds are calculated between atoms known to form H-bonds. The H-bond energy is a function of the bond order of H-bond forming atom \( (X) \), the distance between the hydrogen and the other atom termed as \( (Y) \) and the X-H-Y angle, as the following equation puts it:

\[ E_H = f(BO_{XH}, R_{Y...H}, \theta_{XH...Y}) \tag{17} \]
2.2.2 Non-bonded interactions

In non-reactive force fields the Coulomb and the van der Waals forces are typically only accounted for the atoms that do not share a bond together, whereas in ReaxFF environment the forces are calculated for all of the existing atom pairs independent of their bonding. As for a correction to this calculation a shielding method is used to calculate the Coulomb and van der Waals energy.

For example, the Coulomb equation is shown in Eq. 18.

\[ E = \frac{q_i q_j}{r_{ij}^3 + \left( \frac{1}{\gamma_{ij}} \right)^{3/2}} \]  

(18)

Where \( q_i \) and \( q_j \) are the two charges for the H-bond forming atoms, \( r_{ij} \) is the interatomic distance, and \( \gamma_{ij} \) is the shielding parameter.

Figure 7. Shielding included for both Coulomb and van der Waals in order to avoid excessive interaction between atoms sharing bond and/or bond angle [20]
2.2.2.1 ReaxFF summary

The ReaxFF is capable of simulating chemical reactions for a wide range of materials significantly faster than \textit{ab initio} and semi-empirical methods. Accuracy of ReaxFF is similar to semi-empirical methods and it can be used in reactive hydrocarbons, CHNO explosives, silicon oxides, etc. and simulation sizes are up to a half million atoms, for this type of simulation the main limitation is governed by the size of reaction training sets.

2.3 LAMMPS

LAMMPS, or large-scale atomic/molecular massively parallel simulator is a free and open source MD program designed by the \textbf{Sandia National Laboratories}, this program employs MPI for parallel communication.
3 LAMMPS command operations

This section is intended to thoroughly explain the commands and input scripts in which materials with crystalline structure can be modeled by classical molecular dynamic simulation method using the LAMMPS program.

The following simulation method presented in this chapter defines a finite modeling of materials. Finite modeling treats the subject material in a way that the surface energy would relate to all of the molecules in the specimen, regardless of the molecules position. In other words, both the surface molecules and the bulk molecules in the specimen will be treated as surface molecules with the excess surface energy that is associated to surface molecules. This treatment is made possible with the input script explained in this section.

3.1 Introduction to LAMMPS commands

This section clarifies the general format of the LAMMPS simulation input script commands.

3.1.1 LAMMPS input script

LAMMPS reads the commands from the text file input script and executes accordingly. LAMMPS reads one line at a time and exits when the input script is ended. Each command causes the simulator to take a specific action, it may set an internal variable, read in a file, or run a simulation. Many commands within the simulator have predefined settings as default and may be changed if necessary. The reader is directed to the most current version of LAMMPS for any discrepancies.
It is important to note that LAMMPS does not read the entire input script and then perform the simulation. Rather, it reads line by line and reacts according to each line, so the sequence of the lines may result in different executions.

Each non-blank line in the input script is treated as a command. LAMMPS commands are case sensitive. Command names are lower-case, as are specified command arguments. Upper case letters may be used in file names or user-chosen ID strings.

### 3.1.2 Input script structure

This section describes the structure of a typical LAMMPS input script, stating their overall role in the script and the necessary sequence in which the command types are written (see appendix for input scripts).

A LAMMPS input script typically has four sections:

1. Initialization
2. Atom definition
3. Settings
4. Run a simulation

The last two parts can be repeated as many times as desired. I.e., run a simulation, change some settings, run some more, etc. Each of the four parts are now described in more detail. As a reminder it should be noted that almost all the commands need only be used if a non-default value is desired.
3.1.2.1 Initialization

Before atoms are created or read-in from a file, necessary parameter sets have to be defined, this is a part of the initialization.

Relevant commands are as the following:

*units, dimension, newton, processors, boundary, atom_style, atom_modify.*

The following are the commands that tell LAMMPS what kind of force fields are being used:

*pair_style, bond_style, angle_style, dihedral_style, improper_style.*

3.1.2.2 Atom definition

There are three ways to define atoms, read them in from a data set via the read_data command or to restart file via the read_restart command. These files can contain molecular topology information or create atoms on a lattice with no molecular topology, using the following commands:

*Lattice, region, create_box, create_atoms.*

The complete defined atoms can be duplicated with the replicate command.

3.1.2.3 Settings

Once the two previous steps, parameter setup and the atom and molecular topology is defined, a variety of settings can be specified: force field coefficients, simulation parameters, output options, etc.
Force field coefficients are set by these commands: \textit{pair\_coeff}, \textit{bond\_coeff}, \textit{angel\_coeff}, \textit{dihedral\_coeff}, \textit{improper\_coeff}, \textit{and more}.

Many simulation parameters are set by the following commands:

\textit{neighbor}, \textit{neigh\_modify}, \textit{group}, \textit{timestep}, \textit{reset\_timestep}, \textit{run\_style}, \textit{min\_style}, \textit{min\_modify}.

\textbf{Fixes} have a variety of boundary conditions, time integration, and diagnostic options. As the documentation itself suggests, the \textit{fix} command comes in many flavors.

Many computations could be added while the simulation is in progress using the compute, \textit{compute\_modify}, and \textit{variable} commands.

Output options are set by the \textit{thermo}, \textit{dump}, and \textit{restart} commands.

3.1.2.4 Run a simulation

A MD simulation is run with the \textit{run} command. Energy minimization is executed by using the \textit{minimize} command, while a parallel tampering (replica-exchange) simulation can be run using the \textit{temper} command, as other examples.

3.2 Input script and commands

This section presents the input script lines to the LAMMPS, lines of the input script consist of commands. LAMMPS reads the input from the first to the last line, then the simulation ends with the last line. Here we will introduce the applications of the commands used in this study and categorize them according to their implications. These categories alongside the commands are as below.
3.2.1 Command categories

Initialization: *atom_style, boundary, units*

Atom definition: *read_data, region*

Force fields: *pair_style, Pair_coeff*

Settings: *group, neighbor, timestep, velocity*

Fixes: *fix, unfix*

Computes: *compute*

Output: *dump, dump_modify, thermo*

Actions: *run, minimize*

Miscellaneous: *variable*

Although the commands in the LAMMPS documentation exceeds from the above list, but we can rely on knowing the aforementioned commands in order to accomplish our study on characterization of the mechanical behavior of materials.

3.2.2 Initialization: *atom_style, boundary, units*

3.2.2.1 units command

Description
The *unit* command sets the style of units used for a simulation. It determines the units of all quantities specified in the input script and data file, as well as quantities output to the screen, log file, and dump files. Typically, this command is used at the very beginning of an input script.

The first step as mentioned previously is the initialization step. The first initialization used in this script is the *units* commands, followed by the style of the command. This study has designated the *real* style for this command in order for the simulation to associate with the desired units shown below.

**Syntax:**

```
units style
```

- style = *lj* or *real* or *metal* or *si* or *cgc* or *electron* or *micro* or *nano*

**Example:**

```
units real
```

For the style *real*, the following units will be designated to the inputs and outputs:

- mass = grams/mole
- distance = Angstroms
- time = femtoseconds
- energy = Kcal/mole
- velocity = Angstroms/femtosecond
- force = Kcal/mole-Angstrom
- torque = Kcal/mole
• temperature = Kelvin
• pressure = atmospheres
• dynamic viscosity = Poise
• charge = multiple of electron charge (1.0 is a proton)
• dipole = charge*Angstroms
• electric field = volts/Angstrom
• density = gram/cm^dim

3.2.2.2  atom_style command

Description

This command describes the style of the atoms used in the simulation alongside the attributes associated with the atoms. The command should be used prior to the setup of the simulation via a read_data or read_restart and creat_box.

Syntax:

```
Atom_style style args
```

- Style = angle or atomic or body or bond or charge or dipole or electron or full etc.

Example:

```
Atom_style full
```
The choice of style affects what quantities are stored by each atom, the quantities that communicate between processors to make the computation possible. And the quantities listed in the data file read by the read_data command (explained later).

In this study the atom style is set to full. The attributes for the full style are molecular systems consisting charges, which will be well suited for molecular analysis of material’s mechanical behavior, although according to the documentation this style is usually used for bio-molecules.

3.2.2.3  *boundary* command

Description

The boundary command sets the style of the users choosing for the global simulation box in each dimension. If a single letter is assigned as one of the three style slots, the same style will apply to both the upper and lower face of the box. If two styles are chosen for one style slot, the first style will assign to the lower face and the second will be assigned to the upper face of the box. Using this command will enable defining the lower and upper faces of all three directions (x,y,z).

Syntax:

```
Boundary x y z
```

- x,y,z = p or s or f or m, one or two letters
  - *p* is periodic
  - *f* is non-periodic and fixed
  - *s* is non-periodic and shrink-wrapped
  - *m* is non-periodic and shrink-wrapped with a minimum value
Example:

```
Boundary p p p
```

Style p means the box is set to be periodic, meaning that particles interact across the boundary and can exit one end of the box and re-enter the other. The p style must be applied to both faces of a dimension or it will be a contradiction to the definition of p style. This command should be used prior to the `read_data` command.

### 3.2.3 Atom definition: `read_data, region`

#### 3.2.3.1 `read_data` command

**Description**

The `read_data` command reads in data files that contain the necessary information for LAMMPS.

**Syntax:**

```
Read_data filename
```

**Example:**

```
Read_data filename
```
3.2.3.2 region command

The region command defines a geometric region or in other word to define a box boundary for the system. Values for lo/hi block style and other styles can specify the extent of the boundary. For example, if the style is set as block and the args are to be numbers, the boundary box will be determined according to the numbers entered. If the arguments are set as EDGE or INF it means they extend all the way to the global simulation box boundary.

Syntax:

Region ID style args keyword arg ...

- ID = user-assigned name for the region
- style
  = delete or block or cone or cylinder or plane or prism or sphere or union or intersect
- zero or more keyword/arg pairs may be appended
- keyword = side or units or move or rotate

block args = xlo xhi yli yhi zlo zhi

xlo, xhi, yli, yhi, zlo, zhi = bounds of block in all dimensions (distance units)

For style block, the lattice spacing in dimension x is applied to xlo and xhi, similarly the spacings in dimensions y,z are applied to ylo/yhi and zlo/zhi.

Example syntax:

Region 2 block INF INF INF INF INF -12. units box
3.2.4 Force fields: \textit{pair\_style, Pair\_coeff}

3.2.4.1 \textit{pair\_style} and \textit{Pair\_coeff} command

Description

The \textit{pair\_style} command sets the formulas that LAMMPS uses to calculate the pairwise interactions. The pairwise interaction act upon pairs within a certain cutoff distance. Pairwise force fields encompass a variety of interactions. The \textit{pair\_coeff} command sets the associated coefficients of a pair style, this can also be done with reading in a file.

Syntax:

\begin{verbatim}
pair\_style style args
\end{verbatim}

\textit{style} = \textit{none, hybrid, zero, beck} etc.

\textit{args} = arguments used by a particular style

Syntax: set style as \textit{reax/c}:

\begin{verbatim}
pair\_style reax/c cfile keyword value keyword value ... ...
\end{verbatim}

Example:

\begin{verbatim}
pair\_style reax/c NULL safezone 2 minicap 100
\end{verbatim}

cfile = \textit{NULL} or name of a control file

zero or more keyword/value pairs may be appended
safezone = factor used for array allocation

minicap = minimum size for array allocation

The reax/c style computes the ReaxFF potential of van Duin, Goddard et al. [16]. ReaxFF uses distance-dependent bond-order functions to represent the contributions of chemical bonding to the potential energy.

The cfile setting or the control file is specified as NULL, in which the default settings are used, and the purpose of this simulation did not necessitate settings other than the default option. The ReaxFF parameter files were created using a charge equilibration (QEq) model in order to manage the electrostatic interactions, therefore LAMMPS requires the use of fix qeq/reax command when simulating a ReaxFF model for it to be able to equilibrate charge each timestep.

In order to allocate reax/c arrays, the optional safezone and minicap keywords are employed in the used syntax. With increasing the value of aforementioned keywords, memory problems could be avoided.

To set the pair_coeff refer to the LAMMPS potentials directory, some pair styles use tabulated potentials as input, this case is usually for many-body potentials.

3.2.5 Settings: *group, neighbor, timestep, velocity*

3.2.5.1 *group* command

Identify a collection of atoms as belonging to a group. *group* can be used with region as its style, meaning the region style puts all atoms in the region volume into the group. If the region is used
as a style of the group, the groups are static one-time assignment. The atoms remain the same, (assigned or not assigned) to the group even if they leave their region volume.

Syntax:

<table>
<thead>
<tr>
<th>Group ID style args</th>
</tr>
</thead>
</table>

- ID = user-defined name of the group
- style = delete or region or type or id or molecule or union

Syntax example:

| group bot region 2 |

Where bot is the user defined ID for the group, the region is the style and 2 is the ID of the region assigned to the group. The same example can be modified to union as the style, so the args after the union would unite the IDs to one group.

3.2.5.2 neighbor command

Description

The neighbor command sets parameters that contribute to defining the pairwise neighbor lists. The pairwise neighbor list stores atoms within a neighbor cutoff distance, the cutoff distance is equal to the force cutoff plus the skin distance. With larger skin distance, the less often neighbor lists need to be built but more pairs must be checked for possible force interactions in every timestep. The skin distance also determines how often the atoms migrate to a new processor.
The style value selects what algorithm is used to build the list. The *bin* style creates the list by binning which is an operation that scales linearly.

Syntax:

```
Neighbor skin style
```

Example:

```
Neighbor 0.5 bin
```

3.2.5.3 *timestep* command

Description

The *timestep* command sets the time-step size for the subsequent MD simulation to be executed. By defining the units of your simulation *via* the units command, the unit in which the *timestep* is defined will vary. For this script the unit was chosen to be *real*, indicating all units of time in femtoseconds (refer to appendix).

Syntax:

```
Timestep dt
```

- Dt = timestep size (unit is according to units command)

Example:

```
Timestep 0.1
```
3.2.5.4 *velocity* command

The *velocity* commands will set or modify the velocities of a group of atoms in one of several styles.

Syntax:

```
velocity group-ID style args keywords value
```

- **group-ID** = ID of group of atoms whose velocity will be changed
- **style** = *create* or *set* or *scale* or *ramp* or *zero*
- **zero** or more keyword/value pairs may be appended
- **keyword** = *dist* or *sum* or *mom* or *rot* or *temp* or *bias* or *loop* or *units*

Syntax example:

```
velocity fix set 0.0 0.0 0.0 units box
```

- **Set args** = vx vy vz
  
  vx vy vz = velocity value or *NULL* (velocity units)

The set style determines the velocities of all the group atoms to the specified values entered as args.
3.2.6 Fixes: Fix, unfix

3.2.6.1 fix command

Description

Set a fix that will be applied to a group of atoms in which the user specifies. fix is any command that applies any operation to the system while timestepping or minimization. Many types of style may apply to fix, for example nve and npt may apply.

Syntax:

```
Fix ID group-ID style args
```

- ID = user-assigned name for the fix
- group-ID = ID of the group of atoms to apply the fix to
- style = one of a long list of possible style names (refer to LAMMPS documentations)
- args = arguments used by a particular style

3.2.6.2 unfix command

Description

Deletes the fix previously defined with a fix command or to put a stop to the fix that is running.

Syntax:

```
Unfix fix-ID
```

- fix-ID = ID of a previously defined fix
3.2.7 Computes: \textit{compute}

3.2.7.1 \textit{compute} command

The \textit{compute} command defines a certain computation to be executed on a group of atoms specified by the user. Quantities calculated by \textit{compute} command are instantaneous values which are calculated from information about atoms on the current timestep or iteration.

Syntax:

\begin{verbatim}
compute ID group-ID style args
\end{verbatim}

- ID = user-assigned name for the computation
- group-ID = ID of the group of atoms to perform the computation on
- style = one of a list of possible style names
- args = arguments used by a particular style

3.2.8 Output: \textit{dump, dump_modify, thermo}

3.2.8.1 \textit{dump} command

Dump a snapshot of atom quantities to one or more files every N timestep in one of several styles.

Syntax:

\begin{verbatim}
Dump ID group-id style N file args
\end{verbatim}

- ID = user-assigned name for the dump
- group-ID = ID of the group of atoms to be dumped
- style = atom or atom/gz or atom/mpiio or dcd or xtc or xyz etc.
- timesteps
- file = name of file to write dump info to
- args = list of arguments for a particular style

`dump_modify` command is also used after the above syntax, in order to modify the parameters that have previously been defined in the dump command section.

Syntax:

```
dump_modify dump-ID keyword values ...
```

- `dump-ID` = ID of dump to modify
- one or more keyword/value pairs may be appended
- these keywords apply to various `dump` styles
- keyword = `append` or `buffer` or `element` or `every`

Example:

```
dump ID group-ID xyz 500 filename.xyz

dump_modify ID-group element H O
```
3.2.8.2 *thermo* command

The *thermo* command will compute and finally print out the thermodynamic information of the system on every N timesteps, in addition it will print the beginning thermodynamic state and the end state.

Syntax:

```
Thermo N
```

N = output thermodynamics every N timesteps

N can be a variable

3.2.9 Actions: *run, minimize*

3.2.9.1 *run* command

Description

The *run* command runs the specific *fix* for a specified number of timesteps.

Syntax:

```
run N keyword values ...
```

- N = # of timesteps
- zero or more keyword/value pairs may be appended
- keyword = *upto* or *start* or *stop* or *pre* or *post* or *every*
3.2.9.2 minimize command

The minimize command performs an energy minimization to the system by iteratively adjusting atom coordinates. The command will terminate the iterations when one of the stopping criteria (explained further in this section) is satisfied at which the system configuration is at minimal potential energy.

Syntax:

```
Minimize etol ftol maxiter maxeval
```

- etol = stopping tolerance for energy (unit less)
- ftol = stopping tolerance for force (force units)
- maxiter = max iterations of minimizer
- maxeval = max number of force/energy evaluations

Example:

```
Minimize 0.0 1.0e-8 1000 10000
```

The minimization procedure stops if any of several criteria are met:

- the change in energy between outer iterations is less than etol
- the 2-norm (length) of the global force vector is less than the ftol
- the line search fails because the step distance backtracks to 0.0
- the number of outer iterations or timesteps exceeds maxiter
- the number of total force evaluations exceeds maxeval
3.2.10 Miscellaneous: variable

3.2.10.1 variable command

With the variable command, it is possible to assign string(s) to a variable name for evaluation later in the input script or during a simulation.

Syntax:

```
Variable name style args ...
```

- name = name of variable to define
- style = delete or index or loop or getenv or file or atomfile or python or equal

Syntax example:

```
variable total_pe equal c_pot0
```

Where total_pe is the name, style is set to equal and the argument or c_pot0 is the formula that contains the numbers that will be assigned to the variable named total_pe
4 Periodic modeling of materials

In this chapter we discuss the periodic modeling of the subject material. As the commands were thoroughly explained in Chapter 3, they are used here in a way that the model acts as if the simulation is acting upon a very large material sample (e.g., infinite).

The first step for the simulation is to set the initial conditions such as the units, timesteps, and boundary conditions of the model (not the simulation box), the force field, and the output file of the simulation. The second section is to minimize or to bring the system into equilibrium state with the minimum possible system energy, and analyze the system’s thermodynamics. The last section of the input script will focus on the computation goals of this study, for example the commands in which it will cause stress and strain, commands which will perform CNA (Common Neighbor Analysis) and Centro-Symmetry Parameter calculations will be applied.

The aforementioned steps will be explained in the following 3 sections.

4.1 Input script part A: initialization

In this step the units are set, the atom coordinates are inputted to the script and boundary conditions are fixed. The force field will be chosen to act as the ReaxFF and the final (x,y,z) output is defined.

Line 1:

Using the *units* command we will set the *units* as below, using the *real* style (Input script 1, line 1):

- mass = grams/mole
- distance = Angstroms
- time = femtoseconds
- energy = Kcal/mole
- velocity = Angstroms/femtosecond
- force = Kcal/mole-Angstrom
- torque = Kcal/mole
- temperature = Kelvin
- pressure = atmospheres
- dynamic viscosity = Poise
- charge = multiple of electron charge (1.0 is a proton)
- dipole = charge*Angstroms
- electric field = volts/Angstrom
- density = gram/cm^dim

Resulting in the inputs and the outputs to the software to have known units most importantly the forces, energies, time and distance.

Line 2:

Before reading in the atom coordinates the atoms must be defined. In this study the atom style is set to full (see input script 1, line 2). The attributes for the full style are molecular systems consisting of charges which will be well suited for molecular analysis of material’s mechanical behavior, although according to the documentation, this style is usually used for bio-molecules

Line 3:
The time-step of the simulation is set via the *timestep* command, herein the time-steps size will better exhibit the simulation variables and changes in partials of femtoseconds. The *units* command has set the simulation time unit to femtoseconds in order for the reactions to manifest a smooth transition in the creation and the breaking of bonds, complying with the same reason, the time size is set to 0.1 of a femtosecond. The swiftest chemical reaction is 15 femtoseconds, the duration of a vibration of the atoms in an iodine molecule is 300 femtoseconds which shows that the decision of the time size is in reasonable accordance to the mentioned time periods.

Line 4:

The global simulation boundary is set with the boundary command. Here the boundary is set as periodic boundary via the “p p p” style. The periodic boundary for the simulation box is set with the input file in line 5.

Line 5, 6:

The *read_data* is the input command that will read in the data file. The data file will consist of the atom coordinates and their symbol so the software recognizes the structure. Also the dimensions lows and highs are in the data file so the simulation box forms accordingly. The masses of the atoms are also described as it is a necessary input to the calculations. A simulation box lows and highs could be described as below:

```
-12.600000 12.600000 xlo xhi
-10.700000 10.700000 ylo yhi
-13.700000 13.700000 zlo zhi
```
Line 7, 8:

These lines are associated with defining the force field that the user may use to calculate the interatomic interactions. This study has suggested the ReaxFF to be used considering the advantages that this force field has to offer.

Line 9:

The fix line performs the charge equilibration (QEq) method. It is typically used in conjunction with the ReaxFF force field model as implemented in the pair_style reax/c command. The QEq method minimizes the electrostatic energy of the system by adjusting the partial charge on individual atoms based on interactions with their neighbors.

Line 10:

thermo command is placed here to print out the retrieved thermodynamic information of the system. This action will output the information for every 500 step that takes place in the rest of the lines, enabling a deeper more detail insight to the material behavior.

Lines 11, 12:

These lines will output the atom coordination’s as a file facilitating the visual modeling of the system.

1 units real
2 atom_style full
3 timestep 0.1
4 boundary p p p
5 read_data ice.data
Input script 1. Part A

4.2 Input script part B: equilibration

Line 13:

The system will iteratively be brought to minimal potential energy by adjusting the atom coordinates read in by line 5 via the minimize command.

Line 14 to 24:

Herein we perform a constant NVE integration to update the velocity and the positions of the atoms where V is the volume and E is the energy, creating a system trajectory consistent with the micro-canonical ensemble. The actions are executed for the specified numbers of runs and will then be stopped using the *unfix* command.

We will perform constant NVE updates of position and velocity for atoms in the group each timestep via the *nve/limit* style for the fix command before the above operation. A limit is imposed on the maximum distance an atom can move in one timestep. This is useful when starting a simulation with a configuration containing highly overlapped atoms. Normally this would generate huge forces which would blow atoms out of the simulation box, causing LAMMPS to stop with
an error. Using this fix can overcome that problem. Forces on atoms must still be computable (which typically means 2 atoms must have a separation distance > 0.0). But large velocities generated by large forces are reset to a value that corresponds to a displacement of length $xmax$ in a single timestep. $xmax$ is specified in distance units set to be Angstroms in this study. The value of $xmax$ should be consistent with the neighbor skin distance and the frequency of neighbor list re-building, so that pairwise interactions are not missed on successive timesteps as atoms move.

Note that if a velocity reset occurs the integrator (see LAMMPS documentation) will not conserve energy. On steps where no velocity resets occur, this integrator is exactly like the fix nve (line 17) command. Since forces are unaltered, pressures computed by thermodynamic output will still be very large for overlapped configurations.

These commands perform time integration on Nose-Hoover style non-Hamiltonian equations of motion which are designed to generate positions and velocities sampled from the canonical (nvt), isothermal-isobaric (npt), and isenthalpic (nph) ensembles. This updates the position and velocity for atoms in the group each timestep.

The thermostatting and barostatting is achieved by adding some dynamic variables which are coupled to the particle velocities (thermostatting) and simulation domain dimensions (barostatting). In addition to basic thermostatting and barostatting, these fixes can also create a chain of thermostats coupled to the particle thermostat, and another chain of thermostats coupled to the barostat variables. The barostat can be coupled to the overall box volume, or to individual dimensions, including the xy, xz and yz tilt dimensions. The external pressure of the barostat can be specified as either a scalar pressure (isobaric ensemble) or as components of a symmetric stress.
tensor (constant stress ensemble). When used correctly, the time-averaged temperature and stress tensor of the particles will match the target values specified by $T_{\text{start}}/T_{\text{stop}}$ and $P_{\text{start}}/P_{\text{stop}}$.

The thermostat parameters for fix styles nvt and npt is specified using the temp keyword. The thermostat is applied to only the translational degrees of freedom for the particles. The translational degrees of freedom can also have a bias velocity removed before thermostating takes place; see the description below. The desired temperature at each timestep is a ramped value during the run from $T_{\text{start}}$ to $T_{\text{stop}}$. The $T_{\text{damp}}$ parameter is specified in time units and determines how rapidly the temperature is relaxed. For example, a value of 10.0 means to relax the temperature in a timespan of (roughly) 10 time units (e.g., $\tau$ or fmsec or psec - see the units command). The atoms in the fix group are the only ones whose velocities and positions are updated by the velocity/position update portion of the integration.

```
13 minimize  0.0  1.0e-8  1000  10000
14 fix en0 all nve/limit 0.01
15 run  10000
16 unfix en0
17 fix en1 all nve
18 run  20000
19 unfix en1
20 fix en2 all nvt temp 50. 50. 100.
21 run  20000
22 unfix en2
23 fix en3 all npt temp 50. 50. 100. iso 0.0 0.0 1000
```
Input script 1. Part B

4.3 Input script part C: computations

Line 25, 26:

The input script contains a computation part that starts on line 25. The first line of this part or the 25th line of the whole input script calculates the potential energy for each atom in the system. If no keywords are specified in the command line, all the potentials, meaning the sum of the pair, bond, angle, dihedral, improper, kspace (long-range) etc., are calculated. The next line is to reduce one or more vector inputs into scalar values, one per listed input. The inputs are per-atom in this case and reduce the pe/atom style in the above line (line 25). The sum mode for this particular command will output the sum of the calculated potentials and place them into the C_pt0 column.

Line 27, 28:

Lines 27 and 28 compute the symmetric per-atom tensor for each atom in a group. Herein, the group is all atoms, the tensor for each atom has 6 components and it is stored as a 6-element vector in the following order: xx, yy, zz, xy, xz, yz. The stress tensor for any given atom is calculated via a certain predefined equation listed in the appendix.

Line 29:

The centro/atom computation calculates the centro-symmetry for each specified atom, which in this case regards all of the atoms. The formula in which this calculated is done is as the following:

\[ N \] is an input parameter, which should be set to correspond to the number of nearest neighbors in the underlying lattice of atoms. If the keyword fcc or bcc is used, \( N \) is set to 12 and 8 respectively.
More generally, $N$ can be set to a positive, even integer. Herein we have set the $N$ to be 4 and in some cases to be 8. Eq. 19:

$$CS = \sum_{i=1}^{N/2} |\vec{R}_i + \vec{R}_{i+N/2}|^2$$

Line 30:

CNA/atom style for compute, will perform the Common Neighbor Analysis pattern for each atom in the group. Here we define the cutoff for the CNA evaluation to be 0.1 which will be an appropriate value for the cutoff to consider the nearest neighbors cutoff in accordance to the presumed Chrystal structure. The formula that LAMMPS uses to calculate the aforementioned evaluation is as following:

Eq. 20:

$$r_{\text{fcc}}^c = \frac{1}{2} \left( \frac{\sqrt{2}}{2} + 1 \right) a \simeq 0.8536 a$$
$$r_{\text{bcc}}^c = \frac{1}{2} (\sqrt{2} + 1) a \simeq 1.207 a$$
$$r_{\text{hcp}}^c = \frac{1}{2} \left( 1 + \sqrt{\frac{4 + 2x^2}{3}} \right) a$$

Line 31, 32, 33:

The dump files, as mentioned in the previous section will output the calculated information and makes it accessible for further analysis.

Line 34:
Use one or more global values as inputs every few timesteps, and average them over longer timescales. The resulting averages can be used by other output commands such as *thermo_style custom*, and can also be written to a file. Note that if no time averaging is done, this command can be used as a convenient way to simply output one or more global values to a file.

Line 35:

Change the volume and/or shape of the simulation box during a dynamics run. Orthogonal simulation boxes have 3 adjustable parameters (x,y,z). Triclinic (non-orthogonal) simulation boxes have 6 adjustable parameters (x,y,z,xy,xz,yz). Any or all of them can be adjusted independently and simultaneously by this command.

For the x, y, z parameters, the associated dimension cannot be shrink-wrapped. For the xy, yz, xz parameters, the associated second dimension cannot be shrink-wrapped. Dimensions not varied by this command can be periodic or non-periodic. Dimensions corresponding to unspecified parameters can also be controlled by a *fix npt* or *fix nph* command.

The size and shape of the simulation box at the beginning of the simulation run were either specified by the *create_box* or *read_data* or *read_restart* command used to setup the simulation initially if it is the first run, or they are the values from the end of the previous run. The *create_box*, *read data*, and *read restart* commands specify whether the simulation box is orthogonal or non-orthogonal (triclinic) and explain the meaning of the xy,xz,yz tilt factors. If *fix deform* changes the xy,xz,yz tilt factors, then the simulation box must be triclinic, even if its initial tilt factors are 0.0.
The desired simulation box size and shape at the end of the run are determined by the parameters of the `fix deform` command. Every Nth timestep during the run, the simulation box is expanded, contracted, or tilted to ramped values between the initial and final values.

```
25 compute pt0 all pe/atom
26 compute pot0 all reduce sum c_pt0
27 compute st0 all stress/atom NULL
29 compute centrosym dio centro/atom 4
30 compute cnatom dio cna/atom 0.1
31 dump xyzdump all xyz 1000 ice_comp.xyz
32 dump_modify xyzdump element H O
33 dump paramdump dio custom 500 parameterpoint.data id c_centrosym c_cnatom
35 fix stretch all deform 10 x scale 1.15 y scale 1.30 z scale 1.15 remap x units box
36 run 60000
```

Input script 1. Part c
5 Finite modeling of materials

In this chapter we discuss the finite modeling of the subject material. As the commands were thoroughly explained in Chapter 3, they are used here in a way that the model acts as if the simulation is acting upon a very small material sample, similar to how it would be tested in the lab, all the interatomic potentials act as if they were a surface molecule. In other words there are significant surface energy effects regarding all of the modules. The surface energy of any crystal
material is different than the bulk energy and it can potentially strengthen or weaken the system (depending on the material). While the periodic model (Chapter 4) represents a bulk, as if the simulation is a huge, continues crystal. Thus. There are no surface effects associated with the simulation. Basically both extremes are studied (a huge periodic system and a small nano system), the actual behavior of the material should be somewhere in between.

In both simulations the specimen boundary conditions are set to be periodic, but it is important to note that the model’s condition regarding the finite or periodic status is defined in how the simulation box is defined. The simulation box regarding the finite model in comparison to the material boundary is set larger so the specimen boundary has a large distance from the simulation box, making it impossible for the interactions like bind order formalisms to exceed to such level of distances, resulting in the model to be confined and the periodic interactions between the boundaries to halt.

5.1 Input script part A: initialization

In this step the units are set, the atom coordinates are inputted to the script and boundary conditions are fixed. The force field will be chosen to act as the ReaxFF and the final (x,y,z) output is defined.

Line 1:

Using the unit command we will set the units as below, using the real style (Input script 1, line 1):

- mass = grams/mole
- distance = Angstroms
- time = femtoseconds
- energy = Kcal/mole
• velocity = Angstroms/femtosecond
• force = Kcal/mole-Angstrom
• torque = Kcal/mole
• temperature = Kelvin
• pressure = atmospheres
• dynamic viscosity = Poise
• charge = multiple of electron charge (1.0 is a proton)
• dipole = charge*Angstroms
• electric field = volts/Angstrom
• density = gram/cm^dim

Resulting in the inputs and the outputs to the software to have known units most importantly the forces, energies, time and distance.

Line 2:

Before reading in the atom coordinates the atoms must be defined. In this study the atom style is set to full (see input script 1, line 2). The attributes for the full style are molecular systems consisting of charges which will be well suited for molecular analysis of material’s mechanical behavior, although according to the documentation, this style is usually used for bio-molecules.

Line 3:

The time-step of the simulation is set via the timestep command, herein the time-steps size will better exhibit the simulation variables and changes in partials of femtoseconds. The unit command has set the simulation time unit to femtoseconds in order for the reactions to manifest a smooth
transition in the creation and the breaking of bonds, complying with the same reason, the time size is set to 0.1 of a femtosecond. The swiftest chemical reaction is 15 femtoseconds, the duration of a vibration of the atoms in an iodine molecule is 300 femtoseconds which shows that the decision of the time size is in reasonable accordance to the mentioned time periods.

Line 4:

The global simulation boundary is set with the boundary command. Here the boundary is set as periodic boundary via the “p p p” style. The finite boundary for the simulation box is set with the input file in line 5. For the model to simulate finite (and not periodic) the atom coordinates have been defined in a way that the distance between the simulation box and the boundary condition is higher than the interaction cutoff (set by the neighbor command) resulting in the model to be considered as finite, halting the interactions of the atoms from opposite sides.

Line 5:

The read_data is the input command that will read in the data file. The data file will consist of the atom coordinates and their symbol so the software recognizes the structure. Also the dimensions lows and highs are in the data file so the simulation box forms accordingly. The masses of the atoms are also described as it is a necessary input to the calculations. A simulation box lows and highs could be described as below:

\[-40.000000 40.000000 \text{xlo xhi}\]

\[-40.000000 40.000000 \text{ylo yhi}\]

\[-40.000000 70.000000 \text{zlo zhi} \]

Line 6:
The *neighbor* command sets the cutoff in which the pairwise interactions would be defined. As mentioned in the previous line, the cutoff is set lower than the distance between the global boundary conditions and the simulation box to permit the finite modeling of the simulation. The bin style will set the receptacle distance of the reactions to 0.5 which will consider essential reactions and also will not be computationally expensive for the simulation to run.

Line 7, 8:

These lines are associated with defining the force field that the user may use to calculation the interatomic interactions. This study has suggested the ReaxFF to be used considering the advantages that this force field has to offer.

Line 9:

This line performs the charge equilibration (QEq) method. It is typically used in conjunction with the ReaxFF force field model as implemented in the *pair_style reax/c* command. The QEq method minimizes the electrostatic energy of the system by adjusting the partial charge on individual atoms based on interactions with their neighbors.

Line 10:

This command is placed here to print out the retrieved thermodynamic information of the system. This action will output the information for every 500 step that takes place in the rest of the lines, enabling a deeper more detail insight to the material behavior.

Lines 11, 12:

These lines will output the atom coordination’s as a file facilitating the visual modeling of the system.
Herein, the boundaries are described, and enabling the periodic simulation of the system in contradiction to the previous simulation (Chapter 4).

The region command defines a geometric region or in other word to define a box boundary for the system. Values for lo/hi block style and other styles can specify the extent of the boundary. For example if the style is set as block and the args are to be numbers, the boundary box will be determined according to the numbers entered. If the args are set as EDGE or INF it means they extend all the way to the global simulation box boundary.

block args = xlo xhi yli yhi zlo zhi

xlo, xhi, yli, yhi, zlo, zhi = bounds of block in all dimensions (distance units)

For style block, the lattice spacing in dimension x is applied to xlo and xhi, similarly the spacings in dimensions y,z are applied to ylo/yhi and zlo/zhi.

Identify a collection of atoms as belonging to a group. The group command can be used with region as its style, meaning the region style puts all atoms in the region volume into the group. If the region is used as a style of the group, the groups are static one-time assignment. The atoms remain the same, (assigned or not assigned) to the group even if they leave their region volume.

In these lines we will group the regions together in order to fix them with proper commands in the next section. Block 1 and block 2 are grouped as top and bot (bottom and top regions) and further grouped to the fix ID. Region 3 will be grouped to act separately and named mid in the syntax.
units real
atom_style full
timestep 0.1
boundary p p p
read_data ice.data
neighbor 0.5 bin
pair_style reax/c NULL
pair_coeff * * ffield.reax.Fe_O_C_H H O
fix charger all qeq/reax 1 0.0 10.0 1.0e-6 reax/c
thermo 1000
dump xyzDump all xyz 500 ice_crystal_test.xyz
dump_modify xyzDump element H O
region 1 block INF INF INF INF 38. INF units box
region 2 block INF INF INF INF -12. units box
region 3 block INF INF INF -5. 35. units box
group top region 1
group bot region 2
group fixx union top bot
group mid region 3

Input script 2. Part A
5.2 Input script part B: equilibration

Like the previous periodic model simulation, this simulation also requires equilibration of the system. Equilibrium is done to ensure the stability of the system and avoid further error and preventing the system to blow up.

Line 20:

Herein we perform a constant NVE integration to update the velocity and the positions of the atoms where V is the volume and E is the energy, creating a system trajectory consistent with the microcanonical ensemble. The actions are executed for the specified numbers of runs and will then be stopped using the `unfix` command.

We will perform constant NVE updates of position and velocity for atoms in the group each timestep via the `nve/limit` style for the `fix` command before the above operation. A limit is imposed on the maximum distance an atom can move in one timestep. This is useful when starting a simulation with a configuration containing highly overlapped atoms. Normally this would generate huge forces which would blow atoms out of the simulation box, causing LAMMPS to stop with an error. Using this `fix` can overcome that problem. Forces on atoms must still be computable (which typically means 2 atoms must have a separation distance > 0.0). But large velocities generated by large forces are reset to a value that corresponds to a displacement of length $x_{max}$ in a single timestep. $Xmax$ is specified in distance units set to be Angstroms in this study. The value of $x_{max}$ should be consistent with the neighbor skin distance and the frequency of neighbor list rebuilding, so that pairwise interactions are not missed on successive timesteps as atoms move.

Line 21 to 28:
Set each component of force on each atom in the group to the specified values \( fx, fy, fz \). This erases all previously computed forces on the atom, though additional fixes could add new forces. This command can be used to freeze certain atoms in the simulation by zeroing their force, either for running dynamics or performing an energy minimization. For dynamics, this assumes their initial velocity is also zero.

Also the \textit{velocity} commands is set as all 0 components similar to the \textit{setforce} style. This is done to set all the forces and the velocities to 0 in order to \textit{fix} the system boundary at top and bottom (refer to group command in chapter 5, part A)


document code:

```
20 fix          en0 all nve/limit 0.01
21 fix          2 fixx setforce 0.00 0.00 0.00
22 velocity     fixx set 0.0 0.0 0.0 units box
23 run          20000
24 unfix        en0
25 fix          en1 all nve
26 fix          2 fixx setforce 0.00 0.00 0.00
27 velocity     fixx set 0.0000 0.0 0.0 units box
28 run          20000
```

Input script 2. Part B

5.3 Input script part C: computations

Line 29 to 34:

All the potentials, meaning the sum of the pair, bond, angle, dihedral, improper, kspace (long-range) etc., are calculated with the \textit{pe/atom} style. The next line is to reduce one or more vector
inputs into scalar values, one per listed input. The inputs are per-atom in this case and reduce the pe/atom style. The sum mode for this particular command will output the sum of the calculated potentials and place them into the C_pt0 column.

This line will compute the symmetric per-atom tensor for each atom in a group. Herein, the group is all atoms, the tensor for each atom has 6 components and it is stored as a 6-element vector in the following order: xx, yy, zz, xy, xz, yz. Also the next line would reduce the vectors to scalar values and enables the output procedure.

Line 35 to 37:

With the variable command, it is possible to assign string(s) to a variable name for evaluation later in the input script or during a simulation, enabling the output procedure for the computed information.

Line 38, 39:

The fix print command will print the text string every N (500 in this script) steps during the simulation run.

The deform fix will also apply change of volume and shape to the simulation box. For the scale style that is used in the script, for all dimensions, if the scale number is below 1, the simulation box will tighten which means there would be compression applied to the specimen. If the scale number is more than one the simulation box will expand, leaving the specimen to experience tension. Here’s how: the initial box length is 10 and the factor is 1.1, then the final box length will be 11. A factor less than 1 means compression.
Line 40:

Line number 40 of the command list will shorten the distance between the top boundary to keep the appropriate distance to the specimen defined as the mid.

29 compute pt0 mid pe/atom
30 compute pot0 mid reduce sum c_pt0
31 compute st0 mid stress/atom NULL
32 compute sz mid reduce sum c_st0[3]
33 compute zmax mid reduce max z
34 compute zmin mid reduce min z
35 variable total_pe equal c_pot0
36 variable total_stress equal c_sz
37 variable total_length equal c_zmax-c_zmin
38 fix print_data all print 500 "${total_length} ${total_stress} ${total_pe}" append stress_nve_dyn1p20.data
39 fix shift top move linear 0.0 0.0 0.002 units box
40 run 200000

Input script 2. Part C
6 Molecular dynamics case study: mechanical characterization of crystalized ice crystallized

This study has offered a methodology in which mechanical characterization via full atomistic simulation of materials are made possible while employing tools such as the ReaxFF force field. Although MD simulations are not novel approaches neither the ReaxFF, and have been previously researched on and more developed since mid-50s and 2001 respectively, but combining these techniques and suggesting non-case related mechanical characterization for non-biological materials has been the primary goal of this study up until this chapter. Granting all this, this chapter utilizes the aforementioned techniques such as RMD (reactive molecular dynamics), ReaxFF and LAMMPS to characterize the mechanical behavior of ice via a full atomistic simulation. Ice models are executed both as a finite model and a periodic boundary condition model and compared
by their mechanical behavior results since the surface energy in these models will change, and the mechanical behavior may vary. Alongside these examinations, distinction between experimentally determined mechanical properties of ice attained from credible past research and the procured material from the molecular dynamics simulation of ice are analyzed in this chapter.

The inputs to this study have been the LAMMPS input script and ice atom coordinates. The LAMMPS input has been thoroughly explained in Chapter 3, 4 and 5. The coordinates for 1296 atoms, meaning 432 molecules of water have been read-in to the analysis for the periodic model. A number of 2592 atoms equal to 864 molecules have been inputted regarding the finite model. All of which containing atoms of oxygen and hydrogen with the masses of 15.99 and 1.01 mols respectively. For simulation purposes, such as more consistency with the reality and avoiding simulation over-estimations samples were subjected to a flaw, meaning one molecule has been removed from the most center of the simulation specimen.

In order to properly simulate the models, both the periodic and the finite model for ice have been brought to their minimum potential energy. This procedure alongside nve and npt procedures will avoid excess unrealistic velocities for the atoms and stabilizes the simulation.

Anticipation to the mechanical behavior of ice is that the laboratory experimented mechanical properties would fall in between the finite and the periodic model due to the surface energy difference between the bonds. In finite modeling of ice, the surface energy is allocated to all the atoms and the model acts as if it is a very small experimental specimen that contains surface energy throughout the structure, whereas in the periodic model the specimen is without any surface energy acting as a bulk of experimentally evaluate ice in an ideal condition without flaws and discontinuities.
6.1 Tensile and compressive strength of ice

Typically, mechanical properties calculated \textit{via} simulations tend to be higher from the experimentally determined mechanical behaviors. The reason for the mentioned behavior is that in experimental specimens there many defects within the structure of the material, from other particles inside the crystal structure to cracks and irregularities within the structure itself, whereas in simulation, the crystal is perfect, the structure is uniform and there are no adulterations caused by other elements, in result, the mechanical strength both in compressive and tension stress are higher in simulation tests.

\textbf{Figure 8} exhibits a stress strain curve of a flawless material subject to triaxial compressive stress, the curve shows a vibrating change which is typical in simulations. It almost starts from the origin of the plot, increases steadily to a point in which the increase will gradually have lower stress rate increase, this point is the first presumed failure of the specimen, also the differential of the curve almost hits zero on the vertical axis indicating that the slope of the stress-strain curve has become constant (\textbf{Figure 9}) enabling us to state that the failure point might have reached if the simulation box would have vacancies for particle migration. After the presumed failure at point A, the graph starts to increase somewhat linearly until point B, until another linear increase with a higher slope relative to the previous line emerges, starting at around point \((0.7, 3 \times 10^3)\) or point B up to point C where the simulation is stopped. The compressive stress simulation runs are done by performing a triaxial loading upon the system, the simulation box is confined in a way that there are no vacant space for particle migration, causing the stress-strain graph to display the mentioned linear increases after the presumed failure point at A.
Figure 8. Flawless system Stress-Strain curve subject to compression with 0.50 compression rate along the Y axis

Figure 9. Flawless system stress rate-strain curve subject to compression with 0.5 compression rate along the Y axis
In simulation the structure is perfect, causing the specimen to fail in high numbers of stress. In order to partially accommodate the simulation towards a more reasonable outcome, this study has created a fault within the structure, removing one ice molecule from the input coordinates. The fault is enough to initiate the fracture propagation within the structure the curve will show more difference while subjected to higher strains.

**Figure 10** shows the stress strain curve of the faulted system reaching the constant stress rate at point A while subjected to triaxial compression with a total applied strain of 0.50. The strain rate of 0.50 means that the specimen is subject to compression with a strain rate of 0.50 each run, that is subtracting 50% of the model’s original size each run (explained in Chapter 5.1.3). The simulation strains are very high in comparison to experimental tests resulting in a shock to the system. **Figure 11** shows a system subjected to tension with an applied strain of 1.30 pulling alongside the Y axis of the model. Unlike the compression curves, the failure point of the model is apparent at point A. With tension, the model is being pulled and allowing space for particle migration. As a result, the tensile tests are more credible in the study and allow determination of tension parameters such as elastic modulus and tensile strength of simulated ice. Herein, the tensile strength of the models are from 800 to 900 MPa at a strains spanning from 0.22 to 0.35 while the temperature is set at -223.25°C, whereas over the temperature range of -10° to -20° C the tensile strength varies from 0.7 to 3.1 MPa [7] showing a magnitude of 100s to a thousand higher in simulations in comparison to actual experimental ice tensile strengths.
Figure 10. Stress-strain curve of a system subject to compression (with flaw in the system) with 0.50 compression rate along the Y axis.
Figure 11. Stress-strain curve of a model subject to tension (with flaw) with pulling rate of 1.30 along Y axis and 1.15 along X and Z axis

6.2 Ice elastic modulus

For the calculation of the Young modulus on the tensile curves obtained by the simulation runs spanning from strain rates of 1.15 to 2.75 in the Y axis of the model, while equally 1.15 in the X and Z axis, we have considered the portion of the graph that exhibits a linear increase starting at the origin and before the plastic behavior kicks in. Figure 12 shows the stress-strain curve of a model subjected to 1.15 pulling rate along all three axis that shows the linear increase of the stress within the first 0.05 strain of the horizontal axis. Simulations indicate a reasonable range from 300 to 400 MPa for the elastic modulus, the more the strain rate the more the young modulus is becoming. While the experimental young modulus determination suggests from 970 to 1120 Mpa [7] for Polycrystalline ice, subjecting 0.5 m diameter ice plates to biaxial bending.
Figure 12. Stress-strain curve of a model subject to tension (with flaw), pulling rate of 1.15 along Y,X,Z axis. Perfectly showing the linear stress increase below 0.05 strain and 300 Mpa
6.3 Strain rate effects on ice mechanical behavior

In this study, strain rate depends on two parameters, the first parameter being the compression or the expansion rate or the deformation rate of the simulation box which is defined by the deform command in LAMMPS. The second parameter is the timestep defined for running the deform command. While keeping the timestep of the tensile tests constant, we will deploy different deformation rates affecting the model. Figure 13 exhibits the stress-strain curve of two different deformation rates for a model subjected to expansion, curve (A) relating to 1.15 deformation rate and (B) for 1.30, and keeping the timesteps of both the simulations to a constant of 60 timesteps. As mentioned in the previous section, the elastic modulus remain the same. At strain of 0.1 the curves start to behave different, with the 1.30 expansion rate curve exceeding the 1.15 expansion rate curve in terms of stress magnitudes, while the failure point of the higher deformation rate curve (B) emerging later in terms of strain. This effect is due to a shock to the system whereas experimental tests showing sooner failure of the material.

Previously we set the timesteps in which the stress-strain curve occurs in as constant, showing two types of deformation or expansion rates and their specifications. Herein, the expansion rates are kept constant to 1.30, while changing the timesteps in which the expansion acts in, or in other words, the horizontal axis represents the time versus the stress magnitude in the vertical axis. Figure 14 clearly shows sooner rupture in terms of time in curve (A) indicating that applying faster expansion will indeed result in sooner failure in contradiction to experimental tests that fail to exhibit failure difference when strain rate is subject to change [7].
Figure 13. Strain rate effects on stress-strain curves due to change in expansion rate.

Figure 14. Strain rate effects on stress-strain curves due to change in the simulation timestep.
7 Conclusion

This study has successfully suggested a novel methodology to simulate material in order to characterize their mechanical behavior, although simulation methods tend to overestimate mechanical parameters, but it is a useful tool to better understand a structure independent of its real life contaminations. For example, this study clearly shows the strain rate effects in ice specimens that are not possible with experimental tests on ice. The use of MD and the ReaxFF force field are thoroughly explained. This enables readers to perform further simulations using the information in this study and apply them to many other materials.

The compressive stress simulation runs are done by performing a triaxial deformation. The simulation box is confined in a way that there are no vacant space for particle migration, resulting in difficulties to better study the compression of the system. On the other hand, comprehensive studies are performed on the tensile experiments, offering useful mechanical properties that can in future assist researchers to better understand the puzzling attributes of ice.
Appendix

LAMMPS documentation for the real style for the unit command:

- mass = grams/mole
- distance = Angstroms
- time = femtoseconds
- energy = Kcal/mole
- velocity = Angstroms/femtosecond
- force = Kcal/mole-Angstrom
- torque = Kcal/mole
- temperature = Kelvin
- pressure = atmospheres
- dynamic viscosity = Poise
- charge = multiple of electron charge (1.0 is a proton)
- dipole = charge*Angstroms
- electric field = volts/Angstrom
- density = gram/cm^dim
Input script for periodic tension models:

```plaintext
#-----------------------------------------------
#                        
units real

atom_style full

timestep 0.1

boundary p p p

read_data ice.data

neighbor 0.5 bin

pair_style reax/c NULL safezone 2 mincap 80

pair_coeff * * fField.reax.Fe_O_C_H H O

group dio type 2

fix charger all qeq/reax 1 0.0 10.0 1.0e-6 reax/c

thermo 500

#-----------------------------------------------
#minimize 0.0 1.0e-8 1000 10000

#fix en0 all nve/limit 0.01

#run 10000
```
#unfix en0
#fix en1 all nve
#run 20000
#unfix en1
#fix en2 all nvt temp 50. 50. 100.
#run 20000
#unfix en2
#fix en3 all npt temp 50. 50. 100. iso 0.0 0.0 1000.
#run 100000
#-----------------------------------------------

#unfix en3
fix en4 all nvt temp 50. 50. 100.
compute pt0 all pe/atom
compute pot0 all reduce sum c_pt0
compute st0 all stress/atom NULL
#compute centrosym dio centro/atom 4
#compute cnatom dio cna/atom 0.1

dump xyzdump all xyz 1000 ice_comp.xyz

dump_modify xyzdump element H O

#dump paramdump dio custom 500 parameterpoint.data id
c_centrosym c_cnatom


fix stretch all deform 10 x scale 1.15 y scale 1.30 z scale 1.15 remap x units box

run 200000

#-------------------------------------------------------------
----------------------------
Input script for periodic compression models:

```plaintext
#-------------------------------------------------------------
#-------------------------------------------------------------
units real
atom_style full
timestep 0.1
boundary p p p
read_data ice.data
neighbor 0.5 bin
pair_style reax/c NULL safezone 2 mincap 80
pair_coeff * * ffield.reax.Fe_O_C_H H O
group dio type 2
fix charger all qeq/reax 1 0.0 10.0 1.0e-6 reax/c
thermo 500
fix en4 all nvt temp 50. 50. 100.
compute pt0 all pe/atom
compute pot0 all reduce sum c_pt0
compute st0 all stress/atom NULL
```
compute centrosym dio centro/atom 4
compute cнатom dio cna/atom 0.1
dump xyzdump all xyz 1000 ice_comp.xyz
dump_modify xyzdump element H O
dump paramdump dio custom 500 parameterpoint.data id c_centrosym c_cнатom
fix stretch all deform 10 x scale 0.5 y scale 0.5 z scale 0.5 remap x units box
run 500000
Input script for finite tension models:

```plaintext
units real
atom_style full
timestep 0.1
boundary p p p
read_data ice_crystal.dat
neighbor 0.5 bin
pair_style reax/c NULL
pair_coeff * * ffield.reax.Fe_O_C_H O H
fix charger all qeq/reax 1 0.0 10.0 1.0e-6 reax/c
thermo 1000
dump xyzDump all xyz 2000 crystal_nve_dyn0.xyz
dump_modify xyzDump element O H
region 1 block INF INF INF INF 38. INF units box
region 2 block INF INF INF INF -12. units box
region 3 block INF INF INF INF -5. 35. units box
group top region 1
```
group             bot region 2

group             fixx union top bot

group             mid region 3

#---------------------------------------------------------------
#---------------------------------------------------------------

fix             en0 all nve/limit 0.01

fix             2 fixx setforce 0.00 0.00 0.00

velocity        fixx set 0.0 0.0 0.0 units box

run             20000

unfix             en0

fix             en1 all nve

fix             2 fixx setforce 0.00 0.00 0.00

velocity        fixx set 0.0000 0.0 0.0 units box

run             20000

compute         pt0 mid pe/atom

compute         pot0 mid reduce sum c_pt0

compute         st0 mid stress/atom

compute         sz mid reduce sum c_st0[3]

compute         zmax mid reduce max z
compute zmin mid reduce min z

variable total_pe equal c_pot0

variable total_stress equal c_sz

variable total_length equal c_zmax-c_zmin

fix print_data all print 500 "${total_length} ${total_stress} ${total_pe}" append stress_nve_dyn0.data

fix shift top move linear 0.0 0.0 0.002 units box

run 200000

Input script for finite compression models:

units real

atom_style full

timestep 0.1

boundary p p p

read_data ffile_crystal.dat

neighbor 0.5 bin

pair_style reax/c NULL

pair_coeff* * ffield.reax.Fe_O_C_H O H

fix charger all qeq/reax 1 0.0 10.0 1.0e-6 reax/c
thermo 1000
dump xyzDump all xyz 2000 crystal_nve_dyn0.xyz
dump_modify xyzDump element O H
region 1 block INF INF INF INF 38. INF units box
region 2 block INF INF INF INF -12. units box
region 3 block INF INF INF INF -5. 35. units box
group top region 1
group bot region 2
group fixx union top bot
group mid region 3
#
-----------------------------------------------------------------------------------
----------------------------------------
fix  en0 all nve/limit 0.01
fix  2 fixx setforce 0.00 0.00 0.00
velocity fixx set 0.0 0.0 0.0 units box
run  20000
unfix en0
fix  en1 all nve
fix  2 fixx setforce 0.00 0.00 0.00
velocity       fixx set  0.0000  0.0 0.0  units box

run             20000

compute         pt0 mid pe/atom

compute         pot0 mid reduce sum c_pt0

compute         st0 mid stress/atom NULL

compute         sz mid reduce sum c_st0[3]

compute         zmax mid reduce max z

compute         zmin mid reduce min z

variable         total_pe equal c_pot0

variable         total_stress equal c_sz

variable         total_length equal c_zmax-c_zmin

fix print_data all print 500 "${total_length} ${total_stress} ${total_pe}" append periodicpressurep50.data

#fix               stretch all deform 1 x scale 1.0 y scale 1.0 z
scale 0.50 remap x units box

fix               shift top move linear 0.0 0.0 0.002 units box

run             100000

#-----------------------------------------------
The tensor for each atom has 6 components and it is stored as a 6-element vector in the following order: xx, yy, zz, xy, xz, yz. The stress tensor for any given atom is calculated via the following equation:

\[
S_{ab} = - \left[ m v_a v_b + \frac{1}{2} \sum_{n=1}^{N_p} (r_{1a} F_{1b} + r_{2a} F_{2b}) + \frac{1}{2} \sum_{n=1}^{N_b} (r_{1a} F_{1b} + r_{2a} F_{2b}) + \frac{1}{3} \sum_{n=1}^{N_a} (r_{1a} F_{1b} + r_{2a} F_{2b} + r_{3a} F_{3b}) + \frac{1}{4} \sum_{n=1}^{N_d} (r_{1a} F_{1b} + r_{2a} F_{2b} + r_{3a} F_{3b} + r_{4a} F_{4b}) + \frac{1}{4} \sum_{n=1}^{N_i} (r_{1a} F_{1b} + r_{2a} F_{2b} + r_{3a} F_{3b} + r_{4a} F_{4b}) + K_{\text{space}}(r_{1a}, F_{1b}) + \sum_{n=1}^{N_i} r_{ia} F_{ib} \right]
\]

A comprehensive potential energy calculation method in ReaxFF:

\[
E_{\text{potential}} = (E_{\text{bond}} + E_{\text{over}} + E_{\text{under}}) + (E_{\text{valency angle}} + E_{\text{penalty}}) + E_{\text{torsion}}
\]

\[
+ E_{\text{lone pair}} + E_{\text{conjugation}} + E_{\text{H-bond}} + E_{\text{vdWaals}} + E_{\text{Coulomb}}
\]
**Table of figures**

Figure 1. Actual molecular studies performed in this study by VMD ..........................10

Figure 2. Experimental determination of material’s mechanical properties .....................10

Figure 3. Atom formation of hexagonal ice crystal [7]..................................................14

Figure 4. Electron repulsion approximations. Time efficiency versus number of atoms that can be computed ..................................................................................................................22

Figure 5. Interatomic distance dependency of C-C bond order. By Adri C.T van Duin [16] ......29

Figure 6. (a) Effect of the bond order correction on C-C and C-H bond orders. (b) Effects of shortening of the C-C bond to 1.0 Å on the relaxed C-H bond as calculated DFT on the left and ReaxFF on the right [16]. ...........................................................................................................30

Figure 7. Shielding included for both Coulomb and van der Waals in order to avoid excessive interaction between atoms sharing bond and/or bond angle [20] .........................................................32

Figure 8. Flawless system Stress-Strain curve subject to compression with 0.50 compression rate along the Y axis .................................................................................................................................80

Figure 9. Flawless system stress rate-strain curve subject to compression with 0.5 compression rate along the Y axis .................................................................................................................................80

Figure 10. Stress-strain curve of a system subject to compression (with flaw in the system) with 0.50 compression rate along the Y axis .................................................................................................................................82

Figure 11. Stress-strain curve of a model subject to tension (with flaw) with pulling rate of 1.30 along Y axis and 1.15 along X and Z axis .................................................................................................................................83

Figure 12. Stress-strain curve of a model subject to tension (with flaw), pulling rate of 1.15 along Y,X,Z axis. Perfectly showing the linear stress increase below 0.05 strain and 300 Mpa ..............84

Figure 13. Strain rate effects on stress-strain curves due to change in expansion rate ..........86
Figure 14. Strain rate effects on stress-strain curves due to change in the simulation timestep ...

References


