ATOMISTIC MODELING OF ELASTIC AND TRANSPORT PROPERTIES OF CARBON NANOTUBES

A THESIS SUBMITTED TO THE GRADUATE SCHOOL IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

MASTER OF SCIENCE

BY

FERAS G. ALZUBI

ADVISOR

DR. RONALD M. COSBY

DEPARTMENT OF PHYSICS AND ASTRONOMY
BALL STATE UNIVERSITY
MUNCIE, INDIANA
JULY 2008
ABSTRACT

ATOMISTIC MODELING OF ELASTIC AND TRANSPORT PROPERTIES OF CARBON NANOTUBES

FERAS G. ALZUBI
Center for Computational Nanoscience (CCN),
Department of Physics and Astronomy,
Ball State University.

A first principles atomistic calculation and analysis is used to conduct studies on the mechanical and electron transport properties of selected stretched single-wall carbon nanotube segments. The atomic forces, electron densities, current, voltage and total energies are calculated for these carbon nanotube segments using Atomistix’s Virtual NanoLab (VNL) and ToolKit (ATK), a software package for electronic structure calculations and molecular dynamics simulations of different molecular systems. Plots of electronic energy spectra, densities of states, force versus length, and current-voltage data, are presented as output results. The mechanical properties of these carbon nanotube segments under a maximum strain of 1% are studied.

A speculative atomistic-level stress-strain approach is tried for calculating Young’s modulus for a single-wall carbon nanotube segment. The computed total energies are also used to extract the Young’s modulus value. Based on the results, the approach is found to work and we were able to calculate the mechanical parameters for single-wall carbon nanotube segments. The electrical conductance is obtained from the current-voltage curves for strained single-wall metallic carbon nanotube segments placed between copper contacts.
ACKNOWLEDGEMENTS

Many great thanks and praise are due to my parents Ghazi M. Alzubi and Ameera S. Ramadan who are always the main source of my success in my life and who are the only people that I will never be able to repay for their endless patience, encouragement, care and love. I am also in debt to my brothers from the Islamic Center of Muncie who were my family here in America during my studying and living and who, because of them, I have never felt homesick. They are all worthy of praise.

I would like to gratefully acknowledge my advisor and mentor, Dr. Ronald M. Cosby for his enthusiastic supervision and for the knowledge that I have gained from him over the past year. With his great character, he was a good example for me in regards to my future scientific life. With his enthusiasm, inspiration and great efforts in explaining concepts clearly and simply, he helped make physics more fun for me. Throughout my personal life here in the USA, he provided patience, encouragement, sound advice, good teaching and never-ending support. I would have been lost without him.

I would like to thank Dr. David Ober for his kindness and willingness to help, especially in the first days of my study at Ball State University. Also I would like to thank Dr. Thomas Robertson and all the professors in the department, especially under whom I have taken classes, for sharing their knowledge with me. Finally, my thanks go to the all students in the department who were supportive and helpful friends.
CONTENTS

List of Figures .......................................................................................................................... vi

List of Tables ............................................................................................................................ ix

1 Introduction ............................................................................................................................ 1

   1.1 Carbon Nanotubes (CNTs) .......................................................................................... 3

   1.2 Carbon Nanotube Properties ....................................................................................... 5

   1.3 Carbon Nanotube Applications .................................................................................... 6

       1.3.1 Energy Storage ...................................................................................................... 7

       1.3.2 Carbon Nanotubes-based Electronics ................................................................. 8

   1.4 Project Goals and Thesis Overview ............................................................................ 10

2 Literature Review of Carbon Nanotube Properties ....................................................... 12

   2.1 Mechanical Parameters .............................................................................................. 12

       2.1.1 Definitions of the Mechanical Parameters ......................................................... 12

       2.1.2 Literature Review of the CNT Mechanical Parameters ....................................... 14

   2.2 Electron Transport in Carbon Nanotube ..................................................................... 21

       2.2.1 Electron Transport for Strained CNTs ................................................................. 21
3 Theoretical Foundations and Software

3.1 Density Functional Theory (DFT)

3.2 Non-Equilibrium Green’s Formalism (NEGF)

3.3 Atomistix Software

3.3.1 Virtual NanoLab (VNL)

3.3.2 Atomistix ToolKit (ATK)

4 Methods and Procedures

4.1 NanoSystems Adressed in the Project

4.1.1 Carbon Chain

4.1.2 Isolated CNT Segments

4.1.3 CNT In Between Two Copper Electrodes

4.1.4 Calculation Parameters

4.2 Approaches and Calculations

4.2.1 Mechanical Properties

4.2.1.1 The Approach of Stretching the CNT and Defining Poisson’s Ratio

4.2.1.2 The Script of the Stretching Process

4.2.1.3 Calculation Parameters Convergence Test

4.2.1.4 The Methods of Calculating Young’s Modulus

4.2.2 Two-Probe Method for Calculating the Conductance
5 Results and Analyses................................................................. 106

5.1 Mechanical Properties Results............................................. 107
  5.1.1 Carbon Chain................................................................. 107
  5.1.2 Carbon Nanotube Segments........................................... 111
    5.1.2.1 The Results with Poisson’s Effect.............................. 112
    5.1.2.2 The Results without Poisson’s Effect......................... 129
    5.1.2.3 The Electron Densities............................................ 138
    5.1.2.4 CNT (5, 5) and CNT (6, 6) Analysis........................... 142

5.2 Transport Properties for Strained CNT (4, 4)......................... 145

6 Summary and Conclusions.................................................... 153

Appendix A Python Script for the C-Chain................................... 157

Appendix B Python Script for the CNT (4, 4)................................ 161

Appendix C Python Script for the Two Probe System ...................... 165

Appendix D Python Script for the Convergence Test....................... 170

List of References......................................................................... 175
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Honey-comb lattice and the chiral vector in terms of the basis vectors.</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>CNT types based on rolling-up graphite sheet.</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>$sp^2$ orbital formation [11].</td>
<td>5</td>
</tr>
<tr>
<td>1.4</td>
<td>Set-up CNTFET [8].</td>
<td>8</td>
</tr>
<tr>
<td>1.5</td>
<td>p-n-p carbon nanotube device [8].</td>
<td>9</td>
</tr>
<tr>
<td>1.6</td>
<td>MWCNT as a tip [8].</td>
<td>9</td>
</tr>
<tr>
<td>2.1</td>
<td>$Y_e$ as a function of the diameter [17].</td>
<td>18</td>
</tr>
<tr>
<td>2.2</td>
<td>Poisson’s ratio dependence on the nanotube diameter [26].</td>
<td>20</td>
</tr>
<tr>
<td>2.3</td>
<td>Conductance as a function of strain for MWCNT [38].</td>
<td>22</td>
</tr>
<tr>
<td>3.1</td>
<td>VNL: Work-Flow [69].</td>
<td>34</td>
</tr>
<tr>
<td>3.2</td>
<td>Lab Floor in the Virtual Nanolab.</td>
<td>35</td>
</tr>
<tr>
<td>3.3</td>
<td>A water molecule visualized in nanoscope.</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>The context menu can be opened by right-clicking anywhere in the nanoscope window.</td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>A molecule constructed by Molecular Builder.</td>
<td>37</td>
</tr>
<tr>
<td>3.5</td>
<td>NanoLanguage Scripter screen and tabs.</td>
<td>39</td>
</tr>
<tr>
<td>3.6</td>
<td>A molecule Method Editor.</td>
<td>40</td>
</tr>
<tr>
<td>3.7</td>
<td>Atomic Manipulator.</td>
<td>42</td>
</tr>
<tr>
<td>3.8</td>
<td>Crystal Cupboard.</td>
<td>43</td>
</tr>
<tr>
<td>3.9</td>
<td>Nanotube Grower-window.</td>
<td>45</td>
</tr>
<tr>
<td>3.10</td>
<td>Tight-binding structure for a carbon nanotube constructed in the Nanotube Grower.</td>
<td>45</td>
</tr>
<tr>
<td>3.11</td>
<td>The Result Browser window.</td>
<td>47</td>
</tr>
<tr>
<td>3.12</td>
<td>Final log window of a script run through Job Manager.</td>
<td>48</td>
</tr>
<tr>
<td>3.13</td>
<td>The electrodes of the two-probe system in the Atomic Manipulator.</td>
<td>49</td>
</tr>
<tr>
<td>3.14</td>
<td>The finished configuration of two-probe system.</td>
<td>50</td>
</tr>
<tr>
<td>4.1</td>
<td>Three neighboring (111) layers in fcc crystal [75].</td>
<td>57</td>
</tr>
<tr>
<td>4.2</td>
<td>A 3x3 (111) layer [75].</td>
<td>58</td>
</tr>
<tr>
<td>4.3</td>
<td>The location of the B and C layers relative to the A layer.</td>
<td>59</td>
</tr>
</tbody>
</table>
4.4 The final layout of the constructed two-probe system: a carbon chain positioned in-between the (111) surfaces of two Cu crystals.  
4.5 The unit cell of CNT (4, 4) created by VNL.  
4.6 The final structure of the metallic single-wall CNT (4, 4).  
4.7 The final structures of the metallic single-wall CNT (4, 4) segments. 
The difference between each two horizontally adjacent segments is one unit cell.  
4.8 The final structure of the metallic single-wall CNT (4, 4) in between two copper electrodes.  
4.9 Poisson’s ratio as a function of the strain for a carbon nanotube.  
4.10 Top view on the CNT before and after the stretching. The radius gets smaller after the axial stretching according to Poisson’s effect.  
4.11 Computed total energy for a 48-atom CNT segment plotted for a sequence of mesh cut-off values.  
4.12 Computed total energies for 96-, 144-, and 192-atom CNT segments plotted for a sequence of mesh cut-off values.  
4.13 Computed total energy for the CNT between the copper surface layers plotted for a sequence of mesh cut-off values.  
4.14 The view of the carbon nanotube end.  
5.1 Change in force on the right end atom versus the change in the length of the carbon chain.  
5.2 The forces versus the strain of the left end of the CNT.  
5.3 The forces versus the strain of the right end of the CNT.  
5.4 Total energy versus strain for the CNT with 48-atom.  
5.5 The first derivative of the total energy with respect to the strain for the CNT with 48-atom.  
5.6 The force-strain plots of all the CNT segments when Poisson’s effect is applied. The slope of these plots represents $Y_r$.  
5.7 Modified Young’s modulus as a function of the carbon atoms in the CNT (4, 4).  
5.8 Total forces on the left end of the CNT as a function of the number of C-atoms.  
5.9 The total energy-strain plots of all the CNT segments.  
5.10 The force-strain plots of all the CNT segments were stretched without Poisson’s effect. The slope of these plots represents $Y_r$.  

vii
5.11 Young’s modulus as a function of the carbon atoms in the CNT (4, 4) when Poisson’s ratio is not applied. 132
5.12 The total energy-strain plots of all the CNT segments without Poisson’s ratio. 135
5.13 Contour plot representing electron density for CNT (4, 4) with 48 atoms. 138
5.14 Contour plot representing electron density for CNT (4, 4) with 96 atoms. 139
5.15 Contour plot representing electron density for CNT (4, 4) with 144 atoms. 139
5.16 Isosurface plots representing electron density for CNT (4, 4) with 48 atoms. 140
5.17 Isosurface plots representing electron density for CNT (4, 4) with 96 atoms. 140
5.18 Isosurface plots representing electron density for CNT (4, 4) with 144 atoms. 140
5.19 Volume plot representing electron density for CNT (4, 4) with 48 atoms. 141
5.20 Volume plot representing electron density for CNT (4, 4) with 96 atoms. 141
5.21 Volume plot representing electron density for CNT (4, 4) with 144 atoms. 141
5.22 The I-V curve at zero strain for CNT (4, 4) in between two copper electrodes. 146
5.23 The I-V curve at strain=0.16% for CNT (4, 4) in between two copper electrodes. 147
5.24 The I-V curve at strain=0.32% for CNT (4, 4) in between two copper electrodes. 149
5.25 The I-V curve at strain=0.49% for CNT (4, 4) in between two copper electrodes. 150
5.26 The I-V curve at strain=0.65% for CNT (4, 4) in between two copper electrodes. 151
5.27 The conductance as a function of the strain for CNT (4, 4) in between two copper electrodes. 152
List of Tables

Table                                      Page
4.1  Total energies for a 48-atom CNT segment for different mesh cutoffs.  90
4.2  Mesh cut-off parameter values and the total energies for CNT between the
copper surface layers.                                            92
5.1  The associated forces and energies resulted from stretching the C-chain
by different amounts.                                                108
5.2  The metallic carbon nanotube (4, 4) segments used in the project.  111
5.3  Forces and total energies of the stretched CNT (48 C-atoms).      113
5.4  The values of Young’s modulus for the CNT with 48-atoms based on three
definitions of the area.                                            116
5.5  The values of $Y_s$ for all the CNT segments used in the project.  121
5.6  Young’s modulus for SWCNT (4, 4) based on different definitions of the area.
The values were calculated by using the forces.                    122
5.7  The calculated $Y_s$ values from the strain energy.              127
5.8  Young’s modulus for SWCNT (4, 4) based on different definitions of the area.
The values were calculated by using the total energy.               128
5.9  The values of $Y_s$ for all the CNT segments used in the project. Poisson’s effect
was not applied in these calculations.                               132
5.10 Young’s modulus from force calculations for SWCNT (4, 4) based on three
areas and for two situations, with and without applying Poisson’s ratio.  133
5.11 The calculated $Y_s$ values from the strain energy for zero Poisson ratio.  136
5.12 Young’s modulus extracted from the total energies for SWCNT (4, 4) based
on different definitions of the area for two situations, with and without applying
Poisson’s ratio.                                                     136
5.13 Average values of Young’s modulus for SWCNT (4, 4) based on different
definitions of the area for two situations, with and without applying
Poisson’s ratio. These values were calculated by performing the force
and the energy calculations.                                          137
5.14 The atomic information for CNT segments with chiralities (4, 4), (5, 5) and (6, 6).

5.15 The $Y_s$ values for CNT (4, 4), (5, 5), and (6, 6) were calculated by using force and energy calculation when the Poisson’s effect is considered.

5.16 Young’s modulus in TPa for SWCNT (4, 4), (5, 5), and (6, 6) based on different definitions of the area with Poisson’s effect. These values were calculated by performing the force and the energy calculations.

5.17 The current and voltage values at zero strain for the CNT (4, 4) in between two copper electrodes.

5.18 The current and voltage values at strain=0.16% for the CNT (4, 4) in between two copper electrodes.

5.19 The current and voltage values at strain=0.32% for the CNT (4, 4) in between two copper electrodes.

5.20 The current and voltage values at strain=0.49% for the CNT (4, 4) in between two copper electrodes.

5.21 The current and voltage values at strain=0.65% for the CNT (4, 4) in between two copper electrodes.

5.22 The strain values and the associated conductance values for the CNT (4, 4) in between two copper electrodes.
Computational simulations were used in this project in attempting to study the mechanical behavior of different lengths for metallic single-wall carbon nanotubes (SWCNT) under varied stresses. The mechanical parameters of these single-wall carbon nanotube segments were studied by applying a macroscopic or classical stress-strain point of view on the nanoscale level. Different lengths of carbon nanotubes were generated and their properties were studied using the Atomistix software Virtual NanoLab (VNL) and Atomistix ToolKit (ATK) [68]. First principles calculations were used to extract the value of the elastic modulus; it is predicted that the value of Young’s modulus for the carbon nanotube to be in good agreement with the theoretical and the experimental works that have been done. Moreover, the impact of the mechanical stress on the electronic structure and the electronic transport was studied in this work.
Carbon Nanotubes were discovered the first time by Sumio Iijima in 1991 [1], and since that time many studies have been done to predict the electronic structure for them. There is much interest in carbon nanotubes (CNTs) because of their unusual behavior and remarkable properties. They exhibit high conductivity, flexibility, strength and stiffness which give them the opportunity to produce a new generation of high performance composites. Although many theoretical studies and modeling were done to predict the mechanical and electrical properties of the CNTs, few experimental studies were done. All of these theoretical and experimental studies agree that CNTs have a high value of Young’s modulus which is usually taken to represent the mechanical properties of a material. On the other hand, the groups who made these studies each received different values for Young’s modulus due to the variation in the methods used and differences due to the definition of some parameters that were used to define the CNT and Young’s modulus for CNT.

The research work described in this thesis deals with predicting the mechanical parameters of carbon nanotubes starting from a macroscopic stress-strain point of view while using an atomistic-level computational approach. Specific values and numbers for atomic forces, total energies, electron densities, Poisson ratio and Young’s modulus will be presented. Through theory and computation, the basic physics of these parameters will be studied and the mechanical behavior of carbon nanotubes may be predicted according to these parameters. To provide a background for this work, the values of the elastic modulus for carbon nanotubes, the mechanical and the electrical behavior of carbon nanotubes resulting from various theories and methods found in the literature will be reviewed.
1.1 Carbon Nanotubes (CNTs)

By rolling-up graphene sheets with $sp^2$ bond type between the carbon atoms, one can build carbon nanotubes. A single-wall carbon nanotube, cylindrical structure with a cap on each end in some cases, is defined by a number of parameters related with each other in specific relations [2]. The chiral vector ($C_h$), which has the length $L$, circumference of the nanotube, is given in terms of two integers $(n, m)$ in the directions of two basis vectors of the honey-comb lattice $a_1$ and $a_2$ (see Figure 1.1).

![Figure 1.1 Honey-comb lattice and the chiral vector in terms of the basis vectors.](image)

Depending on the $(n, m)$ values, three types of tubules are defined: armchair when $n=m$, zigzag when $m=0$, and chiral nanotube when $0 < m < n$ (see Figure 1.2). Also these two integers determine the electrical conductivity of the carbon nanotube; for a $(n, m)$ carbon nanotube, it is either metallic if the difference between $(n)$ and $(m)$ is a number that is exactly divisible by 3 or semiconducting for the rest of the values of $(n, m)$. 
The uniqueness of the carbon structure and the high strength of it are mainly due to the types of the bonds that are formed. Due to the periodicity of the hexagon pattern in the carbon nanotube, each carbon atom is bonded with other three neighbor atoms. In general, when two carbon atoms combine a covalent bond is formed between them [3]. Covalent bonding in a molecule is produced when each component in the molecule shares one pair of electrons or more; this sharing produces a mutual attraction that holds the molecule components together. In order to fill their outer electron shells, atoms tend to share electrons with each other. These covalent bonds are always stronger than the intermolecular hydrogen bond and are of similar strength to, or stronger than, the ionic bonds [4]. Unlike the ionic bond, the angles in the covalent-bond have an important impact on the strength of the bond which makes the covalent-bond a directional bond. Materials formed by these covalent directional bonds are more difficult to deform than metals. For this reason, non-metals tend to form covalent bonds because covalent bonding most frequently occurs between atoms with similar electronegativities. Figure 1.3 shows how an $sp^2$ orbital is formed [11].


1.2 Carbon Nanotube Properties

The uniqueness of the carbon nanotube in its structure, quasi-one-dimensional nature, cylindrical symmetry and the covalent $sp^2$ bonds formed between the individual carbon atoms gives the carbon nanotube extraordinary strength and elasticity, which provides supreme mechanical properties. For example, the elastic modulus value for carbon nanotubes is higher than the value for diamond. The electrical nature of carbon nanotubes, as mentioned before, is either metallic or semiconducting depending on the helicity indices. Carbon nanotubes have a capacity to carry an electric current 1000 times better than copper wires. Moreover, pressing or stretching the carbon nanotubes can change their electrical properties by changing the quantum states of the electrons in the carbon bonds. This exceptional electrical property opens the way to produce nanotube-based junctions and devices with high electrical conductivity, strength, dependability and efficiency.
Young’s modulus measures the stiffness and flexibility of a material; as the value of Young’s modulus increases the flexibility of the material decreases. For steel Young’s modulus is 0.21 terapascal (TPa). One TPa, $10^{12}$ Pa, is a pressure very close to $10^7$ times atmospheric pressure [5]. For a carbon nanotube, the Young’s modulus value on average is 10 times greater than for steel.

All nanotubes are expected to have a high thermal conductivity along the tube, exhibiting a property known as “Ballistic Conduction” in which the electrons are allowed to flow through the material without any collisions. It is predicted that carbon nanotubes will be able to transmit up to 6000 watts per meter per Kelvin at room temperature; on the other hand, copper which is a well-known metal for its good thermal conductivity transmits $385 \text{ W} \cdot \text{m}^{-1} \cdot \text{K}^{-1}$ at the room temperature [6]. Also carbon nanotubes are thermally stable in high temperatures as high as 2800 degrees Celsius in vacuum and about 750 degrees Celsius in air [6]. Regarding the carbon nanotube’s optical activity, theoretical studies found that the optical activity for chiral nanotubes disappears when the diameter becomes larger [7]. As a result, it is predicted that other physical properties will be affected by this change in optical activity. Optical devices might be produced based on using the carbon nanotube and its optical activity property.

1.3 Carbon Nanotubes Applications

The strength, flexibility, and the high conductivity of carbon nanotubes makes them of potential use in controlling other nanoscale structures, which suggests they will have an important role in energy storage, electronics and other applications.
1.3.1 Energy Storage

The common materials usually used in fuel cells, batteries and other electrochemical applications are graphite, carbonaceous materials and carbon fiber electrodes [8]. One important property that determines the efficiency of the electrochemical devices is the electron transfer rate at the carbon electrodes in these devices; according to some studies, carbon nanotubes have the fastest electron transfer rate [9]. Some examples for applications are listed below [8]:

A. Hydrogen storage: Single-wall carbon nanotubes with their unique geometry and nanoscale are able to store a liquid or a gas in their cores which gives them the ability to exceed the level of energy storage in the fuel cells.

B. Lithium intercalation: The efficiency of the Li batteries is determined by the saturation concentration of the electrode materials. It has been found that this saturation is the highest for Li when it is used in carbon nanotubes.

C. Electrochemical super capacitors: Energy storage based on generating a charge-separated situation in electrochemical double layers. Carbon nanotubes are used in this application as electrodes because the energy storage depends inversely on the charge separation which is provided by the small dimensions of the carbon nanotubes.
1.3.2 Carbon Nanotube-based Electronics [8]

Carbon nanotubes have many properties—from their unique dimensions to an unusual current conduction mechanism—that make them ideal components of electrical circuits and applications. A few applications that use carbon nanotubes in their structure are listed below.

A. Field emitting devices: Solid materials emit electrical current when exposed to a high electric field. This current depends on the strength of the electric field, the emission surface and the work function of the surface. Carbon nanotubes are considered as perfect emitters while also displaying high electrical conductivity.

B. Transistors: Carbon nanotube-based transistors are divided into two types. The first one is shown in Figure 1.4 and is called a CNT field-effect-transistor (CNTFET). CNTFET consists of a drain, a source and a single-semiconducting carbon nanotube which works as a channel connecting between them. A third electrode is placed on top of the nanotube.

![Figure 1.4 Set-up of CNTFET [8].](image)
The second type is the CNT single-electron-transistor (CNTSET), as shown in Figure 1.5. The two ends of the carbon nanotube are connected with electrodes, and the ends are covered with a material that keeps them P-doped. The middle part is also doped with another type of material that makes the middle part n-doped.

![Figure 1.5 p-n-p carbon nanotube device [8].](image)

C. A conducting carbon nanotube can be used as a tip for the Scanning Tunneling Microscope and Atomic Force Microscope because it has a high elasticity which prevents it from being crashed with the surface (see Figure 1.6).

![Figure 1.6 MWCNT as a tip [8].](image)
1.4 Project Goals and Thesis Overview

In this project, the stress, strain, elastic modulus, electron density, and total energy are found and studied for different carbon nanotube segments and for different strains. Atomic forces on the C-atoms were extracted using the Atomistix software package. Different lengths of metallic carbon nanotubes (n=4, m=4) are constructed and modeled using the Atomistix ToolKit (ATK) and some of the analyses are done by the Virtual NanoLab (VNL). Other analyses are done using a first principles calculation. A primary focus is on calculating the elastic modulus by using the total energy, force, stress, and strain classical relations. Also the electron densities are studied for these strained carbon nanotube segments. The electrical conductance for a strained carbon nanotube in between two copper electrodes is briefly studied by calculating the current values for different voltages that are applied through the copper electrodes. The ultimate goals are to evaluate the elastic parameters of the carbon nanotubes by using atomistic calculations and to compare these values with the other works that have been done before, and to study the impact of stretching the carbon nanotube on the electron density in the CNT and on its transport properties.

Chapter 2 deals with some math review and definitions for the mechanical parameters: Young’s modulus, Poisson ratio, stress, strain, strain energy and the relations between them. In addition to these definitions, a literature review is presented for some methods used to understand the mechanical behavior of the carbon nanotube. Some values for the mechanical parameters are presented from other groups’ work. The second section in this chapter deals with the definition of the electrical conductance. At the same time it provides a literature review about the transport properties in the carbon nanotubes.
Chapter 3 describes the Density Functional Theory (DFT) which is considered to be the major tool for calculating electronic structure in condensed matter [10]; using this method in calculating the electronic structure leads to using some important parameters that are discussed in separated subsections in the chapter. This chapter also talks about the Non-Equilibrium Green’s Formalism (NEGF), which is a method used to calculate the transport properties for a material. The last section of this chapter is about the Atomistix software that is being used in the calculations of the electronic structures for the carbon nanotubes. It describes the main parts and the functions of the software while discussing the input and the output files.

Chapter 4 focuses mainly on the methods that were used in this research and on the procedure that was done to perform the calculations. It lists the nanosystems that were used and how to set-up each one of them using the nanolanguage provided in the software. Also it discusses in detail how to set up an input file using python scripting to extract the atomic forces, energies and the electron densities. The main idea discussed in this chapter is the hypothesis of stretching the carbon nanotube and the methods that were followed to calculate the forces, and Young’s modulus.

Chapter 5 presents the results, the calculations and the software output files. Graphs for forces, energy, electron density and others are shown.

Chapter 6 shows the conclusions and discussions about the results that have been presented and the future related work.
Chapter 2

Literature Review of Carbon Nanotube Properties

2.1 Mechanical Parameters

In this section, a math review will be done on the mechanical parameters that are related to the carbon nanotubes. Some findings and results regarding these parameters will be presented from theoretical and experimental works that have been done in this area.

2.1.1 Definitions of the Mechanical Parameters

The mechanical behavior for any solid material is usually described by a list of parameters; some of them are discussed in the following pages:

A. Stress (\( \sigma \)) is the measure of the average amount of applied force or system of forces that tends to deform the material per unit area.

\[
\sigma = \frac{F}{A}
\]  (2.1)
If the SI units are used then the unit of the force is Newton and the unit of the area is square meters. Thus, the stress is given in Pascal (Pa).

**B. Strain \( (\varepsilon) \)** is the geometrical expression of the deformation produced by the stress on a solid material. If the material goes from one situation to another due to an applied force, then the strain is calculated by finding the length between two points in the material in those two states.

\[ \varepsilon = \frac{\Delta \lambda}{\lambda_0} = \frac{\lambda - \lambda_0}{\lambda_0} \]  \hspace{1cm} (2.2)

Where: \( \lambda \) is the new length and \( \lambda_0 \) is the original length. Thus the strain is dimensionless.

**C. Young’s Modulus \( (Y) \)** is one of several elastic moduli; for example, bulk modulus for volume deformation, and shear modulus for surface deformation. Young’s modulus gives information about the stiffness of a material and it is defined as the Stress/Strain Ratio when Hooke’s law is valid; Young’s modulus has the unit Pascal (Pa). It can be found by calculating the slope of the stress-strain curve. Also, Young’s modulus is defined as the second derivative of the energy \( (E) \) with respect to the strain divided by the volume \( (V) \).

\[ Y = \frac{\text{Stress}}{\text{Strain}} = \frac{\sigma}{\varepsilon} = \frac{F/A}{\Delta \lambda/\lambda_0} \]  \hspace{1cm} (2.3)

\[ Y = \frac{1}{V} \frac{\partial^2 E}{\partial \varepsilon^2} \]  \hspace{1cm} (2.4)
Hooke’s law states that the extension produced in a material due to a load is proportional to that load:

\[ F = -K(\Delta X) \]  \hspace{1cm} (2.5)

K here is a force constant.

**D. Poisson Ratio \( (\nu) \):** When a sample of material is stretched in one direction, it tends to contract in the other two directions. Poisson's ratio is defined as the ratio of the contraction strain (normal to the applied force) divided by the extension strain (in the direction of the applied force). If the applied force in y-direction then:

\[ \nu = \frac{\varepsilon_{\text{contraction}}}{\varepsilon_{\text{extension}}} = \frac{\Delta x/x}{\Delta y/y} \]  \hspace{1cm} (2.6)

**2.1.2 Literature Review of the CNT Mechanical Parameters**

This literature review brings together the work of different groups working to find the mechanical properties of carbon nanotubes. Apparently, from the work that they have done, it is clear that the elastic parameters of the carbon nanotube depend on its structural parameters, and there is no agreement on a certain value for Young’s modulus for carbon nanotubes. Since each group used a different methodology to calculate the mechanical properties for CNT, then each group found that this value of Young’s modulus depended on different parameters. However, all of them have the same order of magnitude for the value of Young’s modulus.
Lu (1997) used the force-constant model to calculate the elastic parameters and to calculate Young’s modulus for single-wall CNT, multi-wall CNT and nanoropes for different chiralities [12]. The force-constant model approximates the atomic interactions around the equilibrium configuration by the addition of the pairwise harmonic potentials between the atoms. Then the elastic parameters are found by fitting the force constants. The most structural parameter that affects the calculations in this model is the thickness of the CNT. The average value of Young’s modulus for both single-wall and multi-wall CNT was ~1TPa, which does not depend on the size and the helicity, and it is very close to the diamond’s value 1.063TPa [13]. Poisson’s ratio was found to be 0.28 for SWCNT and 0.27 for MWCNT.

A first-principles method was used by Zhou et al. (2000) to predict the mechanical properties of single-wall carbon nanotubes [14]. The linear combination of atomic orbitals (LCAO) method is utilized by the molecular orbital (MO) cluster model to study the structure and the characteristics of carbon nanotubes. Based on the first-principles calculations, Young’s modulus is given as the second derivative of the total energy with respect to the stress divided by the equilibrium volume. Tensile strength, which is the maximum stress that can be applied on a material without disturbing its equilibrium, can be defined as the first derivative of the total energy with respect to the strain divided by the cross-sectional area of the carbon nanotube. The calculations gave a 0.764 TPa as the Young’s modulus value for the single-wall carbon nanotube and 1.02 TPa for graphite sheet. The tensile strength was calculated as 6.249 GPa for nanotube, and the Poisson ratio was found to be 0.32, which is comparable to Lu’s (1997) result of 0.28. The results above show that carbon nanotubes and graphite have the same order of
magnitude (TPa) in the moduli, which proves that both structures have the same nature of bonds. On the other hand, the small difference between the moduli shows differences in the chemical bonding in both structures. The nanotube’s binding energy is less than the graphite’s, which is a result of rolling-up the graphite sheet to construct the nanotube. This leads to make the Young’s modulus and the tensile strength for the carbon nanotube less than the values for the graphite sheet.

Since it is hard to produce one single CNT experimentally, the majority of the experimental works were done using carbon nanoropes. Nanoropes are a collection of a large number of single-wall or multi-wall carbon nanotubes or both together. Yu et al. (2000) studied the mechanical behavior of nanoropes consisting of 15 single-wall carbon nanotubes (SWCNT) [15]. Those ropes were tensile loaded inside the scanning electron microscope (SEM). An atomic force microscope (AFM) was used to apply the force on the ropes through its tip; at the same time, it was used to read the applied load. The video output of the SEM was used to record the whole tensile-loading. After the sample was mounted inside the AFM, the tensile force applied to the SWCNT was calculated by measuring the deflection in the AFM tip for a given voltage, and that was multiplied by the force constant of the tip. As a result of this process, Young’s modulus was found to range from 0.32 to 1.47 TPa, and the average breaking strength values were from 13 to 52 GPa. The maximum breaking strain was 5.3% for this sample which is close to the theoretical value of 5% that was found by Nardelli (1998) [16]. Another assumption in this method is to take the cross-sectional area to be the area of the Close-packed SWCNT rope instead of the area of the whole rope. According to this assumption, the average values of the breaking strength and Young’s modulus were 0.003 and 1 TPa respectively.
E. Hernandez et al. (1998) used a nonorthogonal tight-binding formalism to compute the value of Young’s modulus for SWCNT [17]. Young’s modulus was found to be almost constant for a large nanotube diameter and has the value 1.26 TPA, which is in a good agreement with Wong’s et al. finding [18] of 1.28 as an average value of Young’s modulus. On the other hand, Young’s modulus depends on the radius for the small nanotubes which contrasts what Lu [12] predicted. The Poisson ratio had an average value of 0.26 according to these calculations. Young’s modulus was calculated using first-principle calculations:

\[ Y = \frac{1}{V_0} \frac{\partial^2 E}{\partial \varepsilon^2} \bigg|_{\varepsilon=0}, \]  \hspace{1cm} (2.7)

Here the equilibrium volume \( V_0 = 2\pi LR\Delta R \), \( L \) is the length, \( R \) is the tube radius and the CNT thickness \( \Delta R = 0.34nm \) according to Lu [14] and \( 0.066nm \) according to Yakobson et al. [19]. The values given above are based on Lu’s value of the CNT thickness. However, this group defined a new parameter to introduce the stiffness of a SWCNT without using the thickness of it:

\[ Y_s = \left( \frac{1}{S_0} \right) \frac{\partial^2 E}{\partial \varepsilon^2} \bigg|_{\varepsilon=0}, \]  \hspace{1cm} (2.8)

Here \( S_0 \) is the tube’s surface-area at the equilibrium (see Figure 2.1 which shows the results that were received by this group for the \( Y_s \) as a function of the CNT diameter). Based on this definition:

\[ Y = \frac{Y_s}{\Delta R} \]  \hspace{1cm} (2.9)
Figure 2.1 $Y_e$ as a function of the diameter [17].

Different values of wall thickness and Young’s modulus are due to applying different methods and relations in the continuum elasticity theory (CET) which give more opportunities to use more methods to calculate these parameters. Zhou et al. (2000) obtained the effective wall thickness and Young’s modulus of SWCNT from the strain energies [20]; Young’s modulus was about 5 TPa and the wall thickness about 0.7 Angstrom which is close to what Yakobson et al. have calculated $Y=5.5$ TPa and thickness=0.66Å. It was found that these two values are insensitive of the radius and the helicity of SWCNT.

The experimental results regarding the elastic moduli of carbon nanotubes are based mainly on the atomic force microscopy (AFM) and the high-resolution transmission electron microscopy (HRTEM). The experimental results due to using these techniques confirm the results from the theoretical predictions, which are more advanced because of the difficulties involved in the manipulation of nanometer-sized systems. The history of the experimental work has been started by Treacy et al. [21] when they used
(TEM) and measured the thermal vibration to obtain the Young’s modulus for a multi-wall carbon nanotube. They found this value to be $1.8\pm0.9$ TPa. After that, and by using (AFM), Wong et al. [18] found this value of Young’s modulus to be $1.28\pm0.59$ TPa and it depends on the diameter of the carbon nanotube. Later, Yu et al. [22] measured Young’s modulus between 0.27 and 0.95 TPa by using the same techniques. On the other hand, Krishnan et al. [23] applied the method of Treacy et al. [21] on a SWCNT that has a diameter between 1.0 and 1.5 nm. They measured the mean value of the elastic modulus for this SWCNT to be of 1.25 TPa.

A theoretical prediction of the mechanical parameters can be done either by computing the mechanical response of the carbon nanotube or by deriving them analytically. The computational methods have been used as strong tools to find these mechanical properties of the carbon nanotubes. For example, the classical molecular dynamics (MD) and ab initio methods were the most used tools. Yakobson and his group [19] used the MD method, and found that the Young’s modulus is about 5.5 TPa. Using the ab initio methods gives more accurate values than the MD tool. Lier et al. [24] have predicted a value of higher than 1TPa for the Young’s modulus by using the ab initio method.

Overney et al. [25] calculated Young’s modulus using an empirical Keating Hamiltonian with parameters determined from first principles in the computation. According to their calculation, Young’s modulus was ranging from 1.5 to 5.0 TPa. Another theoretical investigation was done by T. Natsuki et al. [26] to study the most important mechanical properties of SWCNT and MWCNT as the elastic modulus and Poisson’s ratio. They proved in their theory that the elastic modulus of MWCNT does not
depend on the tube diameter, but it depends on the number of layers, and the value for the elastic modulus is between 1.6 and 0.8 TPa. To compare their work with other’s work, Tu et al. [27] calculated this value in the range of 1.7 to 1.05 TPa for the double wall carbon nanotube. Regarding the SWCNT calculations, they reported that the elastic modulus increases if the nanotube diameter decreases, and has the value 1.1 to 0.73 TPa. They also showed that Poisson’s ratio depends on the chirality or the tube diameter of the CNT, and for the SWCNT, its diameter increases Poisson’s ratio decreases till it reaches a constant value of 0.27 as shown in Figure 2.2.

![Figure 2.2 Poisson’s ratio dependence on the nanotube diameter [26].](image)

Another important method of calculating Young’s modulus is using the strain energy of the carbon nanotube. By computing the second derivative of the energy, Gao et al. [28] obtained the values of Young’s modulus from 0.64 to 0.67 TPa.
2.2 Electron Transport in Carbon Nanotube

Carbon nanotubes, as mentioned before, are either metallic or semiconducting depending on their chirality. For example, all armchair carbon nanotubes (n=m) are metallic. One of the most important electron transport [29] properties is conductance (G), which relates the total current (I) that flows through an object to the voltage drop (V) across it.

The electrical conductance (G) is the real part of the reciprocal of electrical impedance (resistance). It is a measure of how easily electricity flows along a certain path through an electrical element. The SI derived unit of conductance is the siemens (formerly referred to as the reciprocal ohm or mho).

Experimentally, in one technique, a Scanning Tunneling Microscope (STM) is used to measure the conductance $G = I/V$ of a nanotube which is directly related to the electronic density of states and the band gap [5]. When a single-wall carbon nanotube is situated between two metal contacts, measurements show a steplike relation between the current in the nanotube and the applied voltage between the two ends, i.e., the conductance exhibits quantization.

2.2.1 Electron Transport for Strained CNTs

It is expected for the conductance (G) of a single-wall carbon nanotube to change when the nanotube goes under a stretching process. This change in the conductance is due to changes in the energy bandgap in the carbon nanotubes [30-32]. Some experimental work [33, 34] and other theoretical studies [35] have been done to predict
the electron transport response to the applied load on the carbon nanotube. Tombler et al. [36] used the tip of the (AFM) as an applied load on a single-wall carbon nanotube to measure the resistance of the nanotube under this load. They found that the resistance increased by two orders of magnitude when applying a strain of 3%. Maiati et al. [37] predicted that the conductance of armchair nanotubes is less affected by the tensile strain than the zigzag nanotubes. Also they found that the changes in the conductance due to the strain depend on the band structure. For multi-wall carbon nanotubes, Dohn et al. [38] used microcantilevers to study the conductance response to the applied load. Initial conductance for the multi-wall carbon nanotube was $3G_0$, where the quantum conductance $G_0 = 2e^2/h = (12.9 k\Omega)^{-1} = 77 \mu S$. This conductance was decreasing linearly while the strain was increasing; for a maximum strain of 0.6%, the conductance decreased by a factor of four as shown in Figure 2.3 below:

![Figure 2.3 Conductance as a function of strain for MWCNT](image)

**Figure 2.3** Conductance as a function of strain for MWCNT [38].
The conductance of one metallic single-wall carbon nanotube can be given as $G = 2G_0$ at zero-strain and this value is changed when the carbon nanotube is stretched. The change in the electrical conductance comes from the change in the band gap which depends on the chirality and the band structure of the carbon nanotube and other structural parameters. However, there is an agreement between the experimental and the theoretical studies that the conductance of the carbon nanotubes decreases while increasing the strain on the carbon nanotube.
Chapter 3

Theoretical Foundations and Software

The physical, chemical, electrical and optical properties of a solid are determined by quantized electronic states. These quantum states in low-dimensional systems are produced due to the confinement of the carrier charges in one or two dimensions. For example, in nanotubes electrons propagate freely only along the nanotube axis because of this quantum confinement in two dimensions. The long-range periodicity and natural symmetry in the solid systems produce forbidden energy regions among the bands of these quantized states. In order to study these energy bands and to find the electronic structure for the material, the relation between the energy and the Bloch wave vector ($k$), the Schrödinger equation must be solved. Basicly, the process of calculating $E(k)$ permits computation of the density of states (DOS), the electron density, and electronic wave functions. For instance, the number of sub bands in the conduction and valence band depends on the standing wave solutions for the electronic wave functions in the nanotube.
Hence, the approximations and assumptions used in theoretical modeling of the nanoscale systems and analyzing their properties are more restricted compared to bulk solids.

In this chapter, two theoretical approaches for calculating the electronic structure and the transport properties for carbon nanotubes will be described. These two methods are Density Functional Theory (DFT) and the Non-Equilibrium Green’s Function (NEGF) method. The discussion of these two methods will be followed by a description of the Atomistix software, the software that uses these methods to extract the electronic structure and calculate current-voltage curves for different systems. Some parameters that are used in DFT and NEGF will be discussed, and their influence on the calculations will be emphasized.

### 3.1 Density Functional Theory (DFT)

Different approximations derived from quantum mechanical methods are used to calculate the electronic structure for carbon nanotubes. Some methods do not treat the electron-electron interaction explicitly; two good examples are the Nearly Free Electron (NFE) model [39] and the tight-binding method [40]. Another approach is the Hartree equation [39] which takes the electrostatic interaction between the electrons into consideration. This approach was extended by the Hartree-Fock methods where the exchange effects are important [39]. The system energy error remaining in the Hartree-Fock approach is referred to as the “correlation energy”. The density functional theory (DFT) was created to include the correlation effects without using very costly wave function methods. W. Kohn and L. J. Sham [41] were the first investigators to formulate an efficient computational method for implementing the density functional theory (DFT),
which is the basis for many atomistic-scale simulations in science. In 1998 Walter Kohn
was awarded the Nobel Prize in chemistry for his development of DFT.

An essential theorem in DFT is the Hohenberg-Kohn Theorem [42] which obtains
the energy as a functional of the density of the ground-state electrons. The ground-state
energy is found by varying the density to obtain the minimum energy. Implementation of
DFT is accomplished through the Kohn-Sham approach [41] in which a non-interacting
electron reference system with an effective potential is defined and has the same
eigenvalues as the multi-electron system. Solution of this single electron Schrodinger
equation yields the solution to the many-body problem; however, the effective potential
in this equation depends on the electron density. Since finding the electron density
requires solving for the eigenfunctions, a self consistent calculation is necessary. The
accuracy of the DFT calculation depends in part on the degree of convergence of this self
consistent calculation.

The software that is used in this project employs DFT to calculate the energy,
electron density and other physical properties for many-atom systems and molecules. For
this work, the calculation of the total energy and the atomic forces derived from the total
energy are of particular interest. The total energy consists of the sum of the kinetic energy
of the non-interacting electron gas, the mean-field electrostatic energy of the electron gas
and the ions, and the exchange-correlation energy of the electrons. Atomic forces are
found through an analytical differentiation of the total energy terms. Exchange-
correlation is treated within the Kohn-Sham DFT [41] and users can select the local
density approximation [45] (LDA) or the generalized gradient approximation [46]
(GGA). This software is also able to use the standard norm-conserving pseudo potentials
in their non-local form [43, 44]. Minimal basis sets are selected to execute accurate calculations; for example, it uses single or multiple-zeta polarized basis sets. The use of these parameters and others is discussed in detail below. More detailed discussions of density functional theory, including the relevant equations and methods are available in the literature [10, 47-49] and on the internet.

A set of calculation parameters and approximations must be defined in order to use the DFT method. Some of these important parameters are discussed below in detail.

A. Basis Sets

In electronic structure calculations, plane wave basis sets are convenient to use because of their orthogonality and their simplicity in calculating the kinetic energy but often require large numbers of plane wave functions for satisfactory results. In general, the Kohn-Sham equations are solved by expanding the wave functions using localized, numerical orbitals as basis sets [50]. The basis functions can have multiple radial (zeta) and spherical (polarization) functions. Five different kinds of basis set are listed below [51] in order of increasing size and accuracy.

- Single-zeta: Least accurate; a single basis orbital for each valence orbital.
- Single-zeta polarized: Single-zeta plus one basis orbital for the first unoccupied shell.
- Double-zeta: two basis orbitals for each valence orbital.
- Double-zeta polarized: Double zeta plus one basis orbital for the first unoccupied shell.
- Double-zeta double polarized: most accurate; double-zeta and two basis orbitals for the first unoccupied shell.
The number of basis functions per electron affects the convergence of energy calculations. Experience has shown that the Double-zeta polarized basis set gives the most accurate results, but for large systems this basis set might result in a large computational time. For the current work a test on this parameter will be done to find out the best basis set to represent the orbitals while yielding acceptable execution times.

B. K-Points and the Brillouin Zone

Fourier transforms can be used to represent periodic systems by determining a discrete set of Fourier components. The set of such “k-points” is called the reciprocal lattice. The dot product of a reciprocal lattice vector and a direct primitive lattice vector is $2\pi$. The Brillouin zone (BZ) is a cell on the reciprocal lattice related to the direct lattice or crystalline unit cell, and it has certain characteristics [52].

Bloch’s theorem [53] states that the difference between two eigenstates in two different cells is the factor of $e^{ik\mathbf{r}}$ where $k$ is the $k$-point that corresponds to the cell. Each $k$-point outside the Brillouin zone is related to one $k$-point inside the Brillouin zone by the phase factor. Thus, the total energy per unit cell of a periodic structure can be obtained by averaging the energies over all the $k$-points within the Brillouin zone of the occupied eigenstates. Modeling a small number of well chosen $k$-points within the Brillouin zone can approximate this average. Atomistix software uses the method described by Monkhorst and Pack [54] to determine the $k$-points to be modeled [47]. Using this method, the $k$-points are given by:

$$k(n_1, n_2, n_3) = \sum_{i=1}^{3} \frac{2n_i - N_i - 1}{2N_i} G_i$$

(3.3)
Here, \( G_i \) are the reciprocal lattice vectors, the \( N_i \) are specified by the user, and the \( n_i \) are integers that range from zero to \( N_i \). The total number of k-points given by this method is \( \prod_{i=1}^{3} N_i \). The tuple of integers \((n_1, n_2, n_3)\) determines the number of k-points to be used along each respective reciprocal vector. For bulk systems and electrodes, the three numbers should be given.

C. Mesh Cut-Off

A real space grid is used to calculate the external potential and density through DFT method. The spaces of this grid are given in terms of the energy cut-off that would result in the same resolution on the reciprocal lattice. This function defines the mesh (grid) used to represent real-space quantities when calculating integrals and solving the Poisson equation. Mesh cut-off parameter is given as an energy \( E \), which implicitly defines the grid spacing \( \Delta x \) [55].

\[
E = \frac{P^2}{2m_e}
\]

But

\[
P = v k = \frac{h}{2\pi} k, \]

\[
k = \frac{2\pi}{\lambda}
\]

Then

\[
E_{(Joul)} = \frac{h^2}{2m_e \lambda^2}
\]

\[
\frac{\lambda}{2} = \Delta x = \frac{h}{2\sqrt{2m_e E_{(Joul)}}},
\]
here the grid spacing is taken as the half wavelength of a plane wave for the same electron energy. The values for the constants in the above equation are:

\[ h = 6.63 \times 10^{-34} \text{ J} \cdot \text{s} \]
\[ m_e = 9.11 \times 10^{-31} \text{ Kg} \]

\[ 1 \text{ Rydberg} = 13.6 \text{ eV} = 2.18 \times 10^{-18} \text{ Joule} \]

so that the grid spacing may be written as

\[ \Delta x = \frac{2.46 \times 10^{-10}}{\sqrt{E_{(\text{Joul})}}} \text{ Å} \quad (3.4) \]

OR

\[ \Delta x = \frac{\pi}{\sqrt{E_{(\text{Rydberg})}}} \text{ Bohr} \quad (3.5) \]

\[ 1 \text{ Bohr} = 0.5292 \text{ Å} \]

The greater the energy the smaller the grid space which leads to higher accuracy in the total energy value and the calculations. At the same time, it consumes more computational time for the high energy value. In this work a test will be done to find the smallest value of this parameter at which the total energy stabilizes.

D. Exchange Correlation Functional

The exchange correlation functional contributes to the main approximation in the DFT method. Three different functionals involving the local density approximation and the generalized gradient approximation are available when using this method and are listed below. Generally, energy changes are more affected by these functionals than the structural properties of a system. The revPBE GGA functional usually performs better
than the LDA functional; however, in Atomistix software the LDA-PZ functional, which is the default, offers slightly better performance.

The three types of exchange correlation functionals provided by DFT are [56]:

- **LDA-PZ**: The local density approximation (LDA) with the Perdew-Zunger parametrization [57] of the correlation energy of a homogeneous electron gas calculated by Ceperley-Alder [58].
- **PBE GGA**: Perdew-Burke-Ernzerhof parametrization of the generalized gradient approximation (GGA) [59].
- **revPBE GGA**: Revised Perdew-Burke-Ernzerhof parametrization of the GGA [60].

These are available in both spin-unpolarized and spin-polarized forms.

### 3.2 Non-Equilibrium Green’s Formalism (NEGF)

Non equilibrium Green’s function methods [61, 62] are usually used as a basic theory to build quantitative models for studying the transport properties in nanoscale conductors under voltage difference [63-65] and for quantum device simulation [66, 67]. For example, it is used for calculating current, effective potential and charge densities. In order for any program to simulate a quantum device, a self-consistent solution of a transport equation and a Poisson equation must be performed. The calculations of the electron density and the effective potential are iterated till they converge to a self-consistent value.
In this project, the transport calculations are done when a voltage difference is applied across a channel (single-wall carbon nanotube) placed between two metallic contacts.

The modification of the channel by the presence of the semi-infinite electrodes is included in “self-energy” terms in the Hamiltonian and the injection and extraction of electrons as represented by “source” and “drain” terms. To calculate the total wave function of the channel, Green’s functions are applied for the whole system:

\[ G(E) = [EI - H - \Sigma_1 - \Sigma_2]^{-1} \]  \hspace{1cm} (3.1)

Here \([\Sigma_{1,2}]\) are defined as the self-energy matrices. These matrices describe the broadening and shift in the energy levels in the contacts due to the coupling. \([H]\) is the Hamiltonian matrix, and \([I]\) is the identity matrix. These Green functions are used in solving the Schrodinger equation to compute the self-energy matrices, “broadening” matrices, overall wave functions, and the density matrix. The transmission function \(T(E)\) is then calculated from these matrices and used to find the current density:

\[ I = \frac{4e}{h} \int_{-\infty}^{+\infty} T(E)[f_1(E) - f_2(E)]dE \]  \hspace{1cm} (3.2)

In this equation, \(f_1(E)\) and \(f_2(E)\) are the Fermi functions in the source and the drain at the appropriate potentials.
3.3 Atomistix Software

The major source of the atomic data, which are presented in this research, is from the Atomistix software package [68]. This software consists of two main programs; the first one is Virtual NanoLab (VNL) which is used as graphical user interface (or GUI). The second one is the Atomistix ToolKit (ATK); it contains the DFT, NEGF, and auxiliary codes for the primary computations and executes user scripts to generate the final results. VNL and ATK are considered as state of the art tools for the simulation and visualization of nanoscale systems. They can be used to predict the mechanical, electrical, optical, and magnetic properties of systems. Python scripting is used as the code compiled in serial or parallel execution (under MPI) to direct the calculation of these properties.

The Center for Computational Nanoscience (CCN) in the Department of Physics and Astronomy at Ball State University maintains a copy of the latest version of the Atomistix software on a Beowulf cluster [70] which consists of 64 nodes. These nodes are divided into two groups; in the first 32, each node uses two 32-bit Intel Xeon [71] processors. In the second 32 nodes, each one uses two 64-bit Intel Xeon processors. The Beowulf cluster runs on the RedHat [72] distribution of the Linux 4 operating system.

In the following subsections, a general description of the features of Virtual NanoLab (VNL) version 2008.02 is followed by an introduction to the Atomistix ToolKit (ATK) version 2008.02 and the related scripting for complete and direct specification of parameters and program execution. The descriptions of VNL and ATK are based on information from the Atomistix manuals and tutorials and from experience in using the software,
3.3.1 Virtual NanoLab (VNL)

Virtual NanoLab (VNL) is a graphical interface that provides a group of modeling tools; the function of these tools is to set-up, investigate, and study nanoscale structures such as molecules, bulk and two-probe systems. The quantum mechanical equations that describe these systems are solved by VNL by using advanced software architecture and numerical methods to implement “ab initio” calculations. VNL simulates the electronic structure and the transport properties for the different systems based on two main techniques discussed before in this chapter; those techniques are density functional theory (DFT) and non-equilibrium Green’s functions (NEGF). All the calculations performed in VNL are done by the Atomistix ToolKit (ATK) which is the main engine for computing the scripts. VNL allows the user to set up virtual experiments with the general work flow shown in Figure 3.1 that resembles an actual experimental process.

![Figure 3.1 VNL: Work-Flow [69].](image)
The tools that are used to set up systems can be easily controlled on the Lab Floor. The first window that appears when the VNL is launched is called the Lab Floor (see Figure 3.2):

![Lab Floor in the Virtual NanoLab.](image)

**Figure 3.2:** Lab Floor in the Virtual NanoLab.

As shown in the figure above, the lab floor consists of a variety of tools with a different function for each one. These tools are:

1. Nanoscope.
4. Method Editor.
5. Atomic Manipulator.
6. Crystal Cupboard.
8. Script Editor.
9. Result Browser.
10. Job Manager
The Nanoscope

The main job of the Nanoscope is to visualize the atomic geometries and the calculated properties in 3D space. In order to visualize an object by using Nanoscope, this object must be dragged from the directory that contains it and dropped into the Nanoscope. That file must have the extension (.vnl) in order to be viewed by the VNL-tools. Some prepared examples are provided by VNL, and Figure 3.3 shows one of these examples viewed by the Nanoscope. By using the “Properties” choice in the context menu, the user can change the size of the atoms and the thickness of the bonds. The view of the molecule can be changed and zoomed and the molecule itself can be rotated and moved by using the mouse buttons. The final image can be exported or printed by selecting the option “Camera” from the same menu.

Figure 3.3 A water molecule visualized in Nanoscope. The context menu can be opened by right-clicking anywhere in the Nanoscope window.
Molecular Builder

Molecular Builder can construct different molecules that can be used by other VNL tools. Advanced features are provided by this tool to build simple and complex molecular structures. Some molecular properties like the bond lengths, the hybridization types, and other properties are set by the Molecular Builder. Figure 3.4 shows the main screen of the Molecular Builder. More advanced work concerning molecules can be done in this tool [73].

Figure 3.4 A molecule constructed by Molecular Builder.
NanoLanguage Scripter

The main function of this instrument is to produce complete calculation set-ups for the systems that are produced by Molecular Builder, Nanotube Grower, and other tools. After these calculation set-ups are produced, they are saved as nanolanguage script. The general work-flow of the NanoLanguage Scripter is as following:

1. Drag the system from the Lab-Floor or from its directory and drop it in the NanoLanguage Scripter (see Figure 3.5).
2. Define the parameters that control the DFT and NEGF calculations for the configuration by using the Method tab.
3. Determine the physical quantities that should be extracted from the converged results and presented as output files by choosing them from the list in the Analysis tab.
4. Create the script and/or save it.
5. Execute the script using the Job Manager on the VNL or by using ATK directly.

A VNL file can be created by specifying its name under the tab “Analysis” so that this file can be used later in the analysis or to show the output results.
Method Editor

Method Editor is used in the NanoLanguage Scripter when producing nanolanguage scripts to predefine DFT and NEGF parameters. It provides a fast way through the drag and drop action to set-up and store calculation parameters for the optimization of the electron density.

Figure 3.5 NanoLanguage Scripter screen and tabs.
For certain systems the calculation parameters can be defined using this tool as the following:

1. Open the Method Editor from the Lab Floor and choose the right system from the top-right corner as in Figure 3.6.

2. From the drop-down menu, specify the calculation parameters to be used. Some important parameters are listed below and will be discussed in detail when describing ATK.

   A. Basis Set.
   B. Energy Shift in the basis set dialog.
   D. Exchange Correlation Functional.
   E. Tolerance under the Iteration Control parameters.
3. Save the settings to a certain location in the system to be used later.

4. Open the NanoLanguage Scripter for the chosen system as in the previous section.

5. Drag the saved Method Editor-file and drop it in the NanoLanguage Scripter-screen. The parameters that were defined in the Method Editor will now be set under the tab “Method” in the scripter.

6. The last step will be to choose the physical properties to be calculated by using the tab “Analysis” in the Scripter. Next, save the script and run it using the Job Manager or the ATK.

Choosing the right parameter is important for receiving accurate results. For different type of nanosystems and different atoms there are different sets of parameters.

**Atomic Manipulator**

Atomic Manipulator tool is used to construct and manipulate two-probe systems. It can also be used to easily build new molecules by inserting atoms at their positions. In addition, existing molecules can be imported to the Atomic Manipulator to change their structures or to add them to other molecules. After defining a number of atoms, they are treated as one molecule that can be moved or rotated. On the other hand, the complicated systems like two-probe system can be defined as more than one molecule, and these molecules can be defined separately and then put together distorting the internal configuration of the system. A molecule can be inserted or deleted by right clicking in the grey area and selecting the proper choice from the context menu (see Figure 3.7).
The constructed system can be manipulated in the preview window to the right. This window has the same properties as the one in the Nanoscope tool where the molecule can be zoomed, and rotated in addition to other actions; for example, it can produce the electrodes for the two-probe system as seen later in this section. The molecule also can be rotated or translated more accurately by selecting the Translate and Rotate from the context menu, or directly by specifying the origin and orientation of the group.

![Atomic Manipulator](image)

**Figure 3.7** Atomic Manipulator.
Crystal Cupboard

Crystal Cupboard represents the database of bulk crystals; it has more than 500 predefined crystals. These bulk crystals are ready to be generated as separate scripts and can be used directly in constructing other systems or in the calculations. They also can be used as templates by changing molecular parameters such as the bond length and the lattice constants. The lattice constants and the bond length usually are provided in the bottom-left corner of the Cupboard-screen (see Figure 3.8); these values correspond to experimental values. The preview window in the bottom-right box shows the atoms in the basis and the primitive unit cell spanned by the three primitive lattice vectors.

Figure 3.8 Crystal Cupboard.
The script of a chosen crystal can be saved and then dragged and dropped in the NanoLanguage Scripter. Next, run the script to extract the physical properties for the bulk. Also the Nanoscope can be used to visualize this crystal in greater number of atoms and larger volume by using the option of repetition which is available under Bulk Atomic Configuration in the properties dialog in the Nanoscope. This repetition option is also available in the Crystal Cupboard tool.

**Nanotube Grower**

Since the topic of carbon nanotubes is one of the most interesting topics in the nanotechnology research, the Atomistix software has this Nanotube Grower tool in the VNL which can be used to build carbon nanotubes with different chiralities (n, m), which allow the user to use these generated carbon nanotubes in other systems and calculations. As mentioned in the Introduction, a carbon nanotube can be specified by the two indices (n, m). Figure 3.9 shows the window of Nanotube Grower tool, and it shows in the left box the three values that can be changed in this tool: n, m, and distance between the carbon atoms. Changing these values produces different carbon nanotubes which will be viewed in the preview window to the right; in addition, the information in the Tube Properties box will be changed. Also the band structure of the nanotube updates when the indices are changed because n and m control the band structure of the carbon nanotube. See Figure 3.10 which shows the tight-binding band structure for a carbon nanotube with (n=4, m=1).
Figure 3.9 Nanotube Grower-window.

Figure 3.10 Tight-binding structure for a carbon nanotube constructed in the Nanotube Grower.
Script Editor

This tool provides more freedom for the user to make changes in the script that was produced by the NanoLanguage Scripter. Also it is used to add customization to scripts that cannot be done with the NanoLanguage Scripter such as “for loops”, changing labels, and obtaining the results in a different format. The script firstly should be produced by the NanoLanguage Scripter by following the procedure listed in the subsection of the NanoLanguage Scripter. Then, drag and drop the produced script in the Script Editor window which allows the user to make the changes. Finally, the new script can be run to extract the desired results.

Result Browser

The function of the Result Browser is to look through the contents of the generated VNL files from the ATK or from the tools in the VNL itself. A VNL file can have more than one object; for instance, it can have atomic configuration, nanolanguage script, methods, and calculated numerical data such as the electron density. The VNL file can be located in the file system and then dragged and dropped in the Result Browser to display its contents. The left panel in the Result Browser shows the sample name and the description of it: either script or VNL file. The right panel displays the contents of the selected entry in the left panel, and new samples can be added to the list in the left panel.
as the user drags and drops objects in the Browser. Also a selected object can be removed from the list by choosing “Clear Selected Entry” in the window (see Figure 3.11). If the selected object from the left panel is a script, the right window will display the corresponding text. And if the selected sample is a numerical data, such as the energy spectrum, then the right window presents this spectrum (a two-dimensional plot); if other numerical data from computations for locations on a 3D grid are chosen (e.g., electron density) in the left panel, then the right window shows a message saying that the Nanoscope must be used to see these results.

![Figure 3.11 The Result Browser window.](image)

Figure 3.11 The Result Browser window.
Job Manager

The Job Manager tool has the most important function among the VNL tools and is usually used in the final steps. It is used to run nanolanguage scripts using the ATK engine. Drag-and-drop is applied in this tool. Many scripts can be dropped into Job Manager; scripts will run consecutively one at a time. When running a script starts, a log window will appear (see Figure 3.12) which shows the results of the running job. After one run is complete, the results can be exported by pressing “Export” in the window.

![Figure 3.12 Final log window of a script run through Job Manager.](image)

Two-Probe System

One of the most important applications of the Atomistix software is to study the transport of electrons through a molecule or a bulk material. The atomic modeling can be done in the software by building a two-probe system (open system). In this system, a central region (the molecule or the bulk material) is coupled with two semi-infinite
electrodes. Virtual NanoLab or ATK can be used to apply a finite bias across the electrodes and the central region. Next, VNL can be used to study the transmission spectrum for electron transport through the central region.

Firstly, the Crystal Cupboard is used to produce the crystal that will be used as the material for both electrodes (see Figure 3.8). Then this bulk sample is dropped into the Atomic Manipulator (see Figure 3.13) to generate the two electrodes from this bulk crystal. Right click in the preview window of the Manipulator and select cleave from the context menu. The instrument changes the mode from bulk crystal to the two-probe manipulator (see Figure 3.13 below).

![Figure 3.13](image)

**Figure 3.13** *The electrodes of the two-probe system in the Atomic Manipulator.*

The above figure shows a sequence of layers outside the electrode cell extending inside to the center of the system. These layers are called Surface Layers (screening layers), and they are considered part of the central region. The function of these layers is to screen the central region from the influence of the bulk electron densities in the
electrodes. Between, e.g., a molecule and an electrode, the layers provide a transition region over which the electron density changes to that of the electrode material. The number of surface layers is controlled via the parameters called Surface Layers under the Two-Probe tab. Finally, the central region structure (molecule or bulk material) has to be dropped into the open Manipulator and properly positioned between the electrodes as in Figure 3.14. The molecule in the central region can be rotated by right clicking in the preview window and choosing the right entry. Also the electrodes can be translated by following the same procedure. The last step is to save the structure and drop it into the Scripter to run it and extract the electron transport parameters.

**Figure 3.14** The finished configuration of two-probe system.

In a similar way, a carbon nanotube between two metallic electrodes system can be created, and its transmission spectrum and density of states can be calculated along with its conductance.
3.3.2 Atomistix ToolKit (ATK)

As presented in the previous section, the ATK is the main engine in the VNL which is used to run the scripts and the files through using the Job Manager tool in the VNL. ATK calculates the electronic structure and the electron transport calculations in an advanced method, and it calculates the electron transport for a range of voltages. On the other hand, some limitations on setting-up systems and calculations in the Virtual NanoLab exist, such as desired iterations via the “for loop”, manually changing some parameters in the script, producing new structures, and the types of the output files. These limitations and others can be overcome by directly using the ATK without using the VNL. The input file for ATK is written in a Python scripting language, and a set of pre-defined scripted actions in Python called NanoLanguage which provides an extended functionality. The output files can be formatted in a VNL file or in numerical values that can be analyzed. In order to use the ATK directly, a Python script must be written for the structure and for the physical properties that the user wants to extract for the system. Next, the script can be run by using the ATK window.

Python Language

NanoLanguage is a combination of the advanced ATK modules and the Python programming language [74]; some of the structures of Python language are listed below:

A. **Indentation**: The indentation determines if the code belongs to the defined function or to the remaining code. Python stops executing and returns an error if the code is not correctly indented.
B. Comments: In order to make a comment in the script, the comment line must begin with “#” character, and for a multi-line comment triple quotes must be used.

C. Modules: A module is a directory containing a collection of functions, classes, and tools that are not ordinarily available when Python is used. These modules have to be imported into the script in order to use the contained functions, or the needed function can be imported individually from the module. For example, ATK, math, and NumPy modules can be imported.

D. List: An object is used to collect elements between the square brackets. Elements in a list are numbered from zero.

E. Tuples: A tuple is made in a similar fashion as a list. But square brackets are replaced with parentheses.

F. Dictionaries: Dictionaries are constructed using curly braces “{}” to store configuration values of parameter settings etc.

G. Basic commands: “if”, “for”, “while”, “print”.

H. Functions: A function is used to avoid the redundancy in the code.

I. NumPy: The NumPy is used to store values from analysis functions. It is different from the List because it can be used to perform more advanced and complicated mathematical operations.
ATK Modules

After importing the ATK class into the script, then all ATK modules are available. Some examples are listed below:

A. Configurations: Type of systems that can be constructed in the Atomistix software, and the commands used to build these configurations are:

1. MoleculeConfiguration: This command is used to build a molecule, and it belongs to the KohnSham module in the ATK class.

2. BulkConfiguration: It used to create a bulk material, and it belongs to KohnSham module too.

3. PeriodicAtomConfiguration: It is used to construct a semi-infinite system such as long carbon nanotube. This command belongs to both KohnSham and TwoProbe modules.

4. TwoProbeConfiguration: It is used to build two-probe systems, and it belongs to the TwoProbe module.

B. Lattices: Different kinds of lattices are already defined in Kohn-Sham and TwoProbe modules; these predefined functions can be used to build lattices with the desired molecular parameters. Some of these functions are:

1. SimpleCubic

2. BodyCenteredCubic

3. FaceCenteredCubic

4. Hexagonal

5. ...
C. **Calculations:** Calculation commands that are provided in KohnSham and TwoProbe modules are:

1. `calculateTotalEnergy`
2. `calculateElectronDensity`
3. `calculateAtomicForces`
4. `calculateCurrent`
5. `executeSelfConsistentCalculation`
6. ...

In a summary to this chapter, the main two methods, DFT and NEGF, used in the software that performs the calculations were reviewed. Next, the work of the Atomistix software package was presented. VNL can be used to set up nanosystems through a variety of tools to create the scripts for these systems in order to evaluate their physical and the transport properties. ATK can be used directly by writing a nanolanguage script for the desired system. This script uses ATK modules and the Python language. In this work VNL was used to extract the atomic positions for different systems and, also it was used to analyze the results. All the systems in this project were created by manually writing Python scripts for them; finally, ATK software was used to directly execute these scripts on the Beowulf cluster.
Chapter 4

Methods and Procedures

In this chapter the nanostructures that were addressed in the project will be presented; also the procedures to construct these systems will be clarified through presenting the scripts for these systems. Next, the method parameters will be defined through the script to compute the physical and transport parameters such as the total energy, atomic forces, and electrical current. Finally, the methods and the approaches for calculating the mechanical and the electrical properties for these systems will be discussed.

4.1 NanoSystems Addressed in the Project

Three different nanosystems were addressed in this project. The first one was constructed to be a test case in order to test the method, the script, and whether it is possible to extract elastic moduli using ATK.
After the first case succeeded and the results were reasonable, the second system was constructed to study the mechanical properties while the third system was designed to study the transport properties and the conductance. These three systems and their associated computational objectives were:

1) Carbon-Chain in between two copper electrodes: A test case to calculate the atomic forces and Young’s modulus.
2) Isolated Carbon Nanotube: To calculate the atomic forces and Young’s modulus.
3) Carbon nanotube in between two copper electrodes: To compute the transport properties and the current.

4.1.1 Carbon Chain

The twelve carbon atoms that form the carbon chain were situated between two copper (Cu) fcc (111) electrodes. These elements form the two-probe system which will be used to calculate the atomic forces on the carbon atoms in the central region. Specific attention will be paid concerning how to construct the two electrodes and the scattering region using nanolanguage script. This scattering region is the molecule and the surface layers of the electrodes. And the function of these surface layers as mentioned before in the VNL-section is to avoid the interaction between the semi-infinite electrodes and the molecule in the central region. The starting point is to build the Cu electrodes, the Cu (111) layers. For more information about the structure of the fcc lattice and the (111) surface, refer to [75].

Figure 4.1 shows the structure of the Cu-crystal generated by translations of the fcc lattice and the crystal is cut along the (111) plane.
If the layers, shown above, are labeled from top to bottom as A, B, and C, then the layer underneath C layer will be an A layer resulting in a …ABCABC…crystal. To maintain the symmetry in the system and according to this lattice structure then:

- The left electrode is ABC.
- The right electrode is ABC.
- The left screening layers are AB.
- The right screening layers are BC.
- The two-probe system is (ABC)AB-(12 carbon atoms)-BC(ABC).

Knowing the lattice constant (a) one can calculate the distance between the copper atoms (d) and the separation between the layers (d_sep). Now it is possible to define these constants and generate the scattering Cu-layers in ATK with the following procedure:
1. Load ATK.TwoProbe and NumPy modules and define the lattice constants.

```python
from ATK.TwoProbe import *
from numpy import *

# Define constants for the fcc lattice
# a: lattice constant, d: Cu-Cu bond length, d_sep: Cu layer separation
a = 3.6150
d = a / sqrt(2)
d_sep = a / sqrt(3)
```

2. Generate a 3x3 A-layer as in Figure 4.2 by defining some generator vectors \( \mathbf{v} \) and \( \mathbf{u} \) as in the figure.

![A 3x3 (111) layer](image)

**Figure 4.2** A 3x3 (111) layer [75].

```python
# Define vector generators for 3x3 layers
u = array([[1.0, 0.0, 0.0]]) * d
v = array([[0.5, sqrt(3)/2.0, 0.0]]) * d
du = array([[0.5, 1.0/2.0/sqrt(3), 0.0]]) * d

# Setup the A layer
n = 3
A = reshape(zeros(3*n*n), (n*n, 3)) * 1.0
for j in range(n):
    for i in range(n):
        A[i+n*j] = i*u + j*v
```

Here in script, \( \mathbf{d} \) will be used to generate B and C layers by translating the A layer with this vector as in the figure below.
3. Generate the B and the C layers.

```bash
# Setup the B and C layers
B1 = A + 1.0*du
B2 = A + 1.0*du
C  = A + 2.0*du
```

Two B layers were generated here because one will be placed to the left of the chain and the other will be placed to the right of the chain. Note that all the layers that have been generated have their z-coordinate set equal to zero so that later it will be easier to position them in the right position.

4. Define the carbon related parameters such as the Carbon-Copper bond length (dest_CuC), Carbon-Carbon bond length (dest_CC), and the length of the Carbon-Chain (Cchainlength) which is equal to the [number of carbon atoms -1] times the carbon-carbon bond length.

```bash
# Define C-constants
dest_CuC  = 1.6
dest_CC   = 1.27
Cchainlength=dest_CC*11
```

The C-C bond length was found by relaxing the system manually to reach the minimum energy.

5. Construct the central region by generating the left screening layers (A and B1), defining the positions of the carbon atoms, and generating the right screening
layers (B2 and C). One important issue raised here was the location of the C-chain in the (x, y) plane with respect to the layer B1. We found that the minimum energy can be obtained when the first atom in the C-chain is located exactly at the center of mass for any three atoms in the B layer. By looking at Figure 4.3, this means the (x, y) coordinates for the C-chain must be:

\[(x, y) = n \times du\]

\[du = d\left(\frac{\frac{1}{2}}{2\sqrt{3}}\right)\]

\[n = 2, 3, 5, 6\]

Here, du is the same translation vector that was used before to generate B and C layers.

In the script that defines the C-chain positions, n was designated to be 3.

```
# Setting-up AB-Cohain-BC central region
B1[::, 2] +- d_sep
C_chains_positions = [
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 1*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 2*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 3*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 4*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 5*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 6*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 7*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 8*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 9*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 10*dest_CC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_Cuc + 11*dest_CC),
]
B2[::, 2] +- d_sep + dest_Cuc + length_Cohain + dest_Cuc
C[::, 2] +- d_sep + dest_Cuc + length_Cohain + dest_Cuc + d_sep
```

Here in this script, the first line is to shift the B1 layer along the z-axis by the amount “d_sep”; the same is applied for the B2 and C layers to the right of the C-chain.
6. Define the scattering elements and scattering positions.

```python
# Central region elements
scattering_elements = 18*[Copper]
sorting_elements += 12*[Carbon]
sorting_elements += 18*[Copper]

# Central region positions
scattering_positions = concatenate((A,B1,Cchain_positions,B2,C)).tolist()
sorting_positions *= Angstrom
```

7. Define the atoms in the electrodes and the unit cell. A small unit cell that consists of three copper atoms will be defined, and then this unit cell will be repeated in the X and Y direction to produce the complete electrode layers.

```python
# Electrode elements and positions
electrode_elements = 3*[Copper]
electrode_positions = [
    [0.0, 0.0, 0.0],
    [0.5*d, d*0.5/sqrt(3), 1*d_sep],
    [1.0*d, d*1.0/sqrt(3), 2*d_sep]
] * Angstrom

# Electrode unit cell
electrode_unitcell = [
    [d, 0.0, 0.0],
    [0.5*d, sqrt(3)/2.0*d, 0.0],
    [0.0, 0.0, 3.0*d_sep]
] * Angstrom
```

8. Construct the electrodes by using the object “PeriodicAtomConfiguration” which used to represent the semi-infinite electrodes.

```python
# Construct the Electrode
electrode = PeriodicAtomConfiguration(
    electrode_unitcell,
    electrode_elements,
    electrode_positions
)
```
9. Combine the elements together by using “TwoProbeConfiguration” object.

```python
# Two-probe configuration
# Define a TwoProbeConfiguration with the electrodes and central part
mytwo_probe = TwoProbeConfiguration(
    electrode, electrode,
    scattering_elements,
    scattering_positions,
    electrode_repetitions=[[3, 3], [3, 3]],
    equivalent_atoms=[[0, 4], [2, 43]]
)
```

As shown in the script, the unit cell of the electrode is repeated three times in the X and Y directions which produces the electrodes with three layers in each and nine copper atoms in each layer. The variable “equivalent_atoms” connects the first atom in the left electrode with the fifth atom in the central region which is in the center of the most left screening layer, i.e., this gives these two atoms the same (x, y) coordinates. Also it connects the third atom in the right electrode with atom number 44 in center region which is in the center of the most right screening layer.

10. Produce a VNL file to visualize the two-probe setup using VNL software.

```python
# Export the two-probe system to a VNL file.
fileName = "Cu-12Cchain-Cu.vnl"
myVNL_file = VNDFile(fileName)
myVNL_file.addToSample(mytwo_probe, "Cu-12Cchain-Cu")
print "VNL File created: ", fileName, ", you can visualize it with VNL."
```

11. The final setup for a 12 C-atoms chain in between two copper electrodes will be viewed as in Figure 4.4.
The script was written using the Notepad++ and executed in ATK. This script and the produced configuration will be used later in this project to stretch the C-chain and calculate the total energy and the atomic forces. The complete script in the Notepad++ will be attached to this work in Appendix A.

4.1.2 Isolated Carbon Nanotube Segments

The calculation of Young’s modulus for a CNT using an atomistic approach is a goal of this work. Performing the computations within reasonable execution times requires the selection and use of finite CNT lengths. A segment of a metallic single-wall carbon nanotube (4, 4) consisting of 48 atoms is constructed as a molecule. This CNT
segment will be stretched later in this work, and the atomic forces and the total energy will be calculated so that eventually Young’s modulus will be calculated from the force-strain and energy-strain curves. Also the script for this structure will be a base to produce other segments of the same type of carbon nanotube to investigate possible length dependencies of the results. The differences between the selected segments are their lengths and, of course, the number of carbon atoms in each one.

- CNT (4, 4) with 48 atoms (3 unit cells).
- CNT (4, 4) with 64 atoms (4 unit cells).
- CNT (4, 4) with 80 atoms (5 unit cells).
- CNT (4, 4) with 96 atoms (6 unit cells).
- CNT (4, 4) with 112 atoms (7 unit cells).
- CNT (4, 4) with 128 atoms (8 unit cells).
- CNT (4, 4) with 144 atoms (9 unit cells).
- CNT (4, 4) with 160 atoms (10 unit cells).
- CNT (4, 4) with 176 atoms (11 unit cells).
- CNT (4, 4) with 192 atoms (12 unit cells).

The atomic information of CNT (4, 4) was extracted from the VNL. The unit cell of the CNT (4, 4) was created by using the “Nanotube Grower” tool in the VNL (see Figure 4.5). Next, the script for this unit cell was created by dropping the sample into the “NanoLanguage Scripter” tool. Then, the atomic positions were used in writing the script for the isolated CNT with 48 C-atoms, which consists of three of this unit cell.
Figure 4.5 The unit cell of CNT (4, 4) created by VNL.

Because the CNT (4, 4) segment was needed as a molecule system, ATK was directly used to produce this system instead of the VNL software that treats the produced CNTs by its tool as periodic systems. As will be seen the positions of this unit cell will be repeated two times or more in the z-direction in order to produce the desired lengths for the CNT (4, 4) segments.

In order to produce the first isolated CNT (4, 4) segment, the script was written by defining the elements (48 carbon atoms) and the positions (coordinates for these atoms). Next, the object “MoleculeConfiguration” was used to combine the positions with atoms as a molecule structure. Finally, a VNL file was created to visualize the structure (see Figure 4.6).
```python
from ATK.KohnSham import *
from numpy import *

# Define elements
elements = [Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon]  # List of elements

# Define coordinates
coordinates = array([[ 10.32234085, 7.6055209, 0. ],
                     [ 9.5268956,  9.5268956, 1.23148512],
                     [ 7.6055209, 10.32234085, 0. ],
                     [ 5.6861462,  9.5268956, 1.23148512],
                     [ 4.89070095, 7.6055209, 0. ],
                     [ 5.6861462,  5.6861462, 1.23148512],
                     [ 7.6055209,  4.89070095, 0. ],
                     [ 9.5268956,  5.6861462, 1.23148512],
                     [ 9.55848997,  8.96443087, 0. ],
                     [ 8.30942683, 10.22980153, 1.23148512],
                     [ 6.24661093,  5.95848997, 0. ],
                     [ 4.98324027,  8.30942683, 1.23148512],
                     [ 5.25455103,  6.24661093, 0. ],
                     [ 6.9036497,  4.98324027, 1.23148512],
                     [ 8.56443087,  5.25455103, 0. ],
                     [10.22980153,  6.9036497, 1.23148512],
                     [10.32234097, 7.60552113, 2.46297622],
                     [ 9.52689552,  9.52689552, 3.69446445],
                     [ 7.60552113, 10.32234097, 2.46297622],
                     [ 5.68614626,  9.52689552, 3.69446445],
                     [ 4.89070082,  7.60552113, 2.46297622],
                     [ 5.68614626,  5.68614626, 3.69446445],
                     [ 7.60552113,  4.89070082, 2.46297622],
                     [ 9.52689552,  5.68614626, 3.69446445]]
```

Figure 4.6 The final structure of the 48-atom segment of a metallic single-wall CNT (4, 4).
In order to write the scripts for the rest of the CNT (4, 4) segments, the only changes necessary in the above script are to add the number of the additional carbon atoms to the “element” function; and also, adding the coordinates of these additional elements to the “coordinates” function. See Appendix B for the complete scripts.

Figure 4.7 The final structures of the metallic single-wall CNT (4, 4) segments. The difference between each two horizontally adjacent segments is one unit cell.

4.1.3 Carbon Nanotube in Between Two Copper Electrodes

The single-wall carbon nanotube that was constructed in the previous section will be positioned in between two copper electrodes. The same electrodes that were created in the test case, a carbon chain between two electrodes, will be used here also with some changes in the dimensions of the surfaces layers. The dimensions of the copper surface layers were made 5x5 atoms instead of 3x3 to avoid the interaction between the carbon
nanotube and the electrodes. The 5x5 surface layers also insure that the carbon nanotube is totally in the central region and all of the carbon atoms are combined with the surface layers. The same procedure as in the test case will be followed with the following changes:

1. Surface layers setup: To make the A, B, and C layers 5x5 atoms then the following script was made for the layer A.

```python
# Setup the A layer
n = 5
A = reshape(zeros(5*3*n), (n*n, 3))*1.0
for j in range(n):
    for i in range(n):
        A[i+n*j] = i*u + j*v
```

2. Central region setup: The coordinates of the carbon nanotube substituted the coordinates of the chain. The important issue raised here was the location of the carbon nanotube in the (x, y) plane with respect to the surface layer B1. It is expected that the minimum energy situation of the system occurs when the CNT is exactly positioned at the center of the layer B, i.e., the center of the carbon nanotube must be positioned at the same coordinates of the central atom in the layer B1. The coordinates of the carbon nanotube were shifted to achieve this geometry. (The coordinates of this atom are (8.94667, 5.16536), and of the CNT are (7.60652, 7.60652). Thus, all the coordinates of the carbon nanotube must be shifted by (1.34015, -2.44116).)
# Define AB-Chain-BC central region

define CNTL

dest_CuC = 1.6
dest_QC = 1.42
CNTLength = 6.15744066

B1[1,2] = d_sep
CNT_positions = array([[10.3234097, 7.6065213, 0.],
                        [9.52689552, 9.52689552, 1.23148811],
                        [7.6065213, 10.3234097, 0.],
                        [5.68614626, 5.68614626, 1.23148811],
                        [4.98970022, 7.6065213, 0.],
                        [5.68614626, 5.68614626, 1.23148811],
                        [7.6065213, 4.98970022, 0.],
                        [9.52689552, 5.68614626, 1.23148811],
                        [9.52689552, 9.52689552, 1.23148811],
                        [8.96443091, 8.96443091, 0.],
                        [8.30942726, 10.22980118, 1.23148811],
                        [6.24810097, 9.58490387, 0.],
                        [4.98970022, 8.30942726, 1.23148811],
                        [5.25455199, 6.24810097, 0.],
                        [6.903615, 4.98970022, 1.23148811],
                        [8.96443091, 5.25455199, 0.],
                        [10.22980118, 6.903615, 1.23148811],
                        [10.3234097, 7.6065213, 2.46297622],
                        [9.52689552, 9.52689552, 3.69464445],
                        [7.6065213, 10.3234097, 2.46297622],
                        [5.68614626, 9.52689552, 3.69464445],
                        [4.89070082, 7.6065213, 2.46297622],
                        [5.68614626, 5.68614626, 3.69464445],
                        [7.6065213, 4.89070082, 2.46297622],
                        [9.52689552, 5.68614626, 3.69464445],
                        [9.52689552, 9.52689552, 3.69464445],
                        [8.96443091, 5.25455199, 2.46297622],
                        [10.22980118, 6.903615, 3.69464445],
                        [10.3234097, 7.6065213, 4.92595243],
                        [9.52689552, 9.52689552, 6.15744066],
                        [7.6065213, 10.3234097, 4.92595243],
                        [5.68614626, 9.52689552, 6.15744066],
                        [4.89070082, 7.6065213, 4.92595243],
                        [5.68614626, 5.68614626, 6.15744066],
                        [7.6065213, 4.89070082, 4.92595243],
                        [9.52689552, 5.68614626, 6.15744066],
                        [9.52689552, 9.52689552, 6.15744066],
                        [8.96443091, 5.25455199, 4.92595243],
                        [10.22980118, 6.903615, 6.15744066]])

B2[1,2] = d_sep + dest_CuC + CNTLength + dest_CuC
C1[1,2] = d_sep + dest_CuC + CNTLength + dest_CuC + d_sep
3. Central region atoms: Each surface layer has 25 atoms, and the CNT has 48 atoms.

```
# Central region elements
scattering_elements = 50*[Copper]
scattering_elements += 48*[Carbon]
scattering_elements += 50*[Copper]
```

4. The entries of the “equivalent_atoms” and the “electrodes_repetitions” functions will be changed to match the surface layers.

```
# Two-probe configuration
# Define a TwoProbeConfiguration with the electrodes and central part
mytwo_probe = TwoProbeConfiguration(
    (electrode, electrode),
    scattering_elements,
    scattering_positions,
    electrode_repetitions=[[5,5],[5,5]],
    equivalent_atoms = ([0,0],[2,124])
)
```

A VNL file can be created for this structure to view it in VNL (see Figure 4.7).

**Figure 4.8** The final structure of the metallic single-wall CNT (4, 4) in between two copper electrodes.
4.1.4 Calculation Parameters

For all the systems used in this work a set of calculation parameters must be defined in order to calculate the total energy, atomic forces, current, electron density, etc. Some of these parameters were discussed in detail in the ATK section. This set of parameters varies with the kind of constructed system, i.e., some of these parameters apply in the two-probe configuration and do not apply in the molecule configuration. If any small changes occur in the structure then these parameters vary according to these changes. The values of the parameters used in the scripts are given below and are followed by the script to execute the self consistent calculation so that it can be used in calculating the physical quantities for systems. If the user doesn’t specify the value of any parameter in the script, then the ATK takes the default value of that parameter (check the ATK manual [76] for the default values of the calculation parameters). To find the proper values of these parameters, they will be tested before applying them in the script that will be used to compute the physical and the electrical quantities for the systems.

```
# Parameters
exchange_correlation_type = LDA.PZ

# Reduce basis set size for copper
# The default basis set is DoubleZetaPolarized basis set

basis_set_params_Cu = basisSetParameters(
    type = SingleZeta,
    element = Copper
)
```
These parameters can be used for the two-probe system calculations by collecting them in 
TwoProbeMethod as following:

```python
# Collect parameters into a two-probe calculation method
method = TwoProbeMethod(
    electrode_params, electrode_params,
    basis_set_parameters =
        [basis_set_params_Cu, basis_set_params_C],
    iteration_control_parameters = iteration_control_params,
    electron_density_parameters = electron_density_parameters
)
```

Also, it can be used for the molecule system calculations through using the KohnSham 
method:

```python
# method = KohnShamMethod(
    exchange_correlation_type = exchange_correlation_type,
    electron_density_parameters = electron_density_parameters,
    basis_set_parameters = basis_set_parameters,
    iteration_control_parameters = iteration_control_parameters,
)```
After defining the method, the self consistent calculation can be performed through the following script:

```python
# Specify verbosity and checkpoint file
runtime_params = runtimeParameters(
    verbosity_level = 10,
    checkpoint_filename = 'Cu-CNT-Cu_stretched_0_scf.nc'
)

# Perform SCF calculation with chosen parameters
scf = executeSelfConsistentCalculation(
    atomic_configuration = mytwo_probe,
    method = method,
    runtime_parameters = runtime_params
)
```

The entry of the “atomic_configuration” argument depends on the constructed structure, and the same applies in the “method” argument. The “checkpoint” argument produces a file that can be used later to rerun the calculations and calculate the physical quantities without executing the self consistent calculation again. The produced checkpoint file can be restored by using the following function in the script:

```python
scf = restoreSelfConsistentCalculation("Cu-CNT-Cu_stretched_0_scf.nc")
```

### 4.2 Approaches and Calculations

Two main subsections are included under this section. The first one discusses the procedure to find the mechanical properties of the isolated carbon nanotube and to test the calculation parameters. Also it talks about the method of stretching the CNT (4, 4) segments for different values and calculating the atomic forces and Young’s modulus. Moreover, this subsection presents the proposed definitions in this project for Poisson’s ratio and the area of the carbon nanotube. The second subsection presents the procedure for finding the current flows through the carbon nanotube in between two copper
electrodes when a certain voltage difference is applied across them. This carbon nanotube will be stretched and the conductance will be calculated for different strains.

4.2.1 Mechanical Properties

In this section the methods of calculating Young’s modulus and Poisson’s ratio for a stretched carbon nanotube will be discussed. There will be two types of calculations. One will consider both the axial stretching and the radial reduction in the CNT which is represented by Poisson’s ratio. In the other situation the axial stretching will only be considered and the Poisson’s ratio will be neglected.

4.2.1.1 The Approach of Stretching The CNT And Defining Poisson’s Ratio

A speculative atomistic-level approach is tried for calculating Young’s modulus for the C-chain and the single-wall carbon nanotube; also the total energies are used to extract the Young’s modulus value. In this approach the carbon nanotube and the C-chain are stretched in the axial direction by $\Delta \lambda$. One end of the C-chain or the CNT is fixed and the atoms in the C-chain or the layers in the CNT are moved along the axial direction by the same amount. The $z$-coordinates of the C-atoms in the script are changed according to the following equation:

$$Z_i = Z_{i0} + i \Delta Z$$

where

$$i = 0, 1, 2, \ldots, (n - 1),$$

(4.1)
The variable \( i \) starts from zero because the z-coordinate of the first atomic layer has to be fixed, i.e., no incremental change is applied. In the above equation, \( n \) is the atomic or layer index, \( Z_i \) is the new coordinate, \( Z_0 \) is the original coordinate, and \( \Delta Z \) is the axial increment. According to this approach, the separation between two atoms or two layers is increased by the same amount, which is \( \Delta Z \). Thus, the change in the length is:

\[
\Delta \lambda = (n - 1) \times \Delta Z \tag{4.2}
\]

A first principles calculation is used to find the total force on the end atoms necessary for static equilibrium. In the C-chain case, there is just one end atom, and for the CNT there are 8-carbon atoms in the end layer. Finally, the forces versus the change in lengths or the strains were plotted to extract Young's modulus from the slope of the graph, and also it can be calculated from the total energy (see Equations 2.3 and 2.4).

The method that was applied to define and derive Poisson’s ratio is based on the assumption that the carbon nanotube has the same volume before the stretching and after it. This assumption is consistent with the results of stretching a macroscopic cylinder but is less certain at the scale of a nanotube since the computation of area and volume has less certainty. The parameters of the CNT are:

- \( r_0 \) = The original radius
- \( L_0 \) = The original length
- \( \Delta r \) = The reduction in the CNT radius
- \( \Delta L \) = The stretching amount (elongation) in the CNT
Since the volume of the CNT must be constant when changing the length, so:

The original volume = The volume after the stretching

\[ V_{\text{original}} = V_{\text{after}} \]
\[ \pi r_0^2 L_0 = \pi (r_0 - \Delta r)^2 (L_0 + \Delta L) \]
\[ r_0^2 = (r_0 - \Delta r)^2 \left(1 + \frac{\Delta L}{L_0}\right) \]

From the definition of the strain: \( \varepsilon = \frac{\Delta L}{L_0} \)

Then

\[ r_0^2 = (r_0^2 + \Delta r^2 - 2r_0\Delta r)(1 + \varepsilon) \]
\[ 1 = (1 + \varepsilon) + \left(\frac{\Delta r}{r_0}\right)^2 (1 + \varepsilon) - 2\left(\frac{\Delta r}{r_0}\right)(1 + \varepsilon) \]

Define the radius reduction as

\[ R = \frac{\Delta r}{r_0} \quad \text{(4.3)} \]

Then:

\[ R^2 - 2R + \left(1 - \frac{1}{(1 + \varepsilon)}\right) = 0 \]
\[ R = 1 \pm \sqrt{\frac{1}{1 + \varepsilon}} \]

Apply the condition that:

At zero elongation (\( \Delta L = 0 \)) then there is no strain (\( \varepsilon = 0 \)) meaning that there is no radial reduction (\( \Delta r = 0 \)), i.e., \( R = 0 \). Then,

\[ R = \frac{\Delta r}{r_0} = 1 - \frac{1}{\sqrt{(1 + \varepsilon)}} \quad \text{(4.4)} \]

This equation shows the relation between the radius reduction and the strain.
Poisson’s ratio ($\nu$) is the ratio of the transverse strain (R: normal to the applied load), divided by the normal strain ($\varepsilon$):

$$\nu = \frac{R}{\varepsilon} = \frac{1}{\varepsilon} \left[ 1 - \frac{1}{\sqrt{1 + \varepsilon}} \right]$$

(4.5)

This equation can be plotted over a range of strains (see the Figure 4.8 below):

![Figure 4.9 Poisson’s ratio as a function of the strain for a carbon nanotube.](image)

This definition of the Poisson’s ratio was applied on the coordinates of the atoms in the script. In the script of the CNT (4, 4) the atomic positions are given in Cartesian coordinates, (x, y, z), not in Polar coordinates (r, $\theta$). Therefore, the (x, y) coordinates have to be changed for each specific (r), i.e., for each specific strain ($\varepsilon$). From the geometry of the base of the strained CNT displayed in Figure 4.9 and by applying the triangles symmetry for a C-atom on the circumference:
\[
\frac{\Delta r}{r_0} = \frac{\Delta y}{y_0} = \frac{\Delta x}{x_0} = R.
\] (4.6)

Using Equation 4.6 for the atom \((i)\):

\[
\Delta x_i = x_{0i}R, \quad \Delta y_i = y_{0i}R,
\]

where \(i = 1, 2, \ldots, n\) and \(n\) is the number of atoms in the CNT.

Equation 4.7 shows the reduction in \((x, y)\) coordinates for each atom. These calculations and derivations that were made are based on a coordinate system that has its origin at the center of the CNT. However, for our CNT setup, the center of the CNT is at \((x_{center}, y_{center})\), e.g., in the case of the CNT \((4, 4)\) that is used in this project \((x_{center}, y_{center}) = (7.606521, 7.606521)\). Thus, \(\Delta x_i\) and \(\Delta y_i\) must be found by shifting the coordinates for all atoms by \((x_{center}, y_{center})\).

Equation 4.7 becomes:

\[
\begin{align*}
\Delta x_i &= (x_{0i} - x_{center})R \\
\Delta y_i &= (y_{0i} - y_{center})R
\end{align*}
\] (4.8)

These reductions in \((x, y)\) coordinates are applied to the CNT coordinates, which makes them become as follows:

\[
\begin{align*}
X_i &= x_{0i} - \Delta x_i \\
Y_i &= y_{0i} - \Delta y_i
\end{align*}
\] (4.9)

Now to stretch the CNT or the C-chain axially, then Equation 4.1 must be used to modify \(z\)-coordinates in the script. In the CNT \((4, 4)\) case, in order to apply the stretching and Poisson’s effect on the CNT both at the same time, Equation 4.1 is used for the axial
stretching and Equations 4.8 and 4.9 are used to apply Poisson’s effect on the (x, y) coordinates. In the next subsection the method of changing the coordinates in the script will be discussed according to these equations.

**Figure 4.10** Top view on the CNT before and after the stretching. The radius gets smaller after the axial stretching according to Poisson’s effect.
4.2.1.2 The Script of the Stretching Process

A. The C-chain: The script for the C-chain is a modified version of the one used for constructing the C-chain. The change in this script occurs when defining the central region as below.

```
# Define AB-C-chain-BC central region
Bl[::,2] += d_sep

dest_CuC = 1.6
dest_CC = 1.27
initial_Cchainlength=dest_CC*11

# Define the amount of stretching in each atom position
Z_increment = 0.03
# Define the change in length
length_change=11*Z_increment
# Define the new length
new_Cchainlength=initial_Cchainlength + length_change
# Define the strain E
E=length_change/initial_CNTlength
# Strecthing the initial positions according to Zi = Zi0 + iZ_increment

Chain_positions = [
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 1*dest_CC + 1*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 2*dest_CC + 2*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 3*dest_CC + 3*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 4*dest_CC + 4*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 5*dest_CC + 5*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 6*dest_CC + 6*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 7*dest_CC + 7*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 8*dest_CC + 8*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 9*dest_CC + 9*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 10*dest_CC + 10*Z_increment),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 11*dest_CC + 11*Z_increment),
]
Bl[::,2] += d_sep + dest_CuC + new_Cchainlength + dest_CuC
Cl[::,2] += d_sep + dest_CuC + new_Cchainlength + dest_CuC + d_sep
```

In this script, “z_increment” is the same as ΔZ in Equation 4.1; whenever this value is changed new values of strain are generated. A for-loop over this value will be done and the atomic forces will be calculated for each value. See Appendix A for the
complete script with a for-loop, and notice that the first atom in the chain was fixed by not adding any increment to its z-coordinate.

**B. CNT without Poisson’s ratio:** The script of constructing the isolated CNT will be used here with only changes in the z-coordinate. Notice in the script that the first layer, which consists of 8 C-atoms, is fixed, i.e., no increment on the z-coordinate of that layer. A for-loop will be done over the “z_increment” value to calculate multiple atomic forces for multiple strains.

```python
# define the increment in the Z-coordinate
z_increment = 0.002
initial_CNT_length = 6.15744066
length_change = 5 * z_increment
# 5 times from the number of layers in the CNT are stretched
new_CNT_length = initial_CNT_length + length_change
# define the strain
strain = length_change / initial_CNT_length

CNT_positions = array([ [10.32234097, 7.60652113, 0.], [9.52695552, 9.52695552, 1.23148811 + 3 * z_increment], [7.6052113, 10.32234097, 0.], [5.6814626, 9.52695552, 1.23148811 + 3 * z_increment], [4.89070908, 7.60652113, 0.], [5.6814626, 5.6814626, 1.23148811 + 3 * z_increment], [7.6052113, 4.89070908, 0.], [9.52695552, 5.6814626, 1.23148811 + 3 * z_increment], [9.55649037, 8.56443081, 0.], [6.2461097, 9.55649037, 0.], [4.98324013, 6.2461097, 1.23148811 + 3 * z_increment], [5.2945199, 6.2461097, 0.], [6.293615, 4.98324013, 1.23148811 + 3 * z_increment], [8.9643081, 5.25455189, 0.], [10.22980118, 6.903610, 1.23148811 + 3 * z_increment], [10.32234097, 7.60652113, 2.46297622 + 2 * z_increment], [9.52695552, 9.52695552, 3.69446445 + 3 * z_increment], [7.6052113, 10.32234097, 2.46297622 + 2 * z_increment], [5.6814626, 9.52695552, 3.69446445 + 3 * z_increment]], dtype=float)
```
C. CNT with Poisson’s ratio: In writing this script first one must start with calculating the reductions in (x, y) coordinates for the atoms. Next, these reductions are applied to the original coordinates. Finally, one should modify the z-coordinates for the C-atoms, i.e., stretch the structure.

• The (x, y) coordinates for all the 48 C-atoms are defined through the script.

```bash
# Define the script

# Define the stretching amount
S_increment = 0.000

# Calculate the new length
new_CNTLength = initial_CNTLength + length_change

# Define the coordinates
CNT_coordinates = array([[10.29284095, 7.6665209, 0],
                         [9.5268956, 9.5268956, 0],
                         [7.6665209, 10.29284095, 0],
                         [5.6861642, 9.5268956, 0],
                         [4.89707029, 7.6665209, 0],
                         [5.6861642, 5.6861642, 0],
                         [7.6665209, 4.89707029, 0],
                         [9.5268956, 5.6861642, 0],
                         [9.9584899, 8.96443087, 0],
                         [8.30942726, 10.22980118, 0],
                         [6.24861097, 9.9584899, 4.92959242],
                         [4.98324011, 8.30942726, 4.92959242],
                         [5.8455109, 6.24861097, 4.92959242],
                         [6.903615, 4.98324011, 4.92959242],
                         [9.9644001, 5.8455109, 4.92959242],
                         [10.29284095, 6.903615, 4.92959242]])
```
• Calculate the reductions and apply them to the coordinates to extract the modified (x, y) coordinates.

```plaintext
# Shifting the center of the CNT
coordinates = CNT_coordinates - [ x0, y0, 0.0 ]
print coordinates
# Reduction in the X,Y coordinates = R(X-X0) and R(Y-Y0)
Change_in_coordinates = R*coordinates

# the new X,Y coordinates after the strain and the radial reduction
New_CNT_coordinates = CNT_coordinates - Change_in_coordinates
```

• Add the new (x, y) coordinates to the z-coordinate for all the C-atoms and apply the stretching on the z-coordinates.

```plaintext
# Adding the Z-components
CNT_Full_coordinates = New_CNT_coordinates + [( 0, 0, 0.0 ),
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
[ 0, 0, 1.23148812 + 1*Z_increment ],
[ 0, 0, 0.0 ],
]

CNT_Full_coordinates = CNT_Full_coordinates * Angstrom
```
Note that the same procedure is applied for all the other CNT (4, 4) segments.

Again the main difference between them is the number of atoms which leads to more coordinates.

**D. CNT between copper electrodes without Poisson’s effect:** This case is similar to the C-chain one. The change will occur in the central region coordinates.

```plaintext
# Define AB-Chain: BC central region
D([:2] += d_sepc)
dest_CuC = 1.6
dest_{Cu} = 1.42
z_increment = 0.002
initial_CNTLength=6.15744006
length_change= 5*z_increment
# 5 cones from the number of layers in the CNT are stretched

# Define the strate
E= length_change/initial_CNTLength

new_CNTLength=initial_CNTLength + length_change

CNT_positions = array([ 10.3234097, 7.6062113, 0.0, + d_sepc + dest_{CuC}],
[ 9.5269552, 9.5268952, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 7.6062113, 10.3234097, 0.0, + d_sepc + dest_{CuC}],
[ 5.68814264, 9.5268952, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 4.88070093, 7.6062113, 0.0, + d_sepc + dest_{CuC}],
[ 5.68814264, 5.6864626, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 7.6062113, 4.88070093, 0.0, + d_sepc + dest_{CuC}],
[ 9.5269552, 5.6864626, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 9.5269552, 9.52695037, 0.0, + d_sepc + dest_{CuC}],
[ 8.39042762, 10.22990011, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 6.24961097, 9.52695037, 0.0, + d_sepc + dest_{CuC}],
[ 4.90524013, 8.39042762, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 5.25451085, 6.24961097, 0.0, + d_sepc + dest_{CuC}],
[ 6.503615, 4.90524013, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 8.96443081, 5.25451085, 0.0, + d_sepc + dest_{CuC}],
[ 10.22990011, 6.503615, 1.23148811 + d_sepc + dest_{CuC} + 1*z_increment],
[ 10.3234097, 7.6062113, 2.46297622 + d_sepc + dest_{CuC} + 2*z_increment],
[ 9.5269552, 9.5268952, 3.69446445 + d_sepc + dest_{CuC} + 3*z_increment],
[ 7.6062113, 10.3234097, 2.46297622 + d_sepc + dest_{CuC} + 2*z_increment],
[ 5.68814264, 9.5268952, 3.69446445 + d_sepc + dest_{CuC} + 2*z_increment],
[ 4.88070093, 7.6062113, 2.46297622 + d_sepc + dest_{CuC} + 2*z_increment],
[ 5.68814264, 5.6864626, 3.69446445 + d_sepc + dest_{CuC} + 2*z_increment],
[ 7.6062113, 4.88070093, 2.46297622 + d_sepc + dest_{CuC} + 2*z_increment],
[ 9.5269552, 5.6864626, 3.69446445 + d_sepc + dest_{CuC} + 2*z_increment],
```

This system will be used mainly to calculate the electrical current. One value of the strain will be chosen and multiple values of current will be produced for different values of voltages. Next, another strain will be set and the same process will be repeated.
4.2.1.3 Calculation Parameters Convergence Test

This test of parameters was not computed for the C-chain because it was a test case; however, this parameter test was done for the other two systems. The main point in the test is to try different values for a parameter and to calculate the total energy and see if it converges. All the parameters were fixed except the one parameter that was being tested. The main focus is on the following calculation parameters for the isolated CNT:

1) Basis Set
2) Exchange Correlation Type
3) Tolerance
4) Mesh Cut-off

A. Calculation Parameters Discussion

The parameters were discussed in detail earlier; here the process for choosing the proper value of a parameter and how to apply it in the script will be discussed.

1) Basis Set: In the systems used in this work there are two types of atoms, carbon and copper. The Double-zeta polarized basis set is chosen for the Carbon because the carbon atoms have double bonds among them, which makes a Double-zeta polarized sufficient to represent the electron orbitals in this kind of bonding. On the other hand, a Single-zeta basis was used to represent the orbitals in the metallic bond between the copper atoms.
2) Exchange Correlation Type: LDA-PZ was chosen to be the parameter for all the systems because it is the standard exchange correlation functional for calculating Young’s modulus. Also it is the default value in the ATK.

3) Tolerance: The iteration in calculating the total energy is considered to be converged if the change in the total energy between two successive iterations falls below the tolerance. By lowering the tolerance the accuracy increases, but at the same time the computational time increases. Two values were tried in the convergence test, $1 \times 10^{-5}$ and $1 \times 10^{-4}$. As it will be presented in the results, the difference in energies between these two values is in the third decimal point and is considered as a negligible difference in comparison with the total energy itself. This implies that the performed calculations are in a good level of accuracy. For the isolated CNT, a tolerance of $1 \times 10^{-5}$ is set, and for the CNT between the copper electrodes a tolerance of $1 \times 10^{-4}$ is set because this system takes longer time to execute.
4) Mesh Cut-off: The mesh cut-off will be changed in the range between 30 to 190 Rydberg, while the previous parameters are fixed and ATK will be asked to find the associated total energies. The proper value of the mesh cut-off parameter is the point at which the energy saturates while still maintaining a reasonable execution time for the computations (the greater the mesh cut-off value the longer the execution time).

```python
# Tolerance for convergence
iteration_control_params = iterationControlParameters(
    tolerance = 1e-4,
    max_steps = 200
)

# Calculate total energy against the mesh cut-off
# Set verbosity level so that all energy components are printed
import ATK

# for mesh_cutoff in range(30, 200, 10) * Rydberg:
for mesh_cutoff in range(30, 200, 10) * Rydberg:
    electron_density_parameters = electronDensityParameters(mesh_cutoff = mesh_cutoff)
    method = KohnShamMethod(
        exchange_correlation_type = exchange_correlation_type,
        electron_density_parameters = electron_density_parameters,
        basis_set_parameters = basis_set_parameters,
        iteration_control_parameters = iteration_control_parameters
    )
    print('
    MeshCutoff (Rydberg)  TotalEnergy (eV)')
    print('----------------------------------------
    ', mesh_cutoff.inUnitsOf(Rydberg), ' ',
    calculateTotalEnergy(method.apply(CNT)).inUnitsOf(eV)
```
used for the other CNT (4, 4) segments. For the CNT between the copper electrodes, the
test was done on a similar system and the results were applied to the original system. This
similar system consists of the CNT and the two copper surfaces layers to the left and two
to the right; this system was built as a molecule. See the Appendix D for the complete
convergence test scripts.

B. Results and Analysis of the Convergence Test

Based on the results that will be presented in this section the proper values of the
calculation parameters will be chosen when calculating the atomic forces, total energy,
and the current. The following parameters were chosen and fixed for the isolated CNT (4, 4) and for the CNT between two copper electrodes:

- **Basis Set**: Single-zeta for copper, Double-zeta polarized for carbon.
- **Exchange Correlation Type**: LDA-PZ.
- **Tolerance** is $1 \times 10^{-5}$ for the isolated CNT (4, 4) segments, and it is $1 \times 10^{-4}$ for the
  CNT in between two copper electrodes.

The last parameter, the mesh cut-off, was varied and the total energy was calculated. The
results were as follows:
1) Isolated CNT with 48 C-atoms:

<table>
<thead>
<tr>
<th>Mesh Cut-off (Rydberg)</th>
<th>Energy (eV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30.0</td>
<td>-7523.200500</td>
</tr>
<tr>
<td>40.0</td>
<td>-7522.379757</td>
</tr>
<tr>
<td>50.0</td>
<td>-7522.371580</td>
</tr>
<tr>
<td>60.0</td>
<td>-7522.501308</td>
</tr>
<tr>
<td>70.0</td>
<td>-7522.449559</td>
</tr>
<tr>
<td>80.0</td>
<td>-7522.474599</td>
</tr>
<tr>
<td>90.0</td>
<td>-7522.468398</td>
</tr>
<tr>
<td>100.0</td>
<td>-7522.469679</td>
</tr>
<tr>
<td>110.0</td>
<td>-7522.471779</td>
</tr>
<tr>
<td>120.0</td>
<td>-7522.468418</td>
</tr>
<tr>
<td>130.0</td>
<td>-7522.469599</td>
</tr>
<tr>
<td>140.0</td>
<td>-7522.469486</td>
</tr>
<tr>
<td>150.0</td>
<td>-7522.469017</td>
</tr>
<tr>
<td>160.0</td>
<td>-7522.469333</td>
</tr>
<tr>
<td>170.0</td>
<td>-7522.469149</td>
</tr>
<tr>
<td>180.0</td>
<td>-7522.468807</td>
</tr>
<tr>
<td>190.0</td>
<td>-7522.470052</td>
</tr>
</tbody>
</table>

Table 4.1 Total energies for a 48-atom CNT segment for different mesh cutoffs.

Figure 4.11 Computed total energy for a 48-atom CNT segment plotted for a sequence of mesh cut-off values.
Table 4.1 shows the values of the mesh cut-off parameter and the associated energy for each value. As can be seen, the energy is well stabilized at the point 130 Rydberg because the difference in the total energy after this point falls in the third decimal point, which is small. Figure 4.11, which plots these points in Table 4.1, shows that the total energy tends to be constant with a slight change after the mesh cut-off equal to 100 Rydberg. The 130 Rydberg value was used as insurance for accuracy in strain computations.

2) The rest of the isolated CNT (4, 4) segments:

Figure 4.12 presents the plotted mesh cut-off values against the total energies for CNT segments with 96, 144, and 192 atoms.

Figure 4.12 Computed total energies for 96, 144, and 192-atom CNT segments plotted for a sequence of mesh cut-off values.
As shown in the above figure, the appropriate value of mesh cut-off for all the carbon nanotubes is about 100 Rydberg. However, values greater than 100 Rydberg were chosen for running the calculations. The choice of this value was taken to be 130 Rydberg for CNT with 96 C-atoms, and 160 Rydberg for CNT with 144 C-atoms and 192 C-atoms. Higher values of mesh cut-off were set to insure more accuracy in the force calculations. The carbon nanotube segments between the 48-atom CNT and the 96-atom CNT had the value of 130 Rydberg and the ones between the 96-atom CNT and the 192-atom CNT had the value of 160 Rydberg.

3) The CNT-copper surface layers system:

Table 4.2 and Figure 4.13 provide the mesh cut-offs and total energies for the CNT-copper surface layer “molecular” system used for these tests.

<table>
<thead>
<tr>
<th>Mesh Cut-Off (Rydberg)</th>
<th>Energy (eV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>-141550.8423</td>
</tr>
<tr>
<td>110</td>
<td>-141551.1360</td>
</tr>
<tr>
<td>120</td>
<td>-141551.0169</td>
</tr>
<tr>
<td>130</td>
<td>-141550.9749</td>
</tr>
<tr>
<td>140</td>
<td>-141550.9847</td>
</tr>
<tr>
<td>150</td>
<td>-141550.9803</td>
</tr>
<tr>
<td>160</td>
<td>-141550.9804</td>
</tr>
<tr>
<td>170</td>
<td>-141550.9828</td>
</tr>
<tr>
<td>180</td>
<td>-141550.9772</td>
</tr>
<tr>
<td>190</td>
<td>-141550.9809</td>
</tr>
<tr>
<td>200</td>
<td>-141550.9802</td>
</tr>
</tbody>
</table>

*Table 4.2* Mesh cut-off parameter values and the total energies for CNT between the copper surface layers.

The Table 4.2 and Figure 4.13 show that the energy tends to be stable after the value of mesh cut-off equal to 150 Rydberg. This value was chosen to represent the mesh cut-off parameter in the calculations for the CNT in the two-probe copper electrodes system.
After the proper calculation parameters were selected, the next step was to write the script for calculating the physical and electron transport parameters.

### 4.2.1.4 The Methods of Calculating Young’s Modulus

Young’s modulus for the C-chain and the single-wall carbon nanotube segments are found through a first principles calculation of forces as a function of strain. The total energies are also used to extract the Young’s modulus. In general, the main steps in this method for finding Young’s modulus are:

- Extract the forces on each atom in the C-chain or in the carbon nanotube when there is no stretching (zero strain).
• Find the total force in z-direction on the end atoms in the structures. C-chain has one end atom, and the carbon nanotube has eight end atoms.
• Fix one end of the structure and stretch it by different lengths along the axial direction.
• Extract the atomic forces on each atom for each amount of stretching.
• Add up the z-components of the forces on the end atoms (z-axis is the axial direction).
• Subtract the total force on the end atoms at zero strain from the total force in the stretching cases. This is because the stretching process is not starting from a relaxed system. The system at zero strain is taken to be a reference point for the calculations.
• Plot the forces versus the change in lengths or versus the strains.
• Calculate the slope of the graph.
• The slope of the graph is related to Young’s modulus through the classical relations between the stress and the strain (see Equation 2.3).
• The energy can be extracted for each stretching case and it can be plotted against the strains. Then, Young’s modulus can be found by applying Equation 2.4.

So far in this chapter the scripts of constructing the three systems were clarified and a VNL view was presented for each system. Next, the calculation parameters were tested in order to use the proper parameters in executing the self consistent calculations. Now, the last step in writing the script is to ask ATK to calculate the atomic forces and the total energies for every time that the C-chain or the CNT segments are stretched. This subsection presents the script for calculating the atomic forces for the C-chain and how to
analyze the output results. Also it discusses the script for calculating the atomic forces and total energies for the isolated carbon nanotube segments. Young’s modulus is then calculated using the approach that was discussed above.

A. Carbon Chain

The script that describes the stretching of the C-chain will be added to the script that defines the calculation parameters. Next, a script for calculating the atomic forces and total energies will be written and added to the whole script (see Appendix A). The script for calculating the physical properties is as follows:

```python
# Calculate physical properties

# Perform SCF calculation with chosen parameters
scf = executeSelfConsistentCalculation(atomic_configuration = mytwo_probe, method = method, runtime_parameters = runtime_params)

# Calculate the system energy
total_energy = calculateTotalEnergy(self_consistent_calculation = scf)

# Calculate the system force
atomic_forces = calculateAtomicForces(self_consistent_calculation = scf)

if processIsMaster(): nPrint(total_energy,'Total energy')
if processIsMaster(): nPrint(atomic_forces,'Total Force')
if processIsMaster(): nPrint(z_increment,'z_increment')
if processIsMaster(): nPrint(length_change,'length_change')
```

The argument “z_increment” in the script of stretching the C-chain, which determines the amount of the strain applied on the chain, is increased after each time this script is compiled in order to calculate the atomic forces and total energies for different strains. This value of “z_increment” ranges from 0 to 0.05 Å with a step of 0.005 Å, which
leads to a maximum change in length equal to 0.55Å (see the script of stretching the C-chain). The output file for the above script will contain the “z_increment” value, length change, total energy, and atomic forces on each atom. For instance, the output file for “z_increment” equal to 0.02Å is shown below.

```
Z_increment
0.02
length_change
0.22
Total energy
-56335.1171351 eV
Total Force
# ===============================================================================
# Atomic Forces
# ===============================================================================
# Index     Fx (eV/Ang)     Fy (eV/Ang)     Fz (eV/Ang)
0 2.5233e-04 -2.8545e-04 3.1880e-01
1 2.4748e-01 2.9588e-01 2.5051e-01
2 -2.4767e-01 2.9105e-01 2.5271e-01
3 3.8054e-01 6.7038e-02 2.4952e-01
4 -5.6986e-03 -2.0452e-03 -1.5432e+00
5 -3.7484e-01 6.4791e-02 2.6122e-01
6 1.2879e-01 -3.6092e-01 2.5046e-01
7 -1.3217e-01 -3.5712e-01 2.6000e-01
8 3.3249e-03 1.5914e-03 3.5321e-01
9 2.4267e-02 1.4014e-02 8.5155e-01
10 -8.8853e-05 6.2157e-01 1.7072e+00
11 -2.4178e-02 1.3935e-02 8.5116e-01
12 5.3840e-01 -3.1087e-01 1.7071e+00
13 -5.3775e-01 -3.1040e-01 1.7076e+00
14 -1.2701e-04 2.8731e-02 7.8120e-01
15 -2.3929e-05 -2.7935e-02 8.5126e-01
16 2.4838e-02 -1.4459e-02 7.8120e-01
17 -2.4894e-02 -1.4345e-02 7.8192e-01
18 -6.6949e-04 -2.9500e-04 -2.9297e+00
19 4.3080e-05 1.5217e-05 -1.4711e+00
20 9.3481e-05 3.7138e-05 7.7479e-01
21 -3.9258e-05 -1.5125e-05 -4.4232e-02
22 4.4667e-06 1.5461e-06 -5.9004e-01
23 1.6470e-06 6.7438e-07 6.3391e-01
24 5.8393e-07 3.3449e-07 -6.3252e-01
25 7.5167e-06 3.1630e-06 5.7657e-01
26 -4.5703e-05 -2.0741e-05 -7.0500e-02
27 9.5190e-05 4.5692e-05 -7.6841e-01
28 1.4278e-05 -2.1744e-06 1.5327e+00
29 -3.4512e-04 -1.7275e-04 2.9629e+00
30 4.8658e-02 2.8104e-02 -8.3749e-01
31 -6.7420e-06 5.0434e-01 -1.7271e+00
```
32 -4.8556e-02  2.7962e-02  -8.3752e-01
33  4.3680e-01  -2.5218e-01  -1.7267e+00
34 -4.3660e-01  -2.5205e-01  -1.7272e+00
35  1.6970e-04  3.4606e-02  -7.9800e-01
36 -7.2031e-05  -5.6067e-02  -8.3759e-01
37  2.9902e-02  -1.7450e-02  -7.9800e-01
38 -2.9834e-02  -1.7215e-02  -7.9899e-01
39  7.4677e-02  4.3512e-02  -9.7952e-02
40 -7.7925e-02  3.9678e-02  -1.0726e-01
41  3.1151e-03  2.5826e-02  -3.2168e-01
42 -4.9466e-03  -8.6925e-02  -1.0584e-01
43 -2.1728e-01  -1.2523e-01  -5.1172e-01
44  2.2274e-01  -1.2670e-01  -5.0423e-01
45  2.3612e-02  -1.0611e-02  -3.2143e-01
46  1.4544e-03  2.5571e-01  -5.0477e-01
47 -2.4975e-02  -1.4996e-02  -3.1966e-01

The forces in the above output file are divided into three groups and they are:

1) Forces on the copper atoms of the left surface layers, and they are the first 18 forces because there are two surface layers with 9 Copper atoms in each.

2) Forces on the copper atoms of the right surface layers, and they are the last 18 forces.

3) Forces on the carbon atoms which form the C-chain; these forces are the 12 bolded forces in the middle because the chain consists of 12 C-atoms.

As seen in the output file that the three force components, (x, y, z), on each atom in the system are provided. As presented before in the method of calculating Young’s modulus that the important forces are those which are on the end atom and in the axial direction, z-axis. For example, the force which has the index 29 in the above output file is the force on the right end atom. Also the force with index 18 is the force on the left end atom. For each value of “z_increment” the force at zero strain is subtracted from the force on the end atom, and the process is repeated for all the “z_increment” values. Next, these forces are plotted against the length changes. Based on the definition of the Young’s modulus and according to Equation 2.3:
\[ Y = \frac{F/A}{\Delta \lambda / \lambda_0} \]

(4.10)

\[ F = (Y \frac{A}{\lambda_0}) \Delta \lambda \]

Based on the above equation the slope of the force-length change graph is:

\[ \text{Slope} = Y \frac{A}{\lambda_0} \]

(4.11)

\[ Y = \text{Slope} \frac{\lambda_0}{A} \]

Here, \( A \) is the cross sectional area. In the C-chain this area is taken to be the cross sectional area of the carbon atom with a radius equal to 0.77Å, which represents the covalent radius of the carbon atom [77].

**B. Isolated carbon nanotube segments:**

The script that describes the stretching of each carbon nanotube segment will be added to the script that defines the proper calculation parameters for that segment. For the isolated CNT (4, 4) systems two scripts were written. One takes into account the Poisson’s ratio and the other neglects it. Next, a script for calculating the atomic forces and energies will be written and added to the whole script (Appendix B). The script for calculating the physical properties taking into account the Poisson’s ratio is shown below. Setting “R” equal to zero accounts for the case where the radial changes in coordinates are ignored.
for \( z \) in numpy.arange(0.000, 0.013, 0.001):

```python
length_change = 5*z_increment
E = length_change / initial_CNTlength
R = 1 - sqrt(1 / (1 + E))

v = R/E
Change_in_coordinates = R*coordinates
New_CNT_coordinates = CNF_coordinates - Change_in_coordinates
CNF_Full_coordinates = New_CNT_coordinates + [[0, 0, 0],

[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0],
[0, 0, 1.23148812 + z_increment],
[0, 0, 0]]
```
CNT_Full_coordinates = CNT_Full_coordinates*Å

CNT = MoleculeConfiguration(elements, CNT_Full_coordinates)

scf = executeSelfConsistentCalculation(
    atomic_configuration=CNT,
    method=kohnsham_method
)

if processIsMaster(): nlPrint(E_increment, 'E_increment')

if processIsMaster(): nlPrint(length_change, 'Length change')

if processIsMaster(): nlPrint(E, 'strain')

if processIsMaster(): nlPrint(v, 'Poissons ratio')

total_energy = calculateTotalEnergy(self_consistent_calculation = scf)

if processIsMaster(): nlPrint(total_energy, 'Total energy')

atomic_forces = calculateAtomicForces(self_consistent_calculation = scf)

if processIsMaster(): nlPrint(atomic_forces)
As shown in the above script, a for-loop was done over the argument “z_increment”. This gives one output file that contains all the amounts of stretching and the related forces and total energies for each amount of stretching. Strain values range from zero strain, “z_increment” equal to zero, to a maximum strain equal to 1% at “z_increment” equal to 0.013Å. The output file for the above script will contain the “z_increment” value, length change, strain, Poisson’s ratio, total energy, and the atomic forces on each atom. A plot will be made for the forces versus the strains. From Equation 2.3:

\[ Y = \frac{F}{A} \frac{1}{\varepsilon} \]

\[ F = (YA)\varepsilon \]

The slope of the force-strain graph based on the above equation is:

\[ Slope = YA \]

\[ Y = \frac{Slope}{A} \]

An important question is asked regarding this equation: WHAT IS THE AREA, “A”, FOR THE CNT? This major issue arises when our macroscopic description of mechanical properties is applied to the nanoscale. In studies on single-wall CNTs, the areal uncertainty has been referred to as “Yakobson’s paradox” [78]. Young’s modulus in this research will be calculated and compared using three possible areas.
Figure 4.14 shows the view of one end layer of the carbon nanotube, which contains eight carbon atoms. The variables in the figure are:

- \( r \) is the covalent radius of the carbon atom, \( r = 0.77 \text{Å} \).

- \( R \) is the CNT radius. The mean value of \( R \) is taken in the case in which the Poisson’s ratio is considered. The original radius \( R \) for the unstrained CNT is taken when the Poisson’s ratio is neglected.

Three types of the carbon nanotube area are now defined.

1. In CNT (4,4) there are eight end atoms, so the first area is taken as the sum of the cross-sectional areas for each end atom:

   \[
   A = 8 \times \pi r^2 \quad (4.14)
   \]

   This choice is consistent with the fact that the calculated forces are summed over the forces on the eight end atoms.
2. The end surface-area of the CNT-ring:

\[ A_{\text{Ring}} = \pi(R + r)^2 - \pi(R - r)^2 \]
\[ A_{\text{Ring}} = \pi[(R^2 + 2rR + r^2) - (R^2 - 2rR + r^2)] \]
\[ A_{\text{Ring}} = 4\pi r R \]  \hspace{1cm} (4.15)

3. Cross-sectional area of the CNT (classical view):

\[ A = \pi R^2 \]  \hspace{1cm} (4.16)

As it will be presented in the results, applying these different definitions of the area makes a significant difference in the Young’s modulus value. To avoid using the area of the carbon nanotube in the calculations, a different approach can be followed. This approach has been presented before by E. Hernandez et al. (1998) [17]. A similar approach is to define:

\[ Y_s = Y_A \]  \hspace{1cm} (4.17)

\( Y_s \) is the area independent Young’s modulus. In the case in this research, Equation 4.13 becomes:

\[ Y_s = \text{Slope} = Y_A \]  \hspace{1cm} (4.18)

\( Y_s \) will be calculated for all the carbon nanotube segments. It will be calculated from the force-strain plot and from the total energy-strain plot.
4.2.2 Two-Probe Method for Calculating the Conductance

In this section the method of calculating the current for a stretched carbon nanotube between two copper electrodes is presented. The script for constructing this system and stretching the carbon nanotube in the central region was presented in the previous section. Also the proper calculation parameters of this system were found when the parameter convergence test was discussed in the previous section. The only script to add here is the script of calculating the current for different voltages applied through the electrodes. In the script shown below, a strain value is first defined. Next, a for-loop is made over the voltage difference. In this for-loop, the current value is calculated for each voltage value. The current-voltage points are then plotted to extract the conductance value from the slope. For each strain a current-voltage plot will be generated and a conductance value calculated. Finally, the conductance values are plotted versus the strains to study the behavior of the conductance of the carbon nanotube when it is stretched.

In a summary to this chapter, the systems used in this project and the scripts for constructing them were presented. Next, the method of stretching these systems axially was clarified. A new approach of defining the Poisson’s ratio was derived in this chapter too. Also, the results of the convergence test of the parameters were presented. Finally, the method of calculating Young’s modulus from the forces was discussed. The next chapter discusses the results and the analysis; it will present the values and the numbers that were extracted from the output files for the systems used in the project. The values of Young’s modulus will be found and discussed. Some values of the electrical conductance will be calculated from the current-voltage plots.
# Calculate physical properties

# Perform SCF calculation with chosen parameters

csf = executeSelfConsistentCalculation(
    atomic_configuration = mytwo_probe,
    method = method,
    runtime_parameters = runtime_params
)

if processIsMaster(): nlpnlPrint(E, 'strain')

# P-V Points

from ATK.TwoProbe Import *
import numpy

# Run bias from 0.0 and 0.6 in steps of 0.1

for voltage in numpy.arange(0, 0.6, 0.1):

dft_method = TwoProbeMethod(
    electrode_parameters=(electrode_params, electrode_params),
    basis_set_parameters = [basis_set_params_Cu, basis_set_params_C],
    iteration_control_parameters = iteration_control_params,
    electrode_voltages = (voltage, -voltage)*Volt
)

scf = executeSelfConsistentCalculation(
    atomic_configuration=mytwo_probe,
    method = dft_method,
    initial_calculation = scf
)


current = calculateCurrent(scf)

if processIsMaster(): nlpnlPrint(voltage, 'Bias Voltage')
if processIsMaster(): nlpnlPrint(current.inUnitsOf(Ampere), 'Current')
Chapter 5

Results and Analyses

This chapter is divided into two main sections; the first one presents the results and the analyses that are related to the mechanical parameters and calculations. In this section the atomic forces and energies for the C-chain and the carbon nanotube segments will be presented next to the plots of these parameters against the strains. Young’s modulus will be calculated from these graphs. Electron density graphs will be shown for some carbon nanotube segments to provide more understanding of their structure. The second section shows the current and voltage values that were calculated for the carbon nanotube in between two copper electrodes. Also this section presents the current-voltage plots and the values of the electrical conductance that were extracted from them.
5.1 Mechanical Properties Results

The carbon atomic chain served as an initial test case for the method of calculating Young’s modulus at the atomistic level and provided experience with the software. Verification of an approximately linear stress-strain relationship for small strains at the atomistic scale was an objective. While the chain is structurally different from a nanotube, the carbon bonding was expected to yield an elastic modulus with a similar order of magnitude. The results shown below justified the application of the method to nanotube segments. Calculations of the elastic moduli of these structures are discussed.

5.1.1 C-Chain

The C-chain used in this project consists of 12 carbon atoms along a line and placed between two copper electrodes, as presented before. The initial length of this chain at zero strain, before it is stretched, is equal to the number of the C-C bonds existing times the length of the C-C bond in the chain:

\[
\lambda_0 = (n - 1) \times dest_{CC}, \\
\lambda_0 = 11 \times 1.27\text{Å} = 13.97\text{Å}
\]  

Here, \( n \) is the number of carbon atoms. Regarding the stretching process, the distance between every two carbon atoms was increased from 0.005Å to 0.055Å by a step of 0.005Å; this is represented in the script by the variable “\( z_{\text{increment}} \)”. The change in
the C-chain length is equal to the elongation amount in each bond times the number of carbon bonds:

\[ \Delta \lambda = 11 \times z \text{ _increment} \]  

(5.2)

After running the scripts and producing the output files, the z-components of the forces, on the right end atom, were analyzed according to the method that was described in the previous chapter. The total forces and energies for each elongation are listed in Table 5.1 below.

<table>
<thead>
<tr>
<th>Z\text{ _increment} (Å)</th>
<th>Change in length(Å)</th>
<th>Force (eV/Å)</th>
<th>(\Delta F) (eV/Å)</th>
<th>Total Energy (eV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>0.00</td>
<td>4.39</td>
<td>0.0000</td>
<td>-56340.94</td>
</tr>
<tr>
<td>0.005</td>
<td>0.06</td>
<td>4.02</td>
<td>-0.37</td>
<td>-56332.10</td>
</tr>
<tr>
<td>0.010</td>
<td>0.11</td>
<td>3.66</td>
<td>-0.73</td>
<td>-56338.07</td>
</tr>
<tr>
<td>0.015</td>
<td>0.17</td>
<td>3.50</td>
<td>-0.89</td>
<td>-56329.19</td>
</tr>
<tr>
<td>0.020</td>
<td>0.22</td>
<td>2.96</td>
<td>-1.43</td>
<td>-56335.12</td>
</tr>
<tr>
<td>0.025</td>
<td>0.28</td>
<td>2.63</td>
<td>-1.76</td>
<td>-56340.98</td>
</tr>
<tr>
<td>0.030</td>
<td>0.33</td>
<td>2.30</td>
<td>-2.09</td>
<td>-56332.09</td>
</tr>
<tr>
<td>0.035</td>
<td>0.39</td>
<td>1.99</td>
<td>-2.40</td>
<td>-56337.91</td>
</tr>
<tr>
<td>0.040</td>
<td>0.44</td>
<td>1.60</td>
<td>-2.79</td>
<td>-56328.99</td>
</tr>
<tr>
<td>0.045</td>
<td>0.50</td>
<td>1.39</td>
<td>-3.00</td>
<td>-56334.78</td>
</tr>
<tr>
<td>0.050</td>
<td>0.55</td>
<td>1.10</td>
<td>-3.29</td>
<td>-56340.50</td>
</tr>
<tr>
<td>0.055</td>
<td>0.61</td>
<td>0.82</td>
<td>-3.57</td>
<td>-56331.59</td>
</tr>
</tbody>
</table>

Table 5.1 The associated forces and energies resulted from stretching the C-chain by different amounts.
As seen in Table 5.1 the forces on the right atom in the C-chain have a positive sign which means that they are directed in the positive z-axis, which is toward the copper surface layers to the right. This is because there is more than one copper atom to the right that interacts with the end atom while there is just one carbon atom to the left that interacts with this end atom. A check was made on the other end of the C-chain and the results showed very slight differences in the forces and subsequently in the value of the calculated Young’s modulus. This result was expected because the system has a high symmetry, i.e., the right side is a mirror image to the left side. The energies are expected to increase in a parabolic behavior while increasing the stretching value; however, it is clear from Table 5.1 that the values of the total energy are fluctuating up and down. This fluctuation is presumably due to the default parameters that were used to extract these values and no convergence tests were done on this system.

![Graph showing the change in force on the right end atom versus the change in the length of the carbon chain.](image)

**Figure 5.1** Change in force on the right end atom versus the change in the length of the carbon chain.
The force differences were plotted versus the change in length in Figure 5.1. Although the calculations were relatively crude, linearity is observed. The negative slope is due to the direction of the forces and the scatter in the data is due to the use of default computational parameters. From the fitting model:

\[
\text{Slope} = 5.99 \text{eV}/\text{Å}^2 = 5.99 \times 10^{-9} \text{N}/\text{Å}
\]

\[
\text{Slope} = 95.84 \text{N}/\text{m}
\]

From Equation 4.11 the Young’s modulus can be calculated as following:

\[
\text{Slope} = Y \frac{A}{\lambda_0}
\]

\[
Y = \text{Slope} \frac{\lambda_0}{A}
\]

\[
\lambda_0 = 13.97 \text{Å}
\]

\[
A = \pi r^2 = \pi (0.77 \text{Å})^2
\]

\[
A = 1.86 \times 10^{-20} \text{m}^2
\]

then

\[
Y = 7.20 \text{TPa}
\]

This result of Young’s modulus has the same order of magnitude as experimental values for a CNT (TPa) (see Chapter 2). This test case, the C-chain, proved that it is possible to use ATK and the proposed method of stretching the Carbon chain to calculate the Young’s modulus for nanostructures. Also, the first principles calculation showed that the macroscopic point of view can be applied on the atomistic level.
5.1.2 Carbon Nanotube Segments

Forces and energies were calculated for the carbon nanotube segments listed below in Table 5.2. Different strains were applied to each segment to extract the strain forces and energies. As mentioned before, these segments were stretched axially only in one case. In the other case, the Poisson’s effect was applied while stretching these segments axially. Young’s modulus will be calculated for both situations and the results will be presented in the next subsections.

<table>
<thead>
<tr>
<th>Index</th>
<th>Number of Unit Cells</th>
<th>Number of Carbon Atoms</th>
<th>Length (Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>48</td>
<td>6.2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>64</td>
<td>8.6</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>80</td>
<td>11.1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>96</td>
<td>13.5</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>112</td>
<td>16.0</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>128</td>
<td>18.5</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>144</td>
<td>20.9</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>160</td>
<td>23.4</td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>176</td>
<td>25.9</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>192</td>
<td>28.3</td>
</tr>
</tbody>
</table>

Table 5.2 The metallic carbon nanotube (4, 4) segments used in the project.
5.1.2.1 The Results with Poisson’s Effect

Young’s modulus was calculated for all the carbon nanotube segments. The calculations were performed through using the forces and the energies. All the segments were stretched to a maximum strain of 1%.

1) CNT with 48 atoms.

The initial length of this carbon nanotube at zero strain, before it is stretched, is equal to the \([\text{number of the layers} -1]\) times the separation distance between each two of them:

\[
\lambda_0 = (m -1) \times L, \\
\lambda_0 = 5 \times 1.2315\text{Å} = 6.1575\text{Å} \quad (5.3)
\]

Above, \(m\) is the number of the layers in the carbon nanotube (see Table 5.2 for all the lengths of the carbon nanotube segments). The distance between every two layers was increased from 0.001Å to 0.013Å by a step of 0.001Å; this is represented in the script by the variable “\(z\_\text{increment}\)”. According to this process the strain ranges from 0.008 to 0.01. The strain values were calculated by using the classical definition of strain (see Equation 2.2). The change in the length of the carbon nanotube is equal to the elongation amount in each bond times the \([\text{number of the layers} -1]\):

\[
\Delta \lambda = 5 \times z\_\text{increment} \quad (5.4)
\]
After running the scripts and producing the output files, the z-components of the forces on the eight end atoms were analyzed according to the method that was described in the previous chapter. The total forces and energies for each elongation are listed in Table 5.3 below.

<table>
<thead>
<tr>
<th>Strain Index</th>
<th>Strain (ε%)</th>
<th>ΔF On The Left-surface (eV/Å)</th>
<th>ΔF On The Right-surface (eV/Å)</th>
<th>Total Energy (eV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E0</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>-7522.469</td>
</tr>
<tr>
<td>E1</td>
<td>0.08</td>
<td>0.20</td>
<td>-0.20</td>
<td>-7522.472</td>
</tr>
<tr>
<td>E2</td>
<td>0.16</td>
<td>0.42</td>
<td>-0.42</td>
<td>-7522.472</td>
</tr>
<tr>
<td>E3</td>
<td>0.24</td>
<td>0.64</td>
<td>-0.64</td>
<td>-7522.470</td>
</tr>
<tr>
<td>E4</td>
<td>0.32</td>
<td>0.86</td>
<td>-0.86</td>
<td>-7522.468</td>
</tr>
<tr>
<td>E5</td>
<td>0.41</td>
<td>1.08</td>
<td>-1.08</td>
<td>-7522.463</td>
</tr>
<tr>
<td>E6</td>
<td>0.49</td>
<td>1.29</td>
<td>-1.29</td>
<td>-7522.457</td>
</tr>
<tr>
<td>E7</td>
<td>0.57</td>
<td>1.51</td>
<td>-1.50</td>
<td>-7522.449</td>
</tr>
<tr>
<td>E8</td>
<td>0.65</td>
<td>1.73</td>
<td>-1.73</td>
<td>-7522.440</td>
</tr>
<tr>
<td>E9</td>
<td>0.73</td>
<td>1.94</td>
<td>-1.94</td>
<td>-7522.429</td>
</tr>
<tr>
<td>E10</td>
<td>0.81</td>
<td>2.15</td>
<td>-2.15</td>
<td>-7522.417</td>
</tr>
<tr>
<td>E11</td>
<td>0.89</td>
<td>2.36</td>
<td>-2.37</td>
<td>-7522.403</td>
</tr>
<tr>
<td>E12</td>
<td>0.97</td>
<td>2.57</td>
<td>-2.57</td>
<td>-7522.387</td>
</tr>
<tr>
<td>E13</td>
<td>1.00</td>
<td>2.78</td>
<td>-2.78</td>
<td>-7522.371</td>
</tr>
</tbody>
</table>

*Table 5.3 Forces and total energies of the stretched CNT (48-atom).*
Table 5.3 shows the forces on both ends of the carbon nanotube for each associated amount of strain. The forces on the left end are positive, which means that they are pointing to the right towards the carbon nanotube. As was expected, the forces on the right end are pointing to the left towards the carbon nanotube. The directions of these forces are due to the carbon atoms which try to hold their structure together when stretching. The magnitude of the forces on the right end and the ones on the left end are equal as expected. This result is the same for the rest of the segment; thus, one of these ends will be chosen to perform the calculations. The force differences on the left end were plotted versus the strain as in Figure 5.2. Figure 5.3 shows the forces versus the strains for the right end.

\[
\begin{array}{cccccccc}
\text{Strain} & 0.000 & 0.002 & 0.004 & 0.006 & 0.008 & 0.010 & 0.012 \\
\text{Force (eV/Å)} & 0.0 & 0.5 & 1.0 & 1.5 & 2.0 & 2.5 & 3.0 \\
\end{array}
\]

48 C-Atoms

Fitting Model:
\[ y = ax + b \]
\[ a = 264.70593715 \]
\[ b = 0.00076441703 \]

**Figure 5.2** The forces versus the strain of the left end of the CNT.
Figure 5.3 The forces versus the strain of the right end of the CNT.

The above two graphs have almost equivalent slopes, which eventually leads to the same value of Young’s modulus. From now on, the calculations will be performed using the left end of the carbon nanotube in all the segments since both ends reveal the same results. From the graph 5.2:

\[ \text{Slope} = 264.71 \text{eV/Å} = 4.24 \times 10^{-7} \text{N} \]

From Equation 4.12 and 4.13,

\[ Y = \frac{\text{Slope}}{A} = \frac{4.24 \times 10^{-7} \text{N}}{A} \]  
(5.5)

This equation will produce three different values of Young’s modulus because there are three different definitions for the area \( A \) (see Chapter 4):

1. \( A = 8 \times \pi r^2 \)
2. \( A = 8 \times \pi (0.77\text{Å})^2 \)
3. \( A = 14.88 \times 10^{-20} \text{m}^2 \)  
(5.6)
II. \[ A_{\text{Ring}} = 4\pi rR \]

The mean value of \( R \) is used in this equation, \( R = 2.707 \text{ Å} \).

\[
A = 4\pi \times 0.77 \times 2.707 \times 10^{-20} \text{ m}^2
\]

\[
A = 26.19 \times 10^{-20} \text{ m}^2
\]

(5.7)

III. \( A = \pi R^2 \)

\[
A = \pi (2.707 \text{ Å})^2
\]

\[
A = 23.02 \times 10^{-20} \text{ m}^2
\]

(5.8)

Table 5.4 below shows the three values of the Young’s modulus based on the three areas that were calculated in the above equation.

<table>
<thead>
<tr>
<th>Area Type</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area ( (10^{-20} \text{ m}^2) )</td>
<td>14.88</td>
<td>26.19</td>
<td>23.02</td>
</tr>
<tr>
<td>Young’s Modulus (TPa)</td>
<td>2.85</td>
<td>1.62</td>
<td>1.84</td>
</tr>
</tbody>
</table>

Table 5.4 The values of Young’s modulus for the CNT with 48-atoms based on three definitions of the area.

According to these results, the Young’s modulus values for the SWCNT segment found by applying a first principles calculation range from 1.6 to 2.8 TPa, roughly in agreement with the values in the literature (see Chapter 2). Obviously, Young’s modulus varies significantly when using different areas. This problem is serious for the groups working in this field; thus, no agreement has been made on one value of Young’s modulus for the SWCNT. One solution for this problem discussed previously was to state and use only \( Y_S \) (Equation 4.17 and 4.18). For the carbon nanotube with 48 -atoms:

\[ Y_s = YA = Slope = 264.7 \text{ eV/Å} \]

(5.9)
Another suggested solution to avoid using the area of the carbon nanotube is to extract Young’s modulus from the strain energies by using Equation 2.4 which states:

\[
Y = \frac{1}{V} \frac{\partial^2 E}{\partial \varepsilon^2} \\
Y = \frac{1}{A \times \lambda_0} \frac{\partial^2 E}{\partial \varepsilon^2} \\
YA = Y_s = \frac{1}{\lambda_0} \frac{\partial^2 E}{\partial \varepsilon^2} \\
\]

(5.10)

The second derivative of the total energy will be evaluated numerically and will be substituted in the above equation to calculate \(Y_s\). The first derivative was calculated manually and plotted against the strain. The slope of this plot presents the second derivative which will be used to calculate \(Y_s\). Figure 5.4 shows the behavior of the total energy while stretching the carbon nanotube.

**Figure 5.4** Total energy versus strain for the CNT with 48 atoms.
In the following graph, the first derivative of the energy with respect to the strain is plotted against the strain.

**Figure 5.5** The first derivative of the total energy with respect to the strain for the CNT with 48-atom.

The slope from the above graph was applied to the Equation 5.10 above:

\[
\text{Slope} = \frac{\partial^2 E}{\partial \varepsilon^2} = 2391.9\text{eV}.
\]

\[
Y_s = \frac{1}{\lambda_0} \frac{\partial^2 E}{\partial \varepsilon^2} = \frac{1}{6.2} \times 2391.9\text{eV/Å}
\]

\[
Y_s = 385.8\text{eV/Å}.
\]
2) **The Other Segments.**

The CNT segment with 48 atoms is considered an example of how the calculations were performed. After this step, only the final results will be presented for the other systems because the same procedure was used to calculate Young’s modulus. The final graphs and results will be given for the rest of the CNT segments. The force-strain graphs and the calculated Young’s modulus values for these segments will be followed by the total energy graphs and calculations.

The following pages present the force-strain graphs for the CNT segments.
Figure 5.6 The force-strain plots of all the CNT segments when Poisson’s effect is applied. The slope of these plots represents $Y_s$. 
Table 5.5 below lists the values of $Y_s$ for each CNT segment. These values were extracted from the above graphs.

<table>
<thead>
<tr>
<th>Index</th>
<th>CNT Segment by the Number of Carbon Atoms</th>
<th>Length (Å)</th>
<th>$Y_s$ (eV/Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td>6.2</td>
<td>264.7</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>8.6</td>
<td>123.5</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>11.1</td>
<td>256.9</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
<td>13.5</td>
<td>160.9</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
<td>16.0</td>
<td>265.9</td>
</tr>
<tr>
<td>6</td>
<td>128</td>
<td>18.5</td>
<td>177.8</td>
</tr>
<tr>
<td>7</td>
<td>144</td>
<td>20.9</td>
<td>132.5</td>
</tr>
<tr>
<td>8</td>
<td>160</td>
<td>23.4</td>
<td>187.5</td>
</tr>
<tr>
<td>9</td>
<td>176</td>
<td>25.9</td>
<td>183.4</td>
</tr>
<tr>
<td>10</td>
<td>192</td>
<td>28.3</td>
<td>264.0</td>
</tr>
</tbody>
</table>

**Table 5.5** The values of $Y_s$ for all the CNT segments used in the project.

The values of $Y_s$ are observed to change with the number of carbon atoms (or length of the segment). Figure 5.7 graphically demonstrates the variation of the modulus with segment length.
Dividing $Y_S$ by the area, the maximum, minimum, and average values of Young’s modulus for the segments are calculated for the three definitions of the area and presented in Table 5.6.

<table>
<thead>
<tr>
<th>Area Type</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area ($10^{-20} m^2$)</td>
<td>14.88</td>
<td>26.19</td>
<td>23.02</td>
</tr>
<tr>
<td>Maximum value of Young’s Modulus (TPa)</td>
<td>2.86</td>
<td>1.62</td>
<td>1.85</td>
</tr>
<tr>
<td>Minimum value of Young’s Modulus (TPa)</td>
<td>1.33</td>
<td>0.75</td>
<td>0.86</td>
</tr>
<tr>
<td>Average Value of Young’s Modulus (TPa)</td>
<td>2.17</td>
<td>1.23</td>
<td>1.40</td>
</tr>
</tbody>
</table>

Table 5.6 Young’s modulus for SWCNT (4, 4) based on different definitions of the area. The values were calculated by using the forces.

Figure 5.7 Modified Young’s modulus as a function of the carbon atoms in the CNT (4, 4).
The range of values of Young’s modulus calculated for short nanotube segments in this work is comparable to the values quoted in the literature for experimental and computational work on long nanotubes (see Chapter 3). According to their results, Young’s modulus for a single-wall carbon nanotube might have a value as small as 0.32 TPa [15] and as large as 5.5 TPa [19]. The variation in Young’s modulus is presumably the result of using different methods and approaches. The length dependence found for the Young’s modulus of CNT segments was unexpected. As shown in Figure 5.7, the modulus varies in a somewhat oscillatory manner with the number of atoms in the segment. To determine the cause, tests were initially made on the effects of the computational parameters in an attempt to rule out numerical or computational errors. For example, the sizes of the supercells used in the program were considered to insure that interactions between the ‘molecular’ segments remained negligible as segment lengths were increased. No numerical or computational errors related to parameters were found. In attempting to isolate the cause, the effects of the radial reduction when an axial strain is imposed were eliminated by repeating the computations with a Poisson ratio of zero (see the next section). On further investigation, a potentially important finding was that the initial total force at zero strain varies in a similar manner when the nanotube segment length is increased (Figure 5.8). The segment length is increased by adding a single unit cell containing sixteen carbon atoms. Beginning with the shortest segment, the number of unit cells progresses from three (48 atoms) to twelve (192 atoms), i.e., the number of unit cells oscillates from an odd to an even number. While violations of the pattern occur, the oscillation in the initial force data and in the calculated elastic modulus data with the odd-even number of unit cells present is reminiscent of the effects of parity in atomic chains.
with odd-even numbers of atoms. Future work will explore the wave functions and electron densities as the segment length changes from odd to even numbers of unit cells. Some of this work has been started in regards to calculating the electron density for the segments to look for the changes between the electron densities at the ends of the segments.

![Image](image_url)

**Figure 5.8** Total forces on the left end of the CNT as a function of the number of atoms.

Another important observation to point out is that almost all the groups working in this field receive a range of values for Young’s modulus. Some predicted this range to vary with the radius and others found it to vary with chirality and so on. For example, E. Hernandez et al. [17] predicted Young’s modulus to change with both the chirality and the radius with two different behaviors. Figure 2.1 in Chapter 2 shows the results this group received; in this figure, the behavior of Young’s modulus for carbon nanotubes has
a fluctuation in some points in the curve. This fluctuation is why the average value is usually presented as the reasonable result value for Young’s modulus.

As revealed in the procedure before, Young’s modulus can be also calculated from the relation between the total energy and the strain. The following pages provide the final graphs of the total energy against the strain followed by the calculations of the first derivative of the energy with respect to the strain. Finally, the final results and numbers of Young’s modulus are included Table 5.7.
Figure 5.9 The total energy-strain plots of all the CNT segments.
A smooth parabolic behavior of the total energy can be seen from the plots as expected. This behavior leads to a constant value of the second derivative of the energy against the strain. This behavior also confirms that the calculations were well converged.

The first derivative of the total energy was calculated numerically for each segment by dividing the difference in total energy by the difference in strain for every two points. Next, these derivatives were plotted against the average strain for each two points. Young’s modulus is related to the slope of these graphs by the Equation 5.10. Table 5.7 gives the values of the second derivative of the total energies of the segments, the original length of each one, and the calculated $Y_s$ according to Equation 5.10.

<table>
<thead>
<tr>
<th>Index</th>
<th>Number of C-Atoms in the CNT Segment</th>
<th>Length, $\lambda_0$ (Å)</th>
<th>$\frac{\partial^2 E}{\partial e^2}$ (eV)</th>
<th>$Y_s$ (eV/Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td>6.2</td>
<td>2391.9</td>
<td>385.8</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>8.6</td>
<td>2241.1</td>
<td>260.6</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>11.1</td>
<td>4406.3</td>
<td>396.9</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
<td>13.5</td>
<td>4496.0</td>
<td>333.0</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
<td>16.0</td>
<td>6830.5</td>
<td>426.9</td>
</tr>
<tr>
<td>6</td>
<td>128</td>
<td>18.5</td>
<td>6862.5</td>
<td>370.9</td>
</tr>
<tr>
<td>7</td>
<td>144</td>
<td>20.9</td>
<td>6124.2</td>
<td>293.0</td>
</tr>
<tr>
<td>8</td>
<td>160</td>
<td>23.4</td>
<td>7607.0</td>
<td>325.1</td>
</tr>
<tr>
<td>9</td>
<td>176</td>
<td>25.9</td>
<td>10292.4</td>
<td>397.4</td>
</tr>
<tr>
<td>10</td>
<td>192</td>
<td>28.3</td>
<td>12434.6</td>
<td>439.4</td>
</tr>
</tbody>
</table>

*Table 5.7* The calculated $Y_s$ values from the strain energy.
The behavior of $Y$, extracted from the energy was expected to be the same as the behavior extracted from the forces because the software calculates the physical properties based on the energy of the system.

In Table 5.8, the maximum, minimum and the average values of Young’s modulus for the three nanotube areas are given.

<table>
<thead>
<tr>
<th>Area Type</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area ($10^{-20} m^2$)</td>
<td>14.88</td>
<td>26.19</td>
<td>23.02</td>
</tr>
<tr>
<td>Maximum value of Young’s Modulus (TPa)</td>
<td>4.72</td>
<td>2.68</td>
<td>3.05</td>
</tr>
<tr>
<td>Minimum value of Young’s Modulus (TPa)</td>
<td>2.80</td>
<td>1.59</td>
<td>1.81</td>
</tr>
<tr>
<td>Average Value of Young’s Modulus (TPa)</td>
<td>3.90</td>
<td>2.21</td>
<td>2.52</td>
</tr>
</tbody>
</table>

Table 5.8 Young’s modulus for SWCNT (4, 4) based on different definitions of the area. The values were calculated by using the total energy.

By comparing these values with the ones that were calculated by using the forces, these values that were extracted from the total energies are larger by approximately 0.8 to 1.8 TPa. The main reason for the disagreement in values is believed to reside in the sensitivity of the results to the numerical calculations of derivatives, both in the software for calculation of forces and in the manual calculations performed on the energy data. The values of Young’s modulus derived from the energy method are comparable to the values discussed in the literature review in Chapter two. In the following section, Young’s modulus is calculated by the same procedures that have been done in the above calculations, but this time Poisson’s effect will be ignored.
5.1.2.2 The Results without Poisson’s Effect

The same steps that were done in the previous subsection are repeated here:

- The total forces on the left end of each carbon nanotube segment are calculated.
- These forces are plotted versus the strains.
- The slopes of these graphs represent the values of \( Y_s \).
- The maximum, the minimum, and the average values of \( Y_s \) are used to calculate the related Young’s modulus values.
- The three areas that are used to calculate Young’s modulus in this subsection are the same that were used before. One exception is that the radius of the carbon nanotube that will be used here is the original value (\( R=2.7158\AA \)) instead of the mean value because there is no change in the radius (no Poisson’s effect).
- The graphs of the total energy against the strain will be shown.
- Young’s modulus is calculated from the second derivative of the total energy.
- A table includes all of the calculated parameters.

The force-strain graphs for all the CNT segments were as following:
Figure 5.10 The force-strain plots of all the CNT segments were stretched without Poisson’s effect. The slope of these plots represents $Y_s$.

By comparing these plots with the ones extracted when the Poisson’s ratio was applied, the data appear less scattered and the linear fit is better (see, e.g., the data for the 144- and 160-atom segments). This behavior is due to the fact that the carbon bonds are changed in one dimension when stretching the CNT only axially. On the other hand, the bonds change in three dimensions when applying the Poisson’s effect with stretching. Table 5.9 below lists the values of $Y_s$ for the CNT segments that were stretched without applying the Poisson’s effect. These values were extracted from the above graphs.
<table>
<thead>
<tr>
<th>Index</th>
<th>Number of C-Atoms in CNT</th>
<th>Length (Å)</th>
<th>Ys (eV/Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td>6.2</td>
<td>309.3</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>8.6</td>
<td>203.6</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>11.1</td>
<td>309.6</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
<td>13.5</td>
<td>232.3</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
<td>16.0</td>
<td>305.2</td>
</tr>
<tr>
<td>6</td>
<td>128</td>
<td>18.5</td>
<td>245.3</td>
</tr>
<tr>
<td>7</td>
<td>144</td>
<td>20.9</td>
<td>218.9</td>
</tr>
<tr>
<td>8</td>
<td>160</td>
<td>23.4</td>
<td>272.5</td>
</tr>
<tr>
<td>9</td>
<td>176</td>
<td>25.9</td>
<td>251.5</td>
</tr>
<tr>
<td>10</td>
<td>192</td>
<td>28.3</td>
<td>307.6</td>
</tr>
</tbody>
</table>

Table 5.9 The values of $Y_s$ for all the CNT segments used in the project. Poisson’s effect was not applied in these calculations.

Figure 5.11 Young’s modulus as a function of the Carbon atoms in the CNT (4, 4) when Poisson’s ratio is not applied.
The values of $Y_s$ were plotted versus the number of carbon atoms in order to have a better understanding of the behavior of Young’s modulus while increasing the length of the carbon nanotube. Figure 5.11 shows this behavior.

The maximum, minimum, and the average values of Young’s modulus calculated from the $Y_s$ data are provided in Table 5.10 below. The fixed CNT radius (R=2.7158Å) was used. The values obtained when a radial reduction is included are also listed for comparison.

<table>
<thead>
<tr>
<th>Area</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Without Poisson’s Ratio</td>
<td>With Poisson’s Ratio</td>
<td>Without Poisson’s Ratio</td>
</tr>
<tr>
<td>Maximum value of Young’s Modulus (TPa)</td>
<td>3.33</td>
<td>2.86</td>
<td>1.88</td>
</tr>
<tr>
<td>Minimum value of Young’s Modulus (TPa)</td>
<td>2.19</td>
<td>1.33</td>
<td>1.24</td>
</tr>
<tr>
<td>Average Value of Young’s Modulus (TPa)</td>
<td>2.86</td>
<td>2.17</td>
<td>1.62</td>
</tr>
</tbody>
</table>

**Table 5.10** Young’s modulus from force calculations for SWCNT (4, 4) based on three areas and for two situations, with and without applying Poisson’s ratio.

The values that were calculated without Poisson’s effect are greater, but still fall in the reasonable range for Young’s modulus. Because of the method used in this project, it was possible to calculate Young’s modulus for the CNT without Poisson’s effect which gives the opportunity to study the influence of this effect on the value of Young’s modulus as presented in Table 5.10. The values in the above table show that when applying Poisson’s effect the calculated Young’s modulus increases by ~0.3 to 0.9 TPa.
The data and results for the elastic modulus calculation from the total energy in the case of zero Poisson’s ratio are presented in Figure 5.12 and Tables 5.11 and 5.12.
Figure 5.12 The total energy-strain plots of all the CNT segments without Poisson’s ratio.
<table>
<thead>
<tr>
<th>Index</th>
<th>Number of C-Atoms</th>
<th>Length, $\lambda_0$ (Å)</th>
<th>$\frac{\partial^2 E}{\partial \varepsilon^2}$ (eV)</th>
<th>$Y_s$ (eV/Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48</td>
<td>6.2</td>
<td>2167.7</td>
<td>349.6</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>8.6</td>
<td>2393.7</td>
<td>278.3</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>11.1</td>
<td>3996.9</td>
<td>360.1</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
<td>13.5</td>
<td>4422.7</td>
<td>327.6</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
<td>16.0</td>
<td>6125.6</td>
<td>382.6</td>
</tr>
<tr>
<td>6</td>
<td>128</td>
<td>18.5</td>
<td>6407.0</td>
<td>346.3</td>
</tr>
<tr>
<td>7</td>
<td>144</td>
<td>20.9</td>
<td>6940.7</td>
<td>332.1</td>
</tr>
<tr>
<td>8</td>
<td>160</td>
<td>23.4</td>
<td>8256.4</td>
<td>352.8</td>
</tr>
<tr>
<td>9</td>
<td>176</td>
<td>25.9</td>
<td>9253.5</td>
<td>357.3</td>
</tr>
<tr>
<td>10</td>
<td>192</td>
<td>28.3</td>
<td>10893.6</td>
<td>384.9</td>
</tr>
</tbody>
</table>

**Table 5.11** The calculated $Y_s$ values from the strain energy for zero Poisson ratio.

<table>
<thead>
<tr>
<th>Area</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Without Poisson’s Ratio</td>
<td>With Poisson’s Ratio</td>
<td>Without Poisson’s Ratio</td>
</tr>
<tr>
<td>Maximum value of Young’s Modulus (TPa)</td>
<td>4.14</td>
<td>4.72</td>
<td>2.34</td>
</tr>
<tr>
<td>Minimum value of Young’s Modulus (TPa)</td>
<td>2.99</td>
<td>2.80</td>
<td>1.69</td>
</tr>
<tr>
<td>Average Value of Young’s Modulus (TPa)</td>
<td>3.73</td>
<td>3.90</td>
<td>2.11</td>
</tr>
</tbody>
</table>

**Table 5.12** Young’s modulus extracted from the total energies for SWCNT (4, 4) based on different definitions of the area for two situations, with and without applying Poisson’s ratio.
The energy-derived moduli values are larger by ~0.5 TPa compared to those calculated using forces in Table 5.10. This difference between the forces and the energies was ~1.5 TPa when the Poisson’s effect was applied meaning that the values without Poisson’s agree better between the two methods. Still, these values of Young’s modulus are in good agreement with the values presented before in the literature review in Chapter two.

As can be seen from the above table the values of Young’s modulus extracted from the total energy calculations, with and without Poisson’s ratio, differ by approximately 0.1 to 0.6 TPa. This means that Poisson’s effect does not have as significant an influence on the energy calculations compared to the impact on the forces calculations where the differences were in the 0.3 to 0.9 TPa range.

In order to give a clear picture about the average values of Young’s modulus for the SWCNT (4, 4) Table 5.13 lists these values that were extracted from the total force and energy in both situations, with and without Poisson’s ratio, for the three definitions of the area.

<table>
<thead>
<tr>
<th>Area</th>
<th>8 C-atoms</th>
<th>CNT-ring</th>
<th>Cross-sectional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Without Poisson’s Ratio</td>
<td>With Poisson’s Ratio</td>
<td>Without Poisson’s Ratio</td>
</tr>
<tr>
<td>Y from Force Calculations (TPa)</td>
<td>2.86</td>
<td>2.17</td>
<td>1.62</td>
</tr>
<tr>
<td>Y from Total Energy Calculations (TPa)</td>
<td>3.73</td>
<td>3.90</td>
<td>2.11</td>
</tr>
</tbody>
</table>

Table 5.13 Average values of Young’s modulus for SWCNT (4, 4) based on different definitions of the area for two situations, with and without applying Poisson’s ratio. These values were calculated by performing the force and the energy calculations.
5.1.2.3 The Electron Densities

The electron densities were calculated by ATK for the CNT (4, 4) segments that were used in this project in attempting to understand the change in these densities between the segments. Also, total forces and transport properties are likely related to sudden changes (with respect to energy) in density of state and electron density. More advanced calculations of the electron density and total density of states for each system are desirable and constitute future work in this project. A VNL file containing the values of the electron density is produced for each segment. Then the VNL software was used to view a contour, isosurface, and volume plots for these electron densities. Although no extensive examinations of such plots have been made, some examples are shown below with brief comments. Electron density was calculated for all the segments; however, the figures for CNT with, 48, 96, and 144 C-atoms will be chosen to be presented.

1) Contour Plots: The electron densities (arbitrary units) for the CNT segments with 48, 96, and 144 atoms are shown in Figures 5.13, 5.14, 5.15 respectively.

![Figure 5.13 Contour plot representing electron density for CNT (4, 4) with 48 atoms.](image)
In the above graphs, the values of the electron density are represented by colors; matching the color with the scale shown to the right gives an idea about the electron density in the structure. The highest electron density is between the C-atoms (see the red dots in the graphs), and it is less in the other directions. The blue color inside the CNT shows that the electron density is zero in that region. As was expected, the change in length should not affect the behavior of the electron density in the CNT because it consists of the same type of atoms and one repeated unit cell. The graphs above show a high symmetry in the electron density behavior in the CNT segments.

2) Isosurface plots: An electron density isosurface is a surface showing one constant value of the electron density in the structure. Two different values for this surface
were chosen for each CNT segment. The figures below present the electron density isosurfaces for the three chosen CNT segments.

**Figure 5.16** Isosurface plots representing electron density for CNT (4, 4) with 48 atoms.

**Figure 5.17** Isosurface plots representing electron density for CNT (4, 4) with 96 atoms.

**Figure 5.18** Isosurface plots representing electron density for CNT (4, 4) with 144 atoms.

The first image in each graph shows the isosurface with high electron density; as can be seen the electrons have the highest density between two C-atoms which agrees
with the conclusion in the contour plots. Also, it seems that the electron density is very low, almost zero, at the center of each hexagonal in the CNT.

3) Volume plots: the graphs of the electron density in the CNT segments here will be shown in 3-dimensions as seen in the following graphs.

**Figure 5.19** Volume plot representing electron density for CNT (4, 4) with 48 atoms.

**Figure 5.20** Volume plot representing electron density for CNT (4, 4) with 96 atoms.

**Figure 5.21** Volume plot representing electron density for CNT (4, 4) with 144 atoms.
These graphs give a better view of the electron clouds around the C-atoms, and these graphs agree with the results presented before concerning the highest and the lowest values of the electron density.

As presented above the electron density behavior does not change when the length of the CNT increases. However, the values of the electron density might vary slightly from one segment to another. A suggested topic in the future work will be to extract the values of the electron density and analyze them. Analyzing wave functions and electron densities for CNT segments with even and odd numbers of unit cells is suggested.

### 5.1.2.4 CNT (5, 5) and CNT (6, 6) Analysis

Future work should predict how the elastic modulus depends on the chirality of the carbon nanotube. As introductory work, Young’s modulus was calculated for two different carbon nanotube segments. They are different in chirality which leads to a difference in the radius and number of atoms. However, both tubes are armchair metallic and have the same length. Both CNTS were constructed by the same procedure used previously. The table below presents the information concerning these CNTS beside the information about the CNT (4, 4) for comparison.

<table>
<thead>
<tr>
<th>Chirality</th>
<th>Number of C-Atoms</th>
<th>Radius (Å)</th>
<th>Length (Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNT (4, 4)</td>
<td>48</td>
<td>2.72</td>
<td>6.2</td>
</tr>
<tr>
<td>CNT (5, 5)</td>
<td>60</td>
<td>3.40</td>
<td>6.2</td>
</tr>
<tr>
<td>CNT (6, 6)</td>
<td>72</td>
<td>4.10</td>
<td>6.2</td>
</tr>
</tbody>
</table>

**Table 5.14** The atomic information for CNT segments with chiralities (4, 4), (5, 5) and (6, 6).
These two new systems were stretched in the same manner that was presented before, and Young’s modulus was calculated by following the same methods and procedures that were used for the CNT (4, 4). The forces and energies were calculated in the case when Poisson’s ratio is considered. After plotting the forces and the energies versus the strains for both CNTs, the $Y_s$ values represented by the slopes of the force-strain graphs and the second derivatives of the energy can be seen in the following table (note that the results for CNT (4, 4) are taken from the previous calculations):

<table>
<thead>
<tr>
<th>Chirality</th>
<th>Force Calculations $Y_s$ (eV/Å)</th>
<th>Energy Calculations $Y_s$ (eV/Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNT (4, 4)</td>
<td>264.7</td>
<td>385.8</td>
</tr>
<tr>
<td>CNT (5, 5)</td>
<td>314.8</td>
<td>465.8</td>
</tr>
<tr>
<td>CNT (6, 6)</td>
<td>369.7</td>
<td>557.6</td>
</tr>
</tbody>
</table>

**Table 5.15** The $Y_s$ values for CNT (4, 4), (5, 5), and (6, 6) were calculated by using force and energy calculation when the Poisson’s effect is considered.

One can observe from the above table that $Y_s$ values become greater with increasing radius; however, these values do not depend on the area which is not the same for the three CNTs because of their different radii. The values of Young’s modulus can be found by dividing $Y_s$ by the area. The three definitions of the area will be used once again, and the following table presents these values of Young’s modulus:
Table 5.16 Young’s modulus in TPa for SWCNT (4, 4), (5, 5), and (6, 6) based on different definitions of the area with Poisson’s effect. These values were calculated by performing the force and the energy calculations.

According to the above results Young’s modulus seems to depend on the radius, and this result agrees with some work that has been presented before [18]. Excluding the first definition of the area, Young’s modulus gets smaller while increasing the radius.

To summarize this section, Young’s modulus was found to be ~7 TPa for the C-chain test system, which falls in the expected order of magnitude of Young’s modulus for the CNTs. Next, the force-strain plots and the calculations of the second derivative of the total energy were used to calculate $Y_Y$ when Poisson’s effect was applied and when it was not. Then the maximum, minimum, and average $Y_Y$ values were taken to calculate the related Young’s modulus values. Three different areas were used to calculate these values. When the Poisson’s effect was considered, the average values of Young’s modulus were 2.2, 1.2, and 1.4 TPa by using force calculations, and they were 3.9, 1.6, and 1.8 TPa by using the energy calculations. On the other hand, these average values
were found without Poisson’s effect to be 2.9, 1.6, and 1.8 TPa by using the force calculations, and by using the energy calculations the average values of Young’s modulus were 3.7, 2.1, and 2.4 TPa. All of the Young’s modulus values that were calculated are in good agreement with the work that has been done in this field. Electron densities were calculated and viewed for some CNT (4, 4) segments. Finally, some work on the variation of Young’s modulus with the radius was established by calculating Young’s modulus for CNT (5, 5) and CNT (6, 6) segments. A clear conclusion can not be made because more points and CNTs are needed to understand the whole picture concerning the behavior of the Young’s modulus while increasing the radius; the other reason is that there is no clear definition for the area in the CNT calculations which leads to a variation in the values of the Young’s modulus.

5.2 Transport Properties for Strained CNT (4, 4)

This part of the chapter presents the results that were extracted for the CNT in between two copper electrodes. For this system, the values of the current flows in the CNT were calculated when different voltages were applied through the copper electrodes. Next, the calculations of the CNT conductance were performed (see Section 4.2.2 for more details). The carbon nanotube (4, 4), with 48-atoms, in the central region was
strained from zero to 0.01 Å by a step of 0.002 Å. Next the voltage was varied from zero to 0.5 volts by a step of 0.1 volts for each strain, and the current values were calculated and plotted versus the voltages in order to calculate the conductance which is the slope of the I-V curve. The voltage range was chosen to be low, zero to 0.5 volts, for two reasons. The first one is for minimizing the computational time. The second reason is to ensure that the I-V points fall in the linear region of the I-V curve. Table 5.19 shows the I-V points at zero strain and Figure 5.22 shows the plot of these points.

<table>
<thead>
<tr>
<th>Voltage (volts)</th>
<th>Current (µAmp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.1</td>
<td>13.1</td>
</tr>
<tr>
<td>0.2</td>
<td>26.4</td>
</tr>
<tr>
<td>0.3</td>
<td>39.8</td>
</tr>
<tr>
<td>0.4</td>
<td>52.6</td>
</tr>
<tr>
<td>0.5</td>
<td>64.8</td>
</tr>
</tbody>
</table>

Table 5.17 The current and voltage values at zero strain for the CNT (4, 4) in between two copper electrodes.

Figure 5.22 The I-V curve at zero strain for CNT (4, 4) in between two copper electrodes.
The relation between the current and the voltage is:

\[ I = GV \]
\[ \text{Slope} = G \]  

(5.12)

The slope of the above graph represents the electrical conductance:

\[ \text{Slope} = G = 130.3 \times 10^{-6} (\Omega)^{-1} = 130 \mu S \]
\[ \mu S \equiv (\text{micro siemens}) \]

The accepted value of the conductance for a long single-wall CNT is equal to \( 2G_0 \) where \( G_0 = 77 \mu S \) (see the related section in Chapter 2). However, the value of the conductance for a SWCNT as shown above is less by 24 \( \mu S \) than the accepted value. This difference in the values of the conductance is due to the use of a short CNT segment and the coupling with the copper electrodes. The conductance of the CNT is expected to become less while stretching the CNT, i.e., increasing the strain [37, 38].

The following pages present the tables that contain the I-V points for each strain that was applied on the CNT. These tables will be followed by the graphs showing the plot of the I-V curves for each strain.

<table>
<thead>
<tr>
<th>Voltage (volts)</th>
<th>Current (( \mu \text{Amp} ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.1</td>
<td>13.0</td>
</tr>
<tr>
<td>0.2</td>
<td>26.2</td>
</tr>
<tr>
<td>0.3</td>
<td>39.4</td>
</tr>
<tr>
<td>0.4</td>
<td>52.1</td>
</tr>
<tr>
<td>0.5</td>
<td>64.3</td>
</tr>
</tbody>
</table>

*Table 5.18 The current and voltage values for a strain of 0.16% for the CNT (4, 4) in between two copper electrodes.*
Figure 5.23 The I-V curve at strain=0.16% for CNT (4, 4) in between two copper electrodes.

Table 5.19 The current and voltage values strain=0.16% for the CNT (4, 4) in between two copper electrodes.
Figure 5.24 The I-V curve at strain=0.32% for CNT (4, 4) in between two copper electrodes.

Table 5.20 The current and voltage values strain=0.49% for the CNT (4, 4) in between two copper electrodes.
The I-V curve at strain=0.49% for CNT (4, 4) in between two copper electrodes. 

Table 5.21 The current and voltage values strain=0.65% for the CNT (4, 4) in between two copper electrodes.
**Figure 5.26** The I-V curve at strain=0.65% for CNT (4, 4) in between two copper electrodes.

The following table includes all the values of the conductance that were calculated from the above graphs; it also includes the associated strain values.

<table>
<thead>
<tr>
<th>Strain</th>
<th>Conductance (μS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>130.3</td>
</tr>
<tr>
<td>0.0016</td>
<td>129.1</td>
</tr>
<tr>
<td>0.0032</td>
<td>128.2</td>
</tr>
<tr>
<td>0.0049</td>
<td>127.2</td>
</tr>
<tr>
<td>0.0065</td>
<td>126.2</td>
</tr>
</tbody>
</table>

**Table 5.22** The strain values and the associated conductance values for the CNT (4, 4) in between two copper electrodes.
According to the data that are presented in the above graph the conductance of the CNT decreased by ~4 μS when a strain of 0.65% was applied. This drop in the conductance agrees with what was presented in the literature review that the conductance for CNTs decreases when increasing the strain. Another observation is that the conductance decreased more when the first strain was applied and after that it decreased linearly with the strain. A suggested future work is to study this behavior of the conductance when greater values of strain are applied.
Chapter 6

Summary and Conclusions

The research is mainly divided into two parts. The first part is devoted to obtain the atomic forces and total energies for different strains in order to calculate Young’s modulus for multiple short lengths of the single-wall CNT (4, 4). The second part is devoted to extract the current-voltage curves for the stretched CNT (4, 4) in between two copper electrodes in order to calculate the conductance and study its relation with the strain. These mechanical and electrical parameters were calculated using the Atomistix ToolKit software (ATK), the main engine to compile the input files. These files were written by the NanoLanguage scripts in the Notepad ++ and they were executed using the Cluster version of the ATK.

In the first part a speculative atomistic-level approach was tried for stretching a test carbon atomic chain and various lengths of carbon nanotube segments. Then the classical or macroscopic relations between the stress, strain, and the energy were applied in analyzing the atomic forces and total energies and calculating Young’s modulus.
Three different definitions for the area were defined in order to apply these classical relations. One was defined by considering the cross-sectional areas of the end atoms, and the other one was the area of the end-ring for the CNT. The last area was the cross-sectional area of the whole CNT-end.

In order to get familiarized with the approach and to test it, a linear chain composed of 12 carbon atoms between two copper electrodes was developed. This chain was stretched and the atomic forces were extracted, then Young’s modulus was calculated and found to be ~7TPa which falls in the expected order of magnitude of Young’s modulus for CNTs. This result confirmed the ability of using this proposed approach to study the mechanical behavior of the CNTs. Also, it proved that the macroscopic relations might be applied in the atomistic level in order to calculate the mechanical properties such as Young’s modulus.

As the next step the different segments of the CNT (4, 4) were constructed and their atomic forces and total energies were obtained. Young’s modulus was found for each segment by using two different methods; the first method was by performing the force calculation, and the other method was by evaluating the second derivative of the total energy numerically. These calculations were performed in two situations regarding the Poisson’s effect; one was by applying Poisson’s effect and the other one was without applying it. The calculated elastic moduli and end forces exhibited an oscillatory-like dependence on segment length. In the absence of computational error, it was speculated that this behavior was related to an odd-even or parity-like effect related to the numbers of unit cells making up the CNT segment. Based on the force calculations, the average values for Young’s modulus were 2.18, 1.24, and 1.41 TPa when Poisson’s effect was
applied, and they were 2.87, 1.63, and 1.63 TPa without Poisson’s effect. Based on the total energy calculations the average values for Young’s modulus were 3.93, 2.23, and 2.54 TPa when Poisson’s effect was applied, and they were 3.76, 2.13, and 2.42 TPa without Poisson’s effect. Three average values were presented because three areas were defined. Young’s modulus, according to the presented work in this research, varies significantly by the definition of the area. Also, it can be concluded that applying Poisson’s effect decreases the value of Young’s modulus the figures of the electron densities of the CNT (4, 4) segments were presented. No apparent significant changes occur in the behavior of the electron density among the C-atoms in the CNT when the length of the CNT increases.

Due to the revealed results, the approach that was used in this project for stretching the CNT and applying macroscopic relations at the atomistic-level works and we were able to calculate the Young’s modulus for single-wall carbon nanotube segments. The ranges of calculated values of Young’s modulus are in good agreement with the values reported by other groups.

Concerning the second part of the project, the electrical conductance is obtained from the current-voltage curves for a stretched single wall metallic carbon nanotube length placed between two copper contacts. The results showed that the conductance at zero strain for this coupled CNT is \( \sim 130 \mu S \), and this value decreases while increasing the strain on the CNT. It decreased to \( \sim 4 \mu S \) when a strain of 0.65% was applied. This value of the conductance and its behavior with strain is in a good agreement with the results found by other groups.
Many other topics and ideas related to the present research can be developed by using the Atomistix software package and the approach that was used in this project. Regarding the mechanical and the electrical properties, a future study can be done to address the following issues:

- Investigating the oscillatory behavior of Young’s modulus with the number of unit cells composing the nanotube segment.
- Calculating Young’s modulus for different chiralities, i.e., different radii. Introductory work has been done in this project and the results showed that Young’s modulus decreases when increasing the radius.
- Calculating the tensile strength which is the stress point at which the CNT starts to deform. This would likely require structural relaxations.
- Studying the influence of defects on Young’s modulus and the conductance of the CNT; for example, remove one atom or more from the CNT.
- Adding another type of atom(s) to the surface of the CNT or inside it and study the influence of this process.
- Applying the same procedure that was presented in this work on a double wall CNT.
- Joining two CNTs with different chiralities together.
- Having two CNTs between the contacts with a gap in between them and study the conductance with increasing the separation of the gap.
- Having the CNT tilted with an angle between the electrodes.
Appendix A

Python Script for the C-Chain

The Nanolanguage script that is written in Notepad ++, and executed on ATK to generate and stretch the C-chain in between two copper electrodes, two-probe system, is given by:

```python
print ""
"""""
# "Cu-12Cchain-Cu_streched"
# Sets up a open system (two-probe) configuration:
# A Carbon chain is streched in between two Copper electrodes.
# The atomic forces and energies are calculated.
# (C) Atomistix 2008
# """""

from ATK.TwoProbe import *
from numpy import *
from ATK.MPI import processIsMaster
import ATK

#Open a VNL file
fileName = "Cu-12Cchain-Cu_streched.vnl"

# Define constants for the fcc lattice
# a:lattice constant, d:Cu-Cu bond length, d_sep:Cu layer separation
a = 3.6150
v = a/sqrt(2)
d_sep = a/sqrt(3)

# Define vector generators for 3x3 layers
u = array([1.0, 0.0, 0.0])*d
v = array([0.5, sqrt(3)/2.0, 0.0])*d
du = array([0.5, 1.0/2.0/sqrt(3), 0.0])*d

# Setup the A layer
n = 3
A = reshape(zeros(3*n*n), (n*n,3))*1.0
for j in range(n):
    for i in range(n):
        A[i+n*j] = i*u + j*v

# Setup the B and C layers
```
B1 = A + 1.0*du
B2 = A + 1.0*du
C = A + 2.0*du

# Central region elements
scattering_elements = 18*[Copper]
scattering_elements += 12*[Carbon]
scattering_elements += 18*[Copper]

# Electrode elements and positions
electrode_elements = 3*[Copper]
electrode_positions = [
    [0.0, 0.0, 0.0],
    [0.5*d, d*0.5/sqrt(3), 1*d_sep],
    [1.0*d, d*1.0/sqrt(3), 2*d_sep]
] * Angstrom

# Electrode unit cell
electrode_unitcell = [
    [d, 0.0, 0.0],
    [0.5*d, sqrt(3)/2.0*d, 0.0],
    [0.0, 0.0, 3.0*d_sep]
] * Angstrom

# Construct the Electrode
electrode = PeriodicAtomConfiguration(
    electrode_unitcell,
    electrode_elements,
    electrode_positions
)

# Set the calculation parameters
exchange_correlation_type = LDA.PZ

# Reduce basis set size for Cooper
basis_set_params_Cu = basisSetParameters(
    type = SingleZeta,
    element = Copper
)

# Reduce basis set size for Carbon
basis_set_params_C = basisSetParameters(
    type = DoubleZetaPolarized,
    element = Carbon
)

# Set k-points for electrodes
bz_int_param = brillouinZoneIntegrationParameters((3,3,50))

# Create parameters for electrodes
electrode_params = ElectrodeParameters(
brillouin_zone_integration_parameters = bz_int_param

# Tolerance for convergence (default)
iteration_control_params = iterationControlParameters(tolerance = 1e-4,
                                                max_steps = 200)

# Collect parameters into a two-probe calculation method
method = TwoProbeMethod(electrode_params, electrode_params,
                         basis_set_parameters = [basis_set_params_Cu, basis_set_params_C],
                         exchange_correlation_type = exchange_correlation_type,
                         iteration_control_parameters = iteration_control_params)

# Specify verbosity and checkpoint file
runtime_params = runtimeParameters(verbosity_level = 10,
                                     checkpoint_filename = 'Cu-C-chain-Cu_streched_0_scf.nc')

# Stretching the C-chain
# Define AB-Cchain-BC central region
B1[:,2] += d_sep

# streching the initial positions according to: Zi = Zio + iZ, Z=increment
import numpy

#The variable to change every time the C-chain is stretched
Z_increament = 0.02
dest_CuC = 1.6
dest_CC = 1.27
initial_Cchainlength=dest_CC*11
length_change= 11*Z_increament
new_Cchainlength=initial_Cchainlength + length_change

Cchain_positions = [
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 1*dest_CC + 1*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 2*dest_CC + 2*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 3*dest_CC + 3*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 4*dest_CC + 4*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 5*dest_CC + 5*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 6*dest_CC + 6*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 7*dest_CC + 7*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 8*dest_CC + 8*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 9*dest_CC + 9*Z_increament),
    (1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 10*dest_CC + 10*Z_increament),
]
(1.5*d, sqrt(3)*d/2.0, d_sep + dest_CuC + 11*dest_CC + 11*Z_increament),
]  
B2[:,2] += d_sep + dest_CuC + new_Cchainlength + dest_CuC
C[:,2] += d_sep + dest_CuC + new_Cchainlength + dest_CuC + d_sep

# Central region positions
scattering_positions = concatenate((A,B1,Cchain_positions,B2,C)).tolist()
scattering_positions *= Angstrom

# Two-probe configuration
# Define a TwoProbeConfiguration with the electrodes and central part
mytwo_probe = TwoProbeConfiguration(
    (electrode, electrode),
    scattering_elements,
    scattering_positions,
    electrode_repetitions=[[3,3],[3,3]],
    equivalent_atoms = ([0,4],[2,43])
)

# Save the structure to the VNL file that was opened previously
myVNL_file = VNLFile(fileName)
myVNL_file.addToSample(mytwo_probe, "Cu-C-chain-Cu_streched_0")

##########################################
# Calculate physical properties
##########################################

# Perform SCF calculation with chosen parameters
scf = executeSelfConsistentCalculation(
    atomic_configuration = mytwo_probe,
    method = method,
    runtime_parameters = runtime_params
)

# Calculate the system energy
total_energy = calculateTotalEnergy(self_consistent_calculation = scf)

# Calculate the system force
atomic_forces = calculateAtomicForces(self_consistent_calculation = scf)

if processIsMaster(): nlPrint(total_energy, 'Total energy')
if processIsMaster(): nlPrint(atomic_forces, 'Total Force')
if processIsMaster(): nlPrint(Z_increament, 'Z_increment')
if processIsMaster(): nlPrint(length_change, 'length_change')
Appendix B

Python Script for the CNT (4, 4)

The Nanolanguage script that is written in Notepad ++, and executed on ATK to generate and stretch the isolated CNT (4, 4) with 48 C-atoms, is given by:

```python
from ATK.KohnSham import *
from ATK.MPI import processIsMaster
from numpy import *

# Define elements
elements = [Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon,
            Carbon, Carbon, Carbon, Carbon]  # Define elements

###################################
definitions
###################################
#CNT length
initial_CNTlength=6.15744066

#The coordinates of the center of the CNT
X0=7.606521
Y0=7.606521

# Define (x, y) coordinates
CNT_coordinates = array([[10.32234085, 7.6065209, 0],
                         [9.5268956, 9.5268956, 0],
                         [7.6065209, 10.32234085, 0],
                         [5.6861462, 9.5268956, 0],
                         [4.89070095, 7.6065209, 0],
                         [5.6861462, 5.6861462, 0],
                         [7.6065209, 4.89070095, 0],
                         [9.5268956, 5.6861462, 0],
                         [9.95848997, 8.96443087, 0],
                         [8.96443087, 9.95848997, 0],
                         [7.6065209, 8.96443087, 0],
                         [5.6861462, 7.6065209, 0],
                         [4.89070095, 5.6861462, 0],
                         [3.12558121, 4.89070095, 0],
                         [2.32903551, 5.6861462, 0],
                         [1.53249021, 7.6065209, 0],
                         [0.73594451, 9.5268956, 0],
                         [1.53249021, 10.32234085, 0],
                         [2.32903551, 11.11888616, 0],
                         [3.12558121, 11.91543151, 0],
                         [4.89070095, 12.71197686, 0],
                         [6.65584651, 13.50852221, 0],
                         [8.42129207, 14.30506756, 0],
                         [10.18653762, 15.10151291, 0],
                         [11.95178318, 15.89805826, 0],
                         [13.71702873, 16.69460361, 0],
                         [15.48227429, 17.49114896, 0],
                         [17.24752084, 18.28769431, 0],
                         [19.01276639, 19.08423966, 0],
                         [20.77801195, 19.88078500, 0],
                         [22.54325749, 20.67733035, 0],
                         [24.30850305, 21.47387569, 0],
                         [26.07374860, 22.27042104, 0],
                         [27.83899415, 23.06696639, 0],
                         [29.60423970, 23.86351173, 0],
                         [31.36948525, 24.66005708, 0],
                         [33.13473080, 25.45660242, 0],
                         [34.89997635, 26.25314777, 0],
                         [36.66522190, 27.04969311, 0],
                         [38.43046745, 27.84623846, 0],
                         [40.19571300, 28.64278380, 0],
                         [41.96095855, 29.43932914, 0],
                         [43.72620410, 30.23587449, 0],
                         [45.49144965, 31.03241983, 0],
                         [47.25669520, 31.82896518, 0],
                         [49.02194075, 32.62551052, 0],
                         [50.78718630, 33.42195586, 0],
                         [52.55243185, 34.21840121, 0],
                         [54.31767740, 35.01484655, 0],
                         [56.08292295, 35.81129189, 0],
                         [57.84816850, 36.60773723, 0],
                         [59.61341405, 37.40418258, 0],
                         [61.37865960, 38.20062792, 0],
                         [63.14390515, 39.00707326, 0],
                         [64.90915070, 39.81351860, 0],
                         [66.67439625, 40.62096394, 0],
                         [68.43964180, 41.42840928, 0],
                         [70.20488735, 42.23485462, 0],
                         [71.96013290, 43.04129996, 0],
                         [73.72537845, 43.84774530, 0],
                         [75.49062400, 44.65419064, 0],
                         [77.25586955, 45.46063600, 0],
                         [79.02111510, 46.26708134, 0],
                         [80.78636065, 47.07352668, 0],
                         [82.55160620, 47.88097202, 0],
                         [84.31685175, 48.68741736, 0],
                         [86.08209730, 49.49386270, 0],
                         [87.84734285, 50.29030804, 0],
                         [89.61258840, 51.09675338, 0],
                         [91.37783395, 51.89319872, 0],
                         [93.14307950, 52.69964406, 0],
                         [94.90832505, 53.50608940, 0],
                         [96.67357060, 54.31253474, 0],
                         [98.43881615, 55.11897008, 0],
                         [100.20406170, 55.92541542, 0]]
```

```
# Shifting the center of the CNT to calculate the reductions in (x, y)
coordinates = CNT_coordinates - [X0, Y0, 0.0]

# Parameters
exchange_correlation_type = LDA.PZ

electron_density_parameters = electronDensityParameters(
    mesh_cutoff = 130.0*Rydberg
)
basis_set_parameters = basisSetParameters(
    type = DoubleZetaPolarized
)
iteration_control_parameters = iterationControlParameters(
    tolerance = 0.00001,
    criterion = IterationControl.Strict,
    max_steps = 200
)

# Collect the parameters in the method

kohnsham_method = KohnShamMethod(
    exchange_correlation_type = exchange_correlation_type,
    electron_density_parameters = electron_density_parameters,
    basis_set_parameters = basis_set_parameters,
    iteration_control_parameters = iteration_control_parameters
)

# Stretching and Calculate physical properties

# Set verbosity level so that all energy components are printed
import ATK
verbosity_level=ATK.verbosityLevel()
ATK.setVerbosityLevel(10)
ATK.setVerbosityLevel(verbosity_level)
import numpy

# E=strain, R=reduction ratio, v=Poisson's ratio
for Z_increament in numpy.arange(0.000,0.021,0.001):
    length_change= 5*Z_increament
    E=length_change/initial_CNTlength
    R=1-sqrt(1/(1+E))
    v=R/E
    Change_in_coordinates = R*coordinates
    New_CNT_coordinates= CNT_coordinates - Change_in_coordinates
    CNT_Full_coordinates=New_CNT_coordinates+
        [[ 0, 0,       0.  ],
         [ 0, 0,   1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,    0.        ],
         [ 0, 0,    1.23148812+ 1*Z_increament],
         [ 0, 0,   2.46297622 + 2*Z_increament],
         [ 0, 0,   3.69446445 + 3*Z_increament],
         [ 0, 0,   2.46297622 + 2*Z_increament],
     ]


CNT_Full_coordinates = CNT_Full_coordinates*Angstrom

cnt = MoleculeConfiguration(elements, CNT_Full_coordinates)

scf = executeSelfConsistentCalculation(
    atomic_configuration=CNT,
    method=kohnsham_method
)
if processIsMaster(): nlPrint(Z_increament, 'Z_increament')
if processIsMaster(): nlPrint(length_change, 'Length change')
if processIsMaster(): nlPrint(E, 'strain')
if processIsMaster(): nlPrint(v, 'Poissons ratio')

total_energy = calculateTotalEnergy(self_consistent_calculation = scf)
if processIsMaster(): nlPrint(total_energy, 'Total energy')

atomic_forces = calculateAtomicForces(self_consistent_calculation = scf)
if processIsMaster(): nlPrint(atomic_forces)
Note that the above script is used to produce the other CNT segments by changing the elements and the coordinates. Also, the same script is used to perform the calculations, without considering Poisson’s effect, by setting “R” to be zero.

Appendix C

Python Script for the Two Probe System

The Nanolanguage script that is shown below is written in Notepad ++, and executed on ATK to generate and stretch the CNT in between two copper electrodes, two-probe system, in order to calculate the current for different voltages:

```python
print ""
# "Cu-CNT-Cu_unrelaxed_mod"
# Sets up a open system (two-probe) configuration:
# A CNT is stretched in between two Copper electrodes.
# The current is calculated for multiple voltages.
# (C) Atomistix 2008
#______________________________________________________
""
import ATK
from ATK.TwoProbe import *
from numpy import *
import ATK.MPI import processIsMaster

# Define constants for the fcc lattice, a: lattice constant, d:Cu-Cu bond length, d_sep:Cu layer separation
a = 3.6150
d = a/sqrt(2)
d_sep = a/sqrt(3)

# Define vector generators for 3x3 layers
u = array([1.0, 0.0, 0.0])*d
v = array([0.5, sqrt(3)/2.0, 0.0])*d
du = array([0.5, 1.0/2.0/sqrt(3), 0.0])*d
```

```
# Setup the A layer
n = 5
A = reshape(zeros(5*3*n), (n*n,3))*1.0
for j in range(n):
    for i in range(n):
        A[i+n*j] = i*u + j*v
# Setup the B and C layers
B1 = A + 1.0*du
B2 = A + 1.0*du
C = A + 2.0*du

# Define AB-Cchain-BC central region
B1[:,2] += d_sep
dest_CuC = 1.6
dest_CC = 1.42
# The variabl eto change every time
Z_increment = 0.002
initial_CNTlength=6.15744066
length_change=5*Z_increment
#Define the strain
E=length_change/initial_CNTlength
new_CNTlength=initial_CNTlength + length_change

CNT_positions = array([  
    [10.32234097, 7.60652113, 0. + d_sep + dest_CuC],
    [ 9.52689552, 9.52689552, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 7.60652113, 10.32234097, 0. + d_sep + dest_CuC],
    [ 5.68614626, 9.52689552, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 4.89070082, 7.60652113, 0. + d_sep + dest_CuC],
    [ 5.68614626, 5.68614626, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 7.60652113, 4.89070082, 0. + d_sep + dest_CuC],
    [ 9.52689552, 5.68614626, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 9.95849037, 8.96443081, 0. + d_sep + dest_CuC],
    [ 8.30942726, 10.22980118, 1.23148811 + d_sep + dest_CuC+1*Z_increment],
    [ 6.24861097, 9.95849037, 0. + d_sep + dest_CuC],
    [ 4.98324013, 8.30942726, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 5.25455189, 6.24861097, 0. + d_sep + dest_CuC],
    [ 6.903615 , 4.98324013, 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [ 8.96443081, 5.25455189, 0. + d_sep + dest_CuC],
    [ 10.22980118, 6.903615 , 1.23148811 + d_sep + dest_CuC +1*Z_increment],
    [10.32234097, 7.60652113, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
    [ 9.52689552, 9.52689552, 3.69446445 + d_sep + dest_CuC +3*Z_increment],
    [ 7.60652113, 10.32234097, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
    [ 5.68614626, 9.52689552, 3.69446445 + d_sep + dest_CuC +3*Z_increment],
    [ 4.89070082, 7.60652113, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
    [ 5.68614626, 5.68614626, 3.69446445 + d_sep + dest_CuC +3*Z_increment],
    [ 7.60652113, 4.89070082, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
    [ 9.52689552, 5.68614626, 3.69446445 + d_sep + dest_CuC +3*Z_increment],
    [ 9.95849037, 8.96443081, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
    [ 6.24861097, 9.95849037, 2.46297622 + d_sep + dest_CuC +2*Z_increment],
])
CNT_positions = CNT_positions + [1.34015, -2.44116, 0]

B2[:,2] += d_sep + dest_CuC + new_CNTlength + dest_CuC
C[:,2] += d_sep + dest_CuC + new_CNTlength + dest_CuC + d_sep

# Central region elements
scattering_elements = 50*[Copper]
scattering_elements += 48*[Carbon]
scattering_elements += 50*[Copper]

# Central region positions
scattering_positions = concatenate((A,B1,CNT_positions,B2,C)).tolist()
scattering_positions *= Angstrom

# Electrode elements and positions
electrode_elements = 3*[Copper]
electrode_positions = [
    [0.0, 0.0, 0.0],
    [0.5*d, d*0.5/sqrt(3), 1*d_sep],
    [1.0*d, d*1.0/sqrt(3), 2*d_sep]
] * Angstrom

# Electrode unit cell
electrode_unitcell = [
    [d, 0.0, 0.0],
    [0.5*d, sqrt(3)/2.0*d, 0.0],
    [0.0, 0.0, 3.0*d_sep]
] * Angstrom

# Construct the Electrode
electrode = PeriodicAtomConfiguration(electrode_unitcell,
electrode_elements,
electrode_positions
)

# Two-probe configuration
# Define a TwoProbeConfiguration with the electrodes and central part
mytwo_probe = TwoProbeConfiguration(
    (electrode,electrode),
    scattering_elements,
    scattering_positions,
    electrode_repetitions=[[5,5],[5,5]],
    equivalent_atoms = [(0,0),(2,124)]
)

############################
#Parameters
############################
# Reduce basis set size for copper
basis_set_params_Cu = basisSetParameters(
    type = SingleZeta,
    element = Copper
)

# Reduce basis set size for carbon
basis_set_params_C = basisSetParameters(
    type = DoubleZetaPolarized,
    element = Carbon
)

# Set k-points for electrodes
bz_int_param = brillouinZoneIntegrationParameters( (3,3,50) )

# Create parameters for electrodes
electrode_params = ElectrodeParameters(
    brillouin_zone_integration_parameters = bz_int_param
)

# Tolerance for convergence (default)
iteration_control_params = iterationControlParameters(
    tolerance = 1e-4,
    max_steps = 300
)

electron_density_parameters = electronDensityParameters(
    mesh_cutoff = 150.0*Rydberg
)

# Collect parameters into a two-probe calculation method
method = TwoProbeMethod(
    (electrode_params,electrode_params),
    basis_set_parameters = [basis_set_params_Cu, basis_set_params_C],
    iteration_control_parameters = iteration_control_params,
    electron_density_parameters = electron_density_parameters
)
# Specify verbosity and checkpoint file
runtime_params = runtimeParameters(  
    verbosity_level = 0,  
    checkpoint_filename = 'Cu-CNT-Cu_streched_0_scf.nc'  
)

# Perform SCF calculation with chosen parameters
scf = executeSelfConsistentCalculation(  
    atomic_configuration = mytwo_probe,  
    method = method,  
    runtime_parameters = runtime_params  
)

######################################################################
# Calculate physical properties
######################################################################

total_energy = calculateTotalEnergy(self_consistent_calculation = scf)
if processIsMaster(): nlPrint(total_energy,'Total energy')
if processIsMaster(): nlPrint(Z_increament, 'Z_increament')
if processIsMaster(): nlPrint(length_change, 'Length change')
if processIsMaster(): nlPrint(E, 'strain')

#####################
#I-V Curves
#####################

# Run bias from 0.0 and 0.6 in steps of 0.1
for voltage in numpy.arange(0.0, 0.6, 0.1):
    dft_method = TwoProbeMethod(  
        electrode_parameters=(electrode_params,electrode_params),  
        basis_set_parameters = [basis_set_params_Cu , basis_set_params_C],  
        iteration_control_parameters = iteration_control_params,  
        electrode_voltages = (voltage/2.0, -voltage/2.0)*Volt  
    )
    scf = executeSelfConsistentCalculation(  
        atomic_configuration=mytwo_probe,  
        method = dft_method,  
        initial_calculation = scf  
    )
    current = calculateCurrent(scf)
if processIsMaster(): nlPrint(voltage, 'Bias Voltage')
if processIsMaster(): nlPrint(current.inUnitsOf(Ampere), 'Current')
Appendix D

Python Script for the Convergence Test

The complete convergence test script for the isolated CNT with 48 C-atoms is:

```python
from ATK.KohnSham import *
from ATK.MPI import processIsMaster
from numpy import *

# Define elements
elements = [Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon,  
            Carbon, Carbon, Carbon, Carbon]  

# Define coordinates
coordinates = array([[ 10.32234085,   7.6065209 ,    0.        ],
                     [  9.5268956 ,   9.5268956 ,    1.23148812],
                     [  7.6065209 ,  10.32234085,    0.        ],
                     [  5.6861462 ,   9.5268956 ,    1.23148812],
                     [  4.89070095,   7.6065209 ,    0.        ],
                     [  5.6861462 ,   5.6861462 ,    1.23148812],
                     [  7.6065209 ,   4.89070095,    0.        ],
                     [  9.5268956 ,   5.6861462 ,    1.23148812],
                     [  9.95848997,   8.96443087,    0.        ],
                     [  8.30942683,  10.22980153,    1.23148812],
                     [  6.24861093,   9.95848997,    0.        ],
                     [  4.98324027,   8.30942683,    1.23148812],
                     [  5.25455183,   6.24861093,    0.        ],
                     [  6.90361497,   4.98324027,    1.23148812],
                     [  8.96443087,   5.25455183,    0.        ],
                     [ 10.22980153,   6.90361497,    1.23148812],
                     [ 10.32234097,   7.60652113,   2.46297622 ],
                     [  9.52689552,   9.52689552,   3.69446445 ],
                     [  7.60652113,  10.32234097,   2.46297622 ],
                     [  5.68614626,   9.52689552,   3.69446445 ],
                     [  4.89070082,   7.60652113,   2.46297622 ],
                     [  5.68614626,   5.68614626,   3.69446445 ],
                     [  7.60652113,   4.89070082,   2.46297622 ],
                     [  9.52689552,   5.68614626,   3.69446445 ]],
```
# Add them to a configuration
CNT = MoleculeConfiguration(elements, coordinates)

# Parameters
exchange_correlation_type = LDA.PZ

basis_set_parameters = basisSetParameters(
    type = DoubleZetaPolarized
)

iteration_control_parameters = iterationControlParameters(
    tolerance = 0.00001,
    max_steps = 200
)

# Calculate total energy against the mesh cut-off
# Set verbosity level so that all energy components are printed
import ATK

verbosity_level=ATK.verbosityLevel()
ATK.setVerbosityLevel(10)
ATK.setVerbosityLevel(verbosity_level)

print '    MeshCutoff (Rydberg)     TotalEnergy (eV)'
print '------------------------------------------------'
for mesh_cutoff in range(30,200,10)*Rydberg:
    electron_density_parameters = electronDensityParameters( mesh_cutoff = mesh_cutoff)
method= KohnShamMethod(
    exchange_correlation_type = exchange_correlation_type,
    electron_density_parameters = electron_density_parameters,
    basis_set_parameters = basis_set_parameters,
    iteration_control_parameters = iteration_control_parameters
)
print 't', mesh_cutoff.inUnitsOf(Rydberg), 't	t',
    calculateTotalEnergy(method.apply(CNT)).inUnitsOf(eV)

The complete convergence test script for the CNT between the copper surface layers is:

```python
import ATK
from numpy import *
from ATK.MPI import processIsMaster

# Define constants for the fcc lattice
# a: lattice constant, d: Cu-Cu bond length, d_sep: Cu layer separation
a = 3.6150  
d = a/sqrt(2)  
d_sep = a/sqrt(3)

# Define vector generators for 3x3 layers
u = array([1.0, 0.0, 0.0])*d  
v = array([0.5, sqrt(3)/2.0, 0.0])*d  
du = array([0.5, 1.0/2.0/sqrt(3), 0.0])*d

# Setup the A layer
n = 5
A = reshape(zeros(5*3*n), (n*n,3))*1.0
for j in range(n):
    for i in range(n):
        A[i+n*j] = i*u + j*v

# Setup the B and C layers
B1 = A + 1.0*du  
B2 = A + 1.0*du  
C = A + 2.0*du

# Define AB-Cchain-BC central region
dest_CuC = 1.6  
dest_CC  = 1.42  
CNTlength=6.15744066

B1[:,2] += d_sep
CNT_positions = array([[10.32234097, 7.60652113, 0. + d_sep + dest_CuC], [9.52689552, 9.52689552, 1.23148811 + d_sep + dest_CuC], [7.60652113, 10.32234097, 0. + d_sep + dest_CuC], [5.68614626, 9.52689552, 1.23148811 + d_sep + dest_CuC], [4.89070082, 7.60652113, 0. + d_sep + dest_CuC],
```

```
#To Put the CNT at the center of the Cu layer
CNT_positions1 = CNT_positions + [1.34015, -2.44116, 0]

B2[:,2] += d_sep + dest_CuC + CNTlength + dest_CuC
C[:,2] += d_sep + dest_CuC + CNTlength + dest_CuC + d_sep

# Central region elements
scattering_elements = 50*[Copper]
scattering_elements += 48*[Carbon]
scattering_elements += 50*[Copper]
scattering_positions = concatenate((A,B1,CNT_positions,B2,C)).tolist()
scattering_positions *= Angstrom

Cu_CNT_Cu = MoleculeConfiguration(scattering_elements, scattering_positions)

define the exchange correlation type
exchange_correlation_type = LDA.PZ

# Reduce basis set size for copper

basis_set_params_Cu = basisSetParameters(
    type = SingleZeta,
    element = Copper
)

# Reduce basis set size for carbon

basis_set_params_C = basisSetParameters(
    type = DoubleZetaPolarized,
    element = Carbon
)

# Tolerance

iteration_control_parameters = iterationControlParameters(
    tolerance = 0.00001,
    max_steps = 200
)

# Calculate physical properties

import ATK

for mesh_cutoff in range(100,210,10)*Rydberg:
    electron_density_parameters = electronDensityParameters( mesh_cutoff = mesh_cutoff)
    method= KohnShamMethod(
        exchange_correlation_type = exchange_correlation_type,
        electron_density_parameters = electron_density_parameters,
    basis_set_parameters = [basis_set_params_Cu , basis_set_params_C],
    iteration_control_parameters = iteration_control_parameters,
    )

    calculateTotalEnergy(method.apply(Cu_CNT_Cu)).inUnitsOf(eV)
References:

[47]. Mohan Varanasi, “Geometries of Small Cadmium Selenide (CdSe) Clusters” (Ball State University, 2006).
[61]. Magnus Paulsson, Dept. of micro- and nano-technology, NanoDTU, (Technical University of Denmark, February 1, 2008).
[69]. Jose A. Torres, Software Application Scientist, Atomistix Copenhagen, Atomistix Presentation (California March 2007).
[74]. http://docs.python.org/tut/.