

©2014

TIMOTHY ALLEN NIXDORF

ALL RIGHTS RESERVED

A MATHEMATICAL MODEL FOR CARBON NANOSCROLLS

A Thesis

Presented to

The Graduate Faculty of The University of Akron

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

Timothy Allen Nixdorf

August, 2014

A MATHEMATICAL MODEL FOR CARBON NANOSCROLLS

Timothy Allen Nixdorf

Thesis

Approved:

Accepted:

Advisor
Dr. Dmitry Golovoty

Dean of the College
Dr. Chand Midha

Co-Advisor
Dr. J. Patrick Wilber

Dean of the Graduate School
Dr. George R. Newkome

Faculty Reader
Dr. Malena Español

Date

Department Chair
Dr. Timothy Norfolk

ABSTRACT

Carbon nanoscrolls (CNS) have great potential in engineering applications. Understanding the geometries of CNS and their properties can provide insight into the design of nanoscale devices. We study an energy-based model of CNS. We approximate the 2-D scroll of atoms by a 1-D chain of carbon atoms interacting via van der Waals (VDW) and bending forces. A collection of equilibrium configurations is found by evolving various initial spiral geometries using dissipation-dominated gradient flow dynamics. The structure of the final configuration depends on the relative strengths of the bending and VDW forces. The predictions drawn from our simple model are qualitatively consistent with experimental observations [1], [2]. We then approximate the 2-D scroll by a 2-D lattice of atoms interacting via extensional bond forces, VDW forces, and bending forces.

ACKNOWLEDGEMENTS

This research was supported by NSF grant DMS-1009849. I would like to thank my advisors, Dr. Malena Español, Dr. Dmitry Golovaty, and Dr. J. Patrick Wilber, for their time and guidance. I would also like to thank Andrew Marmaduke for helpful discussions as well as his advice and programming expertise. Last but not least, I would like to thank my research partner, Daniel Rhoads, for his collaborative efforts in addition to his technical and emotional support.

TABLE OF CONTENTS

	Page
LIST OF TABLES	vii
LIST OF FIGURES	viii
CHAPTER	
I. INTRODUCTION	1
1.1 Graphene, Carbon Nanotubes, and Carbon Nanoribbons	1
1.2 Carbon Nanoscrolls	2
1.3 Modeling CNS	4
1.4 Gradient Flow Dynamics	7
1.5 Summary of the Project	7
II. ONE DIMENSIONAL MODEL	9
2.1 Modeling Assumptions	9
2.2 Gradient Flow Dynamics	12
2.3 Setting up the Initial Condition	14
III. ONE DIMENSIONAL RESULTS	17
3.1 Types of Shapes	17
3.2 Varying β for $\sigma = 1$	17
3.3 Varying σ	22

3.4	Setting $\Delta r > \sigma$	33
IV.	TWO DIMENSIONAL MODEL	35
4.1	Modeling Assumptions	35
4.2	Gradient Flow Dynamics	38
4.3	Setting up the Initial Condition	40
V.	CONCLUSIONS	46
	BIBLIOGRAPHY	48
	APPENDIX	51

LIST OF TABLES

Table		Page
3.1	Typical shapes found.	18
3.2	Results obtained by varying β	20
3.3	Results obtained by varying σ : $\sigma < 2$	27
3.4	Results obtained by varying σ : $\sigma > 2$	32

LIST OF FIGURES

Figure	Page
1.1 Comparison of graphene-based structures, Left: graphene, Center: MWNT, Right: CNS [3]	3
1.2 Geometry of a graphene sheet	4
1.3 Different types of chirality, A: armchair, B: zigzag, and C: chiral [4]. . .	5
1.4 Lennard Jones curve $E_{LJ} \left(\frac{d_{ij}}{\sigma} \right)$ for $\epsilon = 1$	6
2.1 Planar chain of atoms approximating a cross-section of a CNS	10
2.2 Geometry of the chain and VDW pairs.	10
2.3 Spiral geometry for initial condition	15
3.1 Double-walled tube cross-section	23
3.2 Symmetric pairs of parallel lines	24
3.3 VDW energy from free atom near fixed edge	25
3.4 Transitional case for $2\pi\sigma = 4$ a) original solution b) doubling ρ and increasing n	28
3.5 Degenerate case for $2\pi\sigma = 3$: energy from free atom with edge	29
3.6 Degenerate case for $2\pi\sigma = 3$: simulation result	29
3.7 Simulation result for $2\pi\sigma = 11$, $n = 250$	30
3.8 $2\pi\sigma = 11$: energy from free atom with edge	31

3.9	Simulation results for $\Delta r = 2, \sigma = 1, a)\beta = 1, b)\beta = 100$	34
4.1	Approximated CNS	36
4.2	Bending force	36
4.3	Model bond	37
4.4	Adjacent normals	37
4.5	Lattice initialization	41
4.6	Isolating graphene sheet	41
4.7	Definition of rectangle	42
4.8	Graphene sheet bond orientations	43
4.9	Graphene sheet bond orientations: individual orientations	44
4.10	3D chain	44
4.11	2-D CNS showing bond orientations	45

CHAPTER I

INTRODUCTION

1.1 Graphene, Carbon Nanotubes, and Carbon Nanoribbons

In the modern world, nanotechnology is a popular field of study. Particularly, significant efforts have been expanded to investigate carbon-based nanostructures and graphene. Graphene exhibits remarkable mechanical properties, including exceptional rigidity and strength, in addition to impressive transport properties like high electron mobility [5]. New uses for graphene are constantly being discovered; for example, it can be used to store hydrogen atoms, with possible applications in hydrogen fuel cells [6].

A large number of studies have been done on carbon nanotubes (CNT). A CNT is essentially a sheet of graphene that has been rolled into a tube. Much like graphene sheets, CNT have impressive mechanical and electronic properties [7]. CNT are of two types: single-walled CNT (SWNT) and multi-walled CNT (MWNT). These structures can be sufficiently large to be described within a framework of continuum theory, yet are still small enough for their properties to be dependent on discreteness of the atomistic structure. Some MWNT have been shown to have polygonal cross-sections [8], while it is expected that these cross-sections should be circular. This

polygonization effect was recently described within the framework of a continuum theory [8].

Under some conditions, the shape of a nanotube may be hard to control. For instance, if the temperature is sufficiently increased, a pair of double-walled CNT (DWNT) can even combine into a “bicable” through a “zipping” process that connects the outer tubes into one large loop that encapsulates both of the inner CNT, [9]. Some other graphene shapes include carbon nanoribbons/nanobelts [10], and carbon nanoscrolls (CNS). Carbon nanoribbons are graphene sheets that have a large aspect ratio [10]. A carbon nanoscrolls is a graphene sheet that is rolled up into the shape of a scroll.

1.2 Carbon Nanoscrolls

MWNT have the cross-sectional shape of concentric, closed curves while CNS have the cross-sectional shape of a spiral (see Figure 1.1). Therefore, CNS are similar to MWNT, but they have more freedom to change shape. One reason CNS are of interest is that, under some conditions, a graphene sheet will spontaneously roll up into a CNS [11]. These CNS are considered to exist at a lower energy than the original sheet [11], but they can get stuck in a meta-stable configuration during the rolling process [12]. There has also been a significant amount of research done to investigate the formation of CNS by allowing a graphene sheet to wrap around a CNT [13, 14, 15].

CNS share many of the same characteristics with CNT, including impressive mechanical and electronic properties [16]. There are applications in the field of CO₂

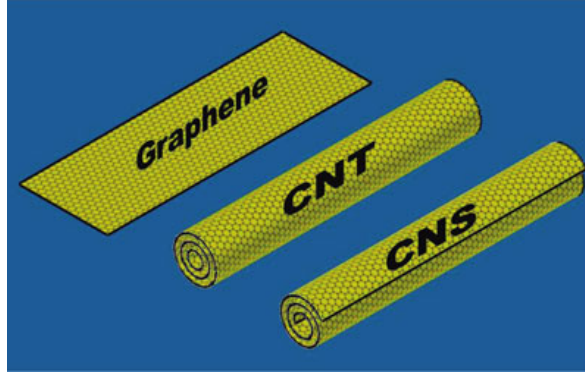


Figure 1.1: Comparison of graphene-based structures, Left: graphene, Center: MWNT, Right: CNS [3]

storage [17]. Important differences between CNS and CNT come into play under loading [16]. Where CNT has an “exact” size that is only stable under very specific conditions, CNS have a “tunable” size that depends on its environment [18]. This torsional instability can be exploited for molecular mass transport [16], [19].

There are several other applications specific to CNS. CNT supported by a substrate have been used as oscillators along their axes, but new research suggests that CNS whose cores are stiffened by CNT can be used as oscillators perpendicular to their axes [20]. Additionally, CNS have been shown to have excellent potential for nanoactuators and nanomotors, even without the CNT stiffeners [21]. Looking at their magnetic properties, arrays of CNS have been theoretically shown to act as a hyperbolic magnetic metamaterial that is magnetically active in a frequency regime for which such materials are extremely rare [22].

Similarities between CNS and CNT suggest that the “polygonization” effects observed in CNT may exist in CNS. This hypothesis is supported by models that

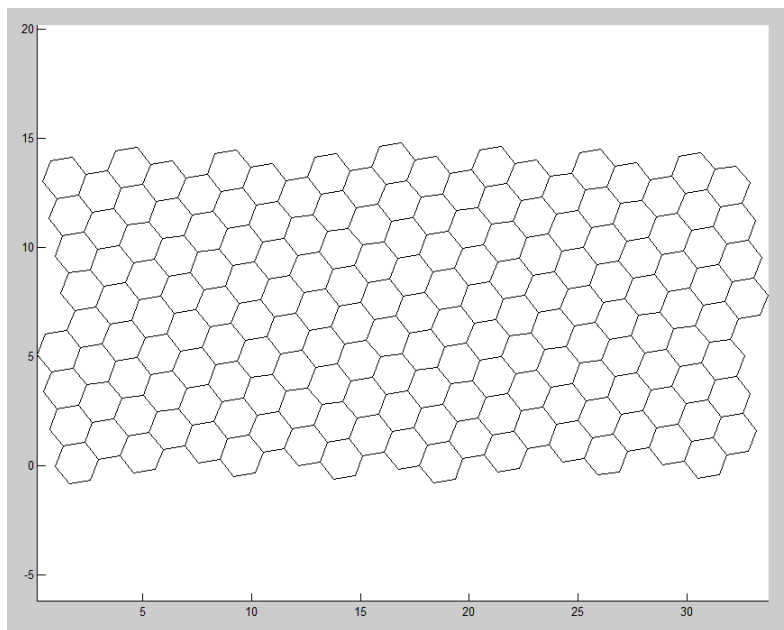


Figure 1.2: Geometry of a graphene sheet

suggest twisted polygonal shapes [23]. We will investigate CNS using a computational model to determine possible equilibrated states.

1.3 Modeling CNS

All graphene-based structures are based on a hexagonal lattice of carbon atoms held together by covalent bonds (see Figure 1.2) [7]. In both CNT and CNS, the structure can be characterized by the orientation (chirality) of the lattice. There are two specific cases called “armchair” (see Figure 1.3A) and “zigzag” (see Figure 1.3B). All other orientations are simply referred to as “chiral” (see Figure 1.3C) [7]. Therefore, a good model should be able to describe any chirality.

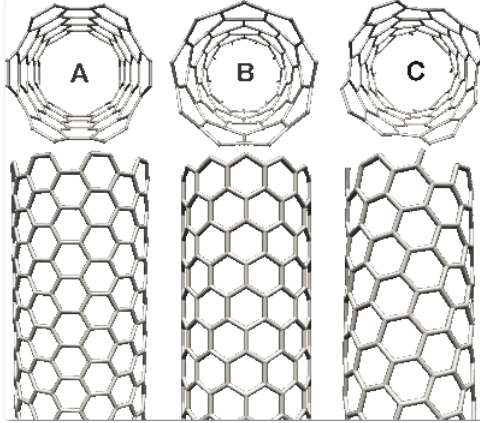


Figure 1.3: Different types of chirality, A: armchair, B: zigzag, and C: chiral [4].

To model the behavior of CNT and CNS, we employ a method based on minimizing an appropriately defined total energy. We use an atomistic model based on positions of atoms. The total energy we define generally consists of three contributions. The first of these is the energy of the covalent bonds between adjacent atoms in the hexagonal lattice. The forces in these bonds are considered to be very strong compared to all other forces within the system. In our model, the covalent bonds are modeled either as linear springs or as rigid rods.

A second contribution to the energy describes the attractive-repulsive interactions between atoms that are not covalently bonded together. This interaction is believed to be critical for explaining polyorganization of CNT and CNS [8]. This interaction is attributed in part to van der Waals effects and is often modeled by the Lennard-Jones potential, [5], [25]. The Lennard-Jones potential energy is

$$E_{LJ}(d_{ij}) = \epsilon \left[\left(\frac{\sigma}{d_{ij}} \right)^{12} - 2 \left(\frac{\sigma}{d_{ij}} \right)^6 \right], \quad (1.1)$$

where d_{ij} is the distance between the atoms i and j , σ is the equilibrium distance, and ϵ is the depth of the potential well. Figure 1.4 shows the Lennard Jones potential

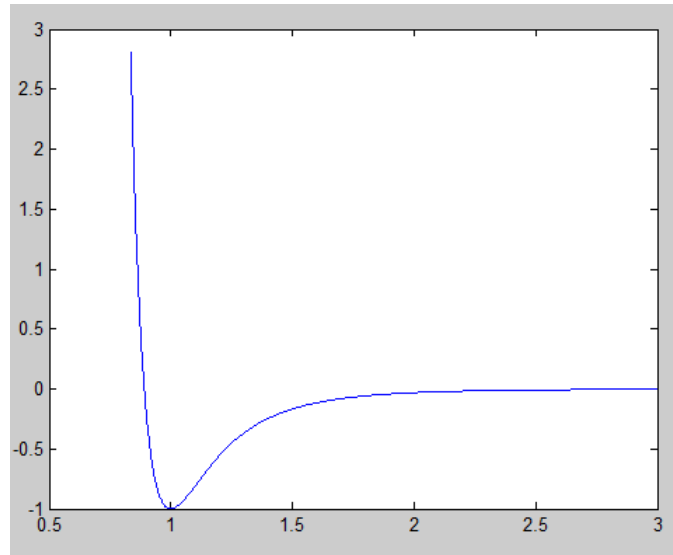


Figure 1.4: Lennard Jones curve $E_{LJ} \left(\frac{d_{ij}}{\sigma} \right)$ for $\epsilon = 1$

as a function of the ratio of the distance between atoms and the equilibrium distance. Observe that this potential has a minimum when $d_{ij} = \sigma$ and converges to 0 as $d_{ij} \rightarrow \infty$.

The final contribution to the total energy is a bending energy, which is minimized when the sheet is flat. It is less clear how this energy should be represented. Some simplified models use a linear torsional spring [26, 25], but more in-depth formulations can be used as well [7]. We will describe our choices of bending below when discussing specific models.

1.4 Gradient Flow Dynamics

An equilibrium state of the graphene sheet corresponds to a configuration of atoms in which the internal energy of the system is at a local minimum. Therefore, to find an equilibrium, we must minimize the internal energy. To do this, we employ a method called gradient flow dynamics, or “steepest descent.” This method moves the state of the system in the direction opposite to the gradient of the internal energy and therefore should find a local minimum of the energy, so long as one exists.

Next we explain how we implement gradient flow dynamics to track the system’s evolution through time. We know that force is related to the total energy of the system by $\vec{F}(\vec{r}_i) = -\vec{\nabla}_{\vec{r}_i} E_{total}$, where \vec{r}_i is the position of the atom i , $\vec{F}(\vec{r}_i)$ is the net force acting on atom i , and E is the total energy of the system. We let

$$\gamma \frac{d\vec{r}_i}{dt} = -\vec{\nabla}_{\vec{r}_i} E, \quad (1.2)$$

where t is time and γ is a constant. Here, we assume that the total energy is a function of positions of atoms. Note that \vec{r}_i can be replaced by any generalized coordinate of the atom i .

1.5 Summary of the Project

We formulate a one-dimensional atomistic model of a CNS with modifiable parameters. We establish an initial condition of a scroll-like configuration. We then evolve this configuration toward a local equilibrium via gradient flow dynamics. We perform a study of the shape of a CNS vs. system parameters. Many of these resulting shapes

correspond to polygonized scrolls; we observe this for a range of parameter values.

We also learn about the effects of parameters on the model and the model limitations.

We then formulate a two-dimensional atomistic model of a CNS with modifiable parameters. This model is similar to the one-dimensional model, but requires fewer assumptions. We intend to consider two types of initial conditions: a sheet-like configuration and a scroll-like configuration.

CHAPTER II

ONE DIMENSIONAL MODEL

2.1 Modeling Assumptions

We assume that a CNS can be described by a chain of atoms in the plane. We can think of this chain as describing a typical cross-section of a CNS. The atoms in the chain are connected by links, where each link represents a strong, inextensible bond of length 1 (see Figure 2.1). This chain should reasonably approximate zigzag or armchair CNS. With each chain, we associate an energy that consists of two contributions: nonadjacent atoms interact via VDW forces that tend to keep these atoms at a certain equilibrium distance σ , and the bending forces between adjacent interatomic bonds that tend to keep these bonds aligned. A parameter β controls the size of the bending energy. We also consider the bonds between each atom and its immediate neighbors to be essentially inextensible. Hence, this model does not include the contribution due to extensibility of the bonds.

Next, we must decide how to track the positions of atoms in the chain. One way to do this is to use the position vector of each atom in \mathbb{R}^2 ; however, because of the assumption that each bond between atoms is inextensible, this approach would require many constraints to be incorporated into the problem. Alternatively, we

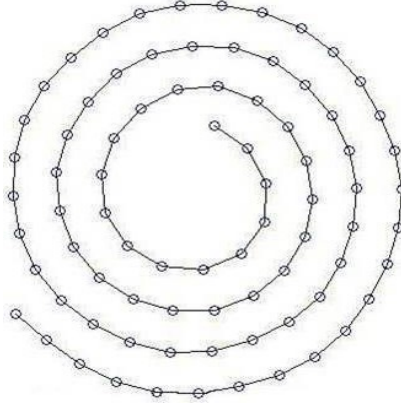


Figure 2.1: Planar chain of atoms approximating a cross-section of a CNS

observe that, because we know the length of each bond, it is sufficient to track the orientations of the bonds. Thus, we can represent the chain by a sequence of angles ψ_l , where ψ_l measures the angle between a bond joining atoms l and $l + 1$ and a prescribed vector \vec{v}_1 (see Figure 2.2).

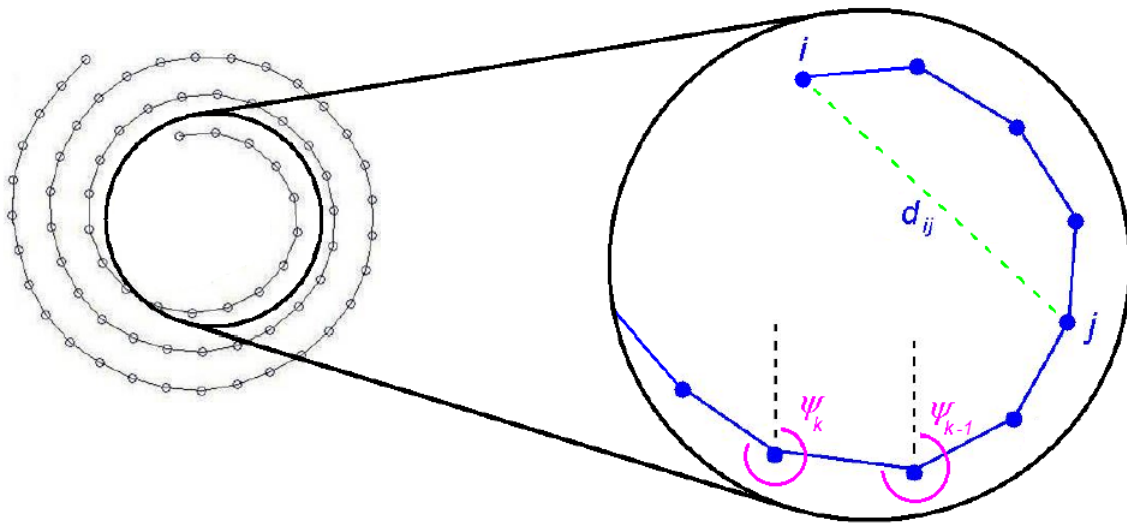


Figure 2.2: Geometry of the chain and VDW pairs.

As mentioned above, the energy of the chain has two components. The first is the energy due to VDW forces between atoms that are not covalently bonded together (see Figure 2.2). To represent this energy, we use the Lennard-Jones potential (1.1) so the VDW energy between atoms i and j is $E_{LJ}(d_{ij})$, where d_{ij} is the distance between atoms i and j . The second component is the bending energy, which accounts for the energy stored when adjacent bonds are not parallel. For this model, we assume that the bending energy is

$$E_B(\Delta\psi_k) = \tan^2\left(\frac{\Delta\psi_k}{2}\right), \quad (2.1)$$

where $\Delta\psi_k = \psi_k - \psi_{k-1}$. Note that $\pi - \Delta\psi_k$ is the angle between the two bonds that meet at the k^{th} atom (see Figure 2.2). The function E_B is even, periodic, vanishes at 0, and blows up to ∞ as $\Delta\psi_k$ approaches $\pm\pi$.

We assume that the total energy E is the sum of the energy due to van der Waals interactions and the bending energy. If we rescale by the strength of the van der Waals forces, we can rewrite E as

$$E = \sum_{i \neq j} E_{LJ}(d_{ij}) + \beta \sum_k E_B(\Delta\psi_k), \quad (2.2)$$

where β is the relative strength of the bending forces with respect to the van der Waals forces. Shortly, we show how to express d_{ij} in terms of the angles $\psi_i, \dots, \psi_{j-1}$. Hence the total energy E in (2.2) is a function of only these angles. Later we shall present the results of a parametric study in which we vary the values of σ and β .

2.2 Gradient Flow Dynamics

We assume that the chain evolves by gradient flow dynamics. Then, we have the following adaptation of system (2.3)

$$\frac{d\psi_l}{dt} = -\gamma \frac{\partial E}{\partial \psi_l}. \quad (2.3)$$

Combining equations (2.2) and (2.3), we get

$$\frac{d\psi_l}{dt} = -\gamma \left[\sum_{i \neq j} \frac{\partial E_{LJ}}{\partial \psi_l} + \beta \sum_k \frac{\partial E_B}{\partial \psi_l} \right]. \quad (2.4)$$

To compute the right-hand side of (2.4), recall (1.1) and (2.1). Via the chain rule,

$$\frac{\partial E_{LJ}}{\partial \psi_l} = \frac{dE_{LJ}}{d(d_{ij}^2)} \frac{\partial (d_{ij}^2)}{\partial \psi_l}, \quad (2.5)$$

and by (1.1)

$$\frac{dE_{LJ}}{d(d_{ij}^2)} = \frac{6}{\sigma^2} \left[\left(\frac{\sigma^2}{d_{ij}^2} \right)^4 - \left(\frac{\sigma^2}{d_{ij}^2} \right)^7 \right]. \quad (2.6)$$

To get the distance d_{ij} between atoms i and j , we consider the vector from atom i to atom j . Observe that this vector is the sum of the $|i - j|$ unit vectors that describe the orientations of the bonds between the adjacent atoms in the part of the chain from atom i to atom j (see Figure 2.2).

To express these vectors in components, we pick an orthonormal frame $\{\hat{v}_1, \hat{v}_2\}$.

The unit vector pointing from atom k to atom $k + 1$ can be written as $\cos(\psi_k)\hat{v}_1 + \sin(\psi_k)\hat{v}_2$. Then, we know the square of the distance between atoms i and j is

$$d_{ij}^2 = \left[\sum_{k=k_1(i,j)}^{k_2(i,j)} \cos(\psi_k) \right]^2 + \left[\sum_{k=k_1(i,j)}^{k_2(i,j)} \sin(\psi_k) \right]^2, \quad (2.7)$$

where $k_1(i, j) = \min\{i, j\}$ and $k_2(i, j) = \max\{i, j\} - 1$.

Returning to (2.5), we have

$$\frac{\partial (d_{ij}^2)}{\partial \psi_l} = \begin{cases} 2 \left[\cos(\psi_l) \sum_{k=k_1}^{k_2} \sin(\psi_k) - \sin(\psi_l) \sum_{k=k_1}^{k_2} \cos(\psi_k) \right] & \text{if } k_1 \leq l \leq k_2, \\ 0 & \text{otherwise.} \end{cases}$$

Thus,

$$\frac{\partial (d_{ij}^2)}{\partial \psi_l} = \begin{cases} 2 \sum_{k=k_1}^{k_2} \sin(\psi_k - \psi_l) & \text{if } k_1 \leq l \leq k_2, \\ 0 & \text{otherwise.} \end{cases} \quad (2.8)$$

Next, we can combine equations (2.5), (2.6), and (2.8) to get

$$\frac{\partial E_{LJ}}{\partial \psi_l} = \begin{cases} \frac{12}{\sigma^2} \left[\left(\frac{\sigma^2}{d_{ij}^2} \right)^4 - \left(\frac{\sigma^2}{d_{ij}^2} \right)^7 \right] \sum_{k=k_1}^{k_2} \sin(\psi_k - \psi_l) & \text{if } k_1 \leq l \leq k_2, \\ 0 & \text{otherwise.} \end{cases} \quad (2.9)$$

Next, we use equation (2.1) to compute

$$\frac{\partial E_B}{\partial \psi_l} = \begin{cases} \sec^2(\Delta\psi_k) \tan(\Delta\psi_k) & \text{if } k = l, \\ -\sec^2(\Delta\psi_k) \tan(\Delta\psi_k) & \text{if } k = l + 1, \\ 0 & \text{otherwise.} \end{cases}$$

Then, if we sum these derivatives for all k , we get the following equation

$$\frac{\partial}{\partial \psi_l} \sum_k E_B(\Delta\psi_k) = \sec^2(\Delta\psi_l) \tan(\Delta\psi_l) - \sec^2(\Delta\psi_{l+1}) \tan(\Delta\psi_{l+1}). \quad (2.10)$$

If we combine equations (2.4), (2.9), and (2.10) and rescale t by γ^{-1} , we get

$$\begin{aligned} \frac{d\psi_l}{dt} &= \frac{12}{\sigma^2} \sum_{i \neq j} \left(\left[\left(\frac{\sigma^2}{d_{ij}^2} \right)^7 - \left(\frac{\sigma^2}{d_{ij}^2} \right)^4 \right] \sum_{k=k_1(i,j)}^{k_2(i,j)} \sin(\psi_k - \psi_l) \right) \\ &+ \beta [\sec^2(\psi_{l+1} - \psi_l) \tan(\psi_{l+1} - \psi_l) + \sec^2(\psi_{l-1} - \psi_l) \tan(\psi_{l-1} - \psi_l)]. \end{aligned} \quad (2.11)$$

Thus, (2.11) gives us a system of ordinary differential equations for the evolution of the chain of atoms. To solve this system, we need only to set up an initial condition and pick the vector \hat{v}_1 .

2.3 Setting up the Initial Condition

Many geometries can be considered as initial conditions for our simulations. We are interested in studying spirals. Therefore, we pick initial conditions that are spiral shapes and that should not make VDW interactions too large. To construct these shapes, we start with a curve described in polar coordinates by

$$r(\theta) = \rho + \frac{\Delta r}{2\pi}\theta, \quad (2.12)$$

where ρ is the distance from the origin to the first point on the curve, Δr controls the distance between adjacent arms in the spiral, and $(r, \frac{\pi}{2} - \theta)$ is a point on the curve.

To construct the initial condition for the chain, we place n atoms on curve (2.12) (see Figure 2.3). Because we have inextensible bonds, adjacent atoms are connected by secant lines of length 1 along the curve. Thus, the length of the chain will be a function of the number of atoms n . If we convert the position of the first atom, $(\rho, \frac{\pi}{2})$, to Cartesian coordinates, we have the point $(0, \rho)$. Given a point (x_i, y_i) in the position of atom i , we can find $(\delta x_i, \delta y_i)$ such that $(x_{i+1}, y_{i+1}) = (x_i + \delta x_i, y_i + \delta y_i)$.

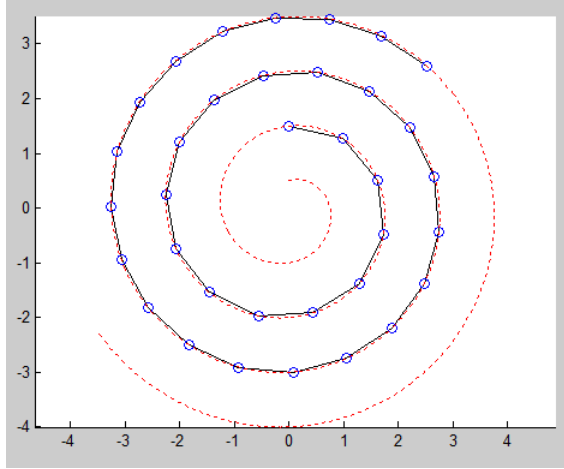


Figure 2.3: Spiral geometry for initial condition

Then we have the following system

$$\left\{ \begin{array}{l} \delta x_i = \left[\rho + \frac{\Delta r}{2\pi} \theta_{i+1} \right] \sin(\theta_{i+1}) - \left[\rho + \frac{\Delta r}{2\pi} \theta_i \right] \sin(\theta_i), \\ \delta y_i = \left[\rho + \frac{\Delta r}{2\pi} \theta_{i+1} \right] \cos(\theta_{i+1}) - \left[\rho + \frac{\Delta r}{2\pi} \theta_i \right] \cos(\theta_i), \\ \delta x_i^2 + \delta y_i^2 = 1, \\ \theta_1 = 0, \end{array} \right. \quad (2.13)$$

where $(r(\theta_i), \theta_i)$ is the coordinates of the i^{th} atom in polar form, and $i \in \{1, \dots, n-1\}$.

We change this into a system of equations with respect to θ only. We can rearrange the first two equations in system (2.13) as follows.

$$\left\{ \begin{array}{l} \delta x_i = \rho [\sin(\theta_{i+1}) - \sin(\theta_i)] - \frac{\Delta r}{2\pi} [\theta_{i+1} \sin(\theta_{i+1}) - \theta_i \sin(\theta_i)], \\ \delta y_i = \rho [\cos(\theta_{i+1}) - \cos(\theta_i)] - \frac{\Delta r}{2\pi} [\theta_{i+1} \cos(\theta_{i+1}) - \theta_i \cos(\theta_i)]. \end{array} \right.$$

Squaring the equation for δx_i , we get

$$\begin{aligned} \delta x_i^2 = & \rho^2 [\sin(\theta_i)^2 - 2 \sin(\theta_i) \sin(\theta_{i+1}) + \sin(\theta_{i+1})^2] \\ & + \frac{\rho(\Delta r)}{\pi} [\theta_i \sin(\theta_i)^2 - (\theta_i + \theta_{i+1}) \sin(\theta_i) \sin(\theta_{i+1}) + \theta_{i+1} \sin(\theta_{i+1})^2] \\ & + \frac{(\Delta r)^2}{4\pi^2} [\theta_i^2 \sin(\theta_i)^2 - 2\theta_i \theta_{i+1} \sin(\theta_i) \sin(\theta_{i+1}) + \theta_{i+1}^2 \sin(\theta_{i+1})^2]. \end{aligned} \quad (2.14)$$

Simplifying equation (2.14) (and its analog for δy_i^2) and adding the conditions from system (2.13) yields the following equation.

$$\begin{aligned} 1 = & 2\rho^2 [1 - \cos(\theta_{i+1} - \theta_i)] + \frac{\rho(\Delta r)}{\pi} [1 - \cos(\theta_{i+1} - \theta_i)] \\ & + \frac{(\Delta r)^2}{4\pi^2} [\theta_i^2 + \theta_{i+1}^2 - 2\theta_i \theta_{i+1} \cos(\theta_{i+1} - \theta_i)]. \end{aligned} \quad (2.15)$$

Finally, we can simplify (2.15) to get

$$\left\{ \begin{aligned} 1 = & \left(2\rho^2 + \frac{\rho(\Delta r)}{\pi} + \frac{\theta_i \theta_{i+1} (\Delta r)^2}{2\pi^2} \right) [1 - \cos(\theta_{i+1} - \theta_i)] \\ & + \frac{(\Delta r)^2}{4\pi^2} [\theta_{i+1} - \theta_i]^2, \\ & \theta_1 = 0. \end{aligned} \right. \quad (2.16)$$

System (2.16) can be quickly solved with a numerical root-finding method. Solving this system will yield a sequence of points in \mathbb{R}^2 given by their polar coordinates; however, for our model, we need a sequence of angles, $\{\psi_k\}$. If we convert the points to Cartesian coordinates as done above and pick the vector from the origin to the first atom as $\rho \hat{v}_1$, we can transform our sequence of points into a sequence of angles

$$\left\{ \begin{aligned} \sin(\psi_k) &= y_{k+1} - y_k, \\ \cos(\psi_k) &= x_{k+1} - x_k. \end{aligned} \right.$$

where $0 \leq \psi_k < 2\pi$. Thus, we are prepared to run our model.

CHAPTER III

ONE DIMENSIONAL RESULTS

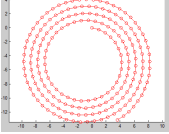
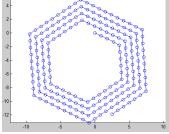
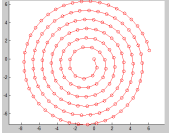
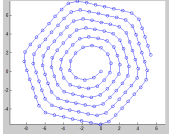
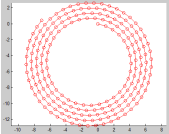
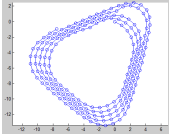
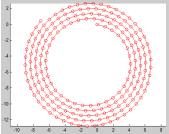
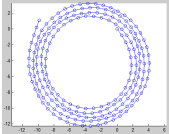
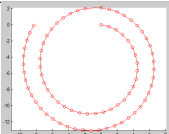
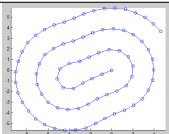
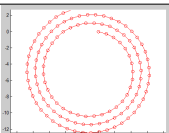
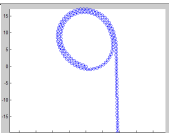
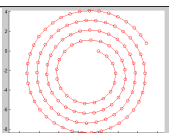
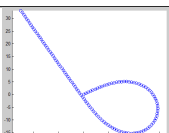
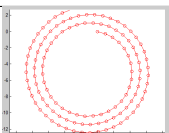
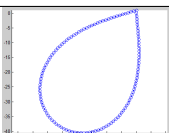
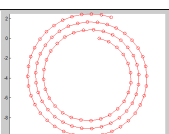
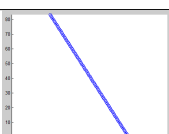
3.1 Types of Shapes

In this chapter, we present results for simulations run for various combinations of parameters. Table 3.1 shows typical examples of the qualitatively different shapes we observe in our simulations when we initialize the system with $\Delta r = \sigma$. From these, we were able to make some observations regarding the behavior of the system. We see a prominent dependence upon β , the relative strength of the bending forces to the VDW forces. We also see a dependence upon σ , the equilibrium distance of the van der Waals forces. We find some interesting results about the limitations of our model. Finally, we briefly investigate what happens if we do not have $\Delta r = \sigma$.

3.2 Varying β for $\sigma = 1$

In this section, we present results in which we fix n , r , Δr , and σ while varying β . The bending forces “want” the CNS to be straight and penalize deviations from this state. Therefore, it is reasonable to expect that as $\beta \rightarrow \infty$, the final configuration of the system will tend to a straight line (corresponding to a flat graphene sheet). Conversely, as $\beta \rightarrow 0$, the system will tend towards a triangular lattice of atoms

Table 3.1: Typical shapes found.

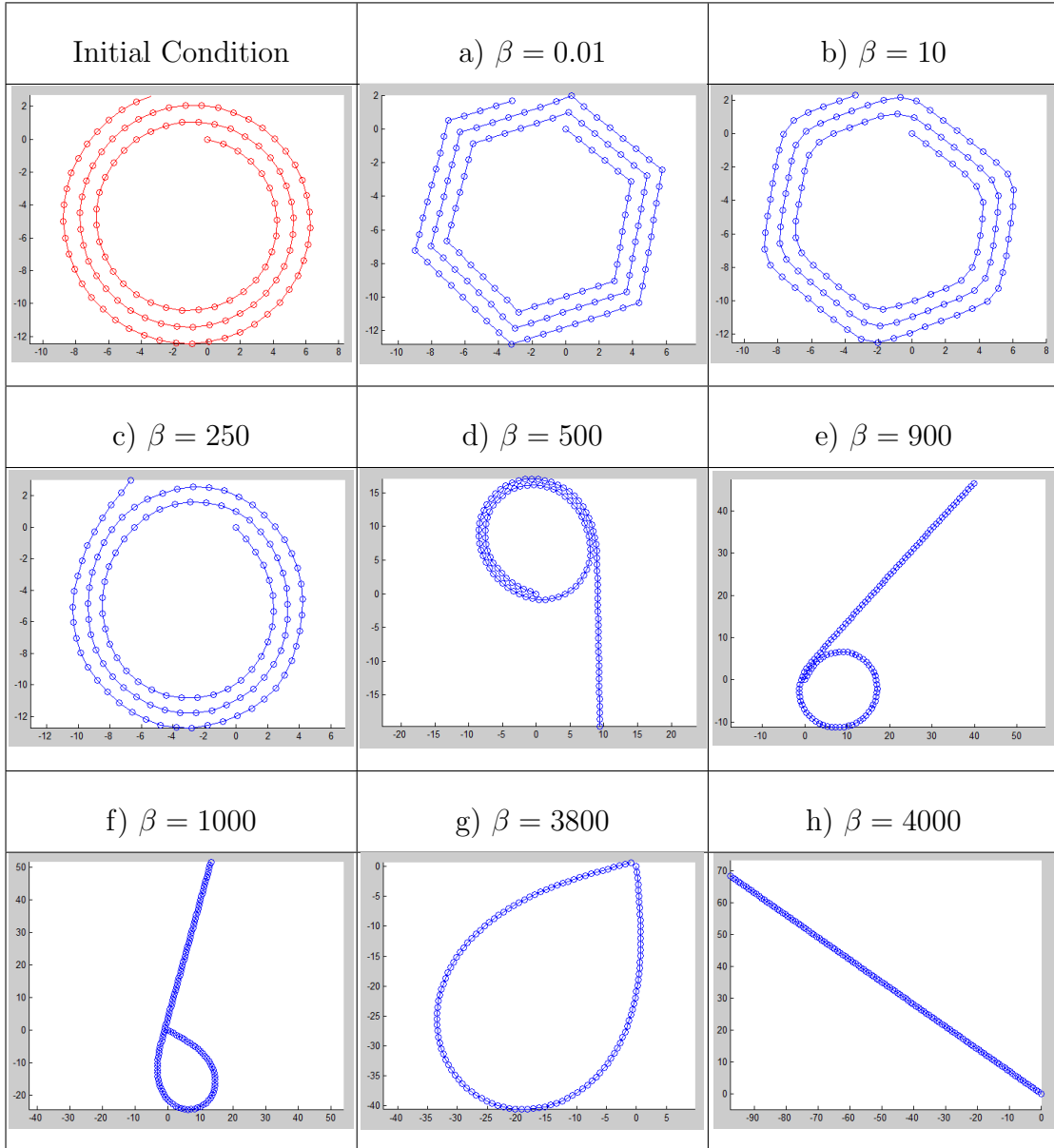
Shape	Initial Condition	Final Configuration
Sharp-Cornered Polygon		
Round-Cornered Polygon		
Polygon with Bubble		
Spiral-Like		
Paperclip		
Partially-Wound Spiral		
Golf Club		
Tennis Racquet		
Line		

favored by VDW interactions; atoms that are not bonded directly will “want” to be a distance σ apart. The behavior between these two cases is less obvious. We find that there is a predictable evolution in the simulation results with increasing β . Table 3.2 presents a series of qualitatively different plots that share all parameters, except β . Specifically, we set $n = 120$, $\rho = 5$, and $\Delta r = \sigma = 1$. These plots confirm expectations and shed some light on what happens for β in the intermediate stages.

Table 3.2a shows a simulation result for a very small value of β . The shape is characterized by a multi-layered polygonal shape with straight sides and sharp corners. Each side of this polygon corresponds to a lattice of atoms. Beginning to increase β preserves the general polygonal shape; however, the corners become less soft (see Table 3.2b). Because the bending energy function is convex and penalizes large angle differences between adjacent bonds, a lower energy can be achieved by spreading the corner out between a few links. Observe that the rounded corners look the same in each layer of the structure, giving the appearance of the same shape with successively larger straight lines separating the corners. Increasing β further and these corners will spread until a mostly-round shape is achieved (see Table 3.2c). At approximately this stage, the outer end of the spiral begins to straighten despite the pull of the van der Waals forces.

There is a large range of β for which the spiral will unwind until it reaches a local energy minimum (see Table 3.2d). As the spiral unwinds, the inner radius increases, making it progressively more difficult for the outer edge to delaminate. For large β within this range, a typical simulation leaves the still-wound portion mostly

Table 3.2: Results obtained by varying β .



one layer thick with a small region in which it is two layers thick (see Table 3.2e). Increasing β slightly from here causes a bifurcation in the solution where the inside edge no longer stays aligned with the mostly-straight portion and the innermost point slides outward until it reaches an equilibrium, forming a tear-drop shape at the end of a straight line, see Table 3.2f).

In this case, referred to as the “golf club” case earlier, two forces compete with each other. The collective force of the bending within the tear-drop shape competes with the VDW forces between the inside endpoint (and its neighbors) with the straight portion of the sheet. The VDW forces do not depend heavily upon which pair the inside point is held near. Conversely, the bending forces would be lowest if all of the angles were more gentle. So, the expansion of the drop-like shape depends on whether the cumulative bending force is enough to overcome the VDW forces holding the point in place. If the bending energy is large enough, the point will slide along the sheet until the bending forces no longer outweigh the VDW forces or the endpoint has reached the end of the sheet, (see Table 3.2g).

Increasing β from here causes the system to bifurcate once more. The endpoints detach from each other and bending energy governs the system, flattening the shape into a straight line, (See Table 3.2h). This is consistent with our expectation from earlier for what would happen as $\beta \rightarrow \infty$. Similar behavior is seen for different values of n and r .

3.3 Varying σ

VDW forces “prefer” lattices of atoms and bending forces “prefer” straight lines within a chain. Therefore, both VDW forces and bending forces “prefer” straight lines; however, a wound shape cannot be straight, though it can have straight parts. When we had a wound shape with straight components, we categorized them as “polygons” (see Table 3.1). These polygonal cross-sections can exist with different numbers of sides per revolution. We would like to find some sort of correlation between the number of sides and the parameters of the model. These configurations were stable because each side of a set of concentric polygons forms a mini-lattice, satisfying VDW forces, and is straight almost everywhere, satisfying bending forces.

The same type of polygonization occurs in CNT [8]. Next, we try to explain polygonization for MWNT and then apply similar ideas to CNS. This approach makes sense if we can assume that the CNS does not wind or unwind as it attains its equilibrium configuration.

We begin by trying to find the cause of polygonization in a DWNT. We can approximate the cross-section of a DWNT with closed, non-intersecting curves such that one is strictly contained within the other. We take these curves and map them to concentric circles (while preserving the curve lengths). We let r_1, r_2 denote the radii of the circles, with $r_1 < r_2$. See Figure 3.1. The difference in length is $2\pi(r_2 - r_1)$. These circles are analogous to the spiral in our model and can be considered to model a chain of bonds. Because we assume that inter-atomic bonds are inextensible, the

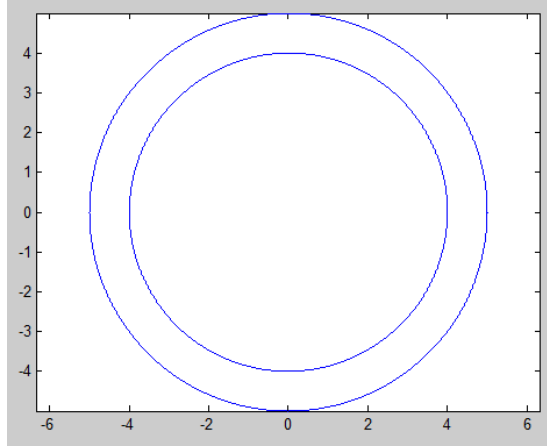


Figure 3.1: Double-walled tube cross-section

length of each curve must be proportional to the number of atoms on the curve. Thus, $2\pi(r_2 - r_1) = n_2 - n_1$ where n_2 is the number of atoms along the outer curve and n_1 is the number of atoms along the inner curve. Our “polygons” (Table 3.1) must have this same relationship.

Now, we assume that we have two concentric polygonal edges that are a constant distance apart and that the VDW energy is at its minimum. Suppose that our system of atoms along these two edges is symmetric about an axis perpendicular to the edges. This symmetry means we have one of two cases, either the perpendicular distance between the edges is approximately σ and a line perpendicular to the edges that passes through one atom will also pass through another (see Figure 3.2 (left)); or, the perpendicular distance between the edges is approximately $\frac{1}{2}\sqrt{4\sigma^2 - 1}$ and a line perpendicular to the edges that passes through an atom also bisects a bond (see Figure 3.2 (right)).

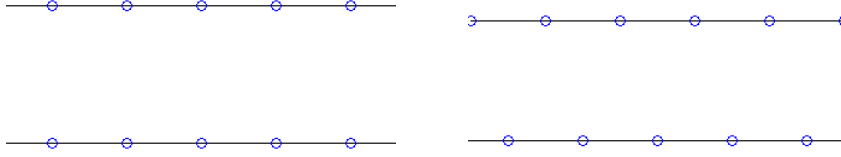


Figure 3.2: Symmetric pairs of parallel lines

To determine which case minimizes the VDW energy, we fix one line of atoms and plot the energy between a single atom and this line as a function of the position of the atom. This energy should be periodic with an axis parallel to the line with a period of 1, because that is the bond length. We will set the origin to be the middle of the line, orient the x -axis along the line, set $\sigma = 1$, and create a three-dimensional colored contour (see Figure 3.3). It is worth noting that the value of energy for this contour was capped at -1 for clarity. The colors correspond to the energy level at various points. The colors are ordered as in a rainbow with red indicating a large amount of energy and blue indicating a small amount of energy. We can see that the point $(0, 1)$ is a saddle point; this corresponds to case a (Figure 3.2 (left)). Similarly, the point $\left(-\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ is a minimum; this corresponds to case b (Figure 3.2 (right)). This behavior is periodic in x , as expected. Thus, we can expect to see the case b in practice.

Now we look at a polygonized DWNT. We assume that the global structure can be approximated by the two concentric circles mentioned above. Locally, we assume that the structure away from a corner can be approximated by the case b from above (Figure 3.2). As we investigate the polygonal corner, we will assume $\beta = 0$.

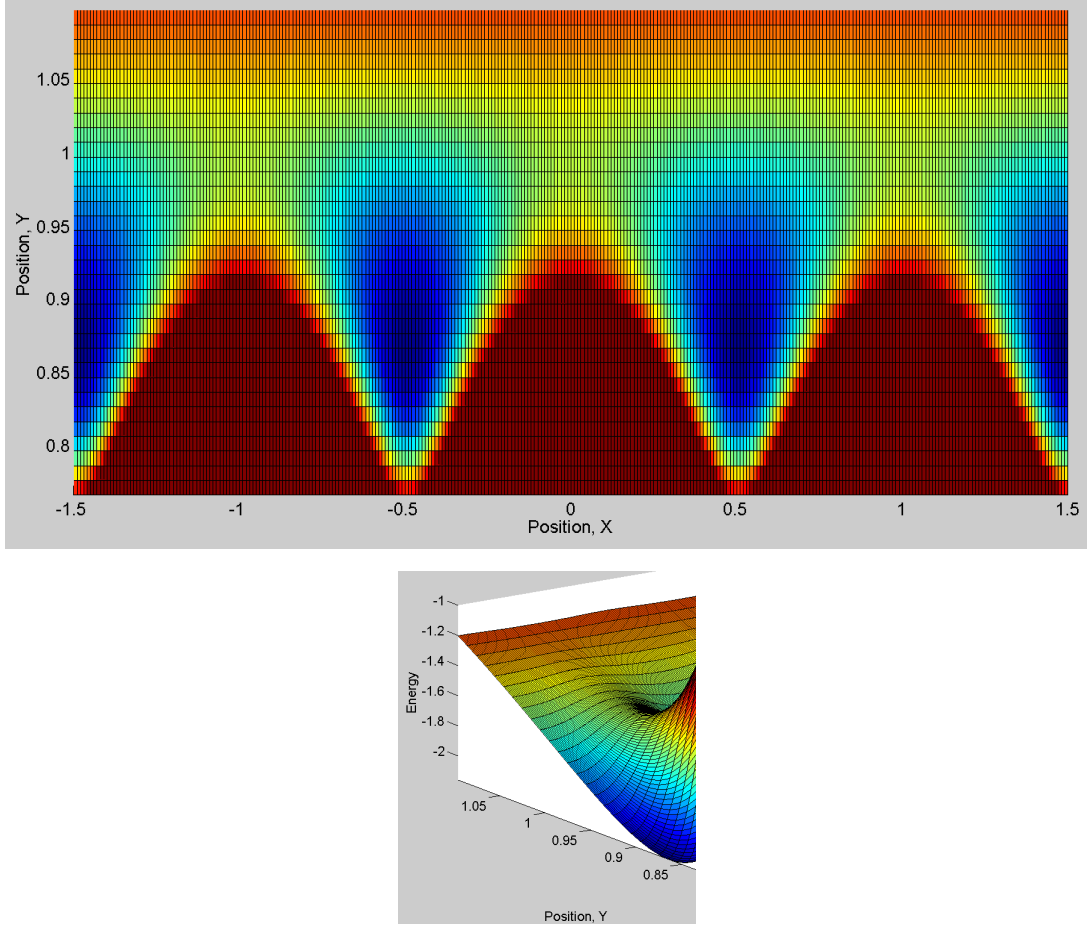


Figure 3.3: VDW energy from free atom near fixed edge

Thus, there is no energy penalty for large angles between normals to bonds and so we will assume that corners are sharp corners where one straight edge immediately switches to another at the junction of an atom. We will assume $\sigma = O(1)$. Thus, each corner of a polygon will have one inner corner-atom, shared between both edges, and one outer corner-atom, also shared between both edges. Also, we are consistent with Figure 3.3. Note that if we have this, we have the same corner-shape for the inside and outside corners, which we observed in the result shown in Table 3.2b and can

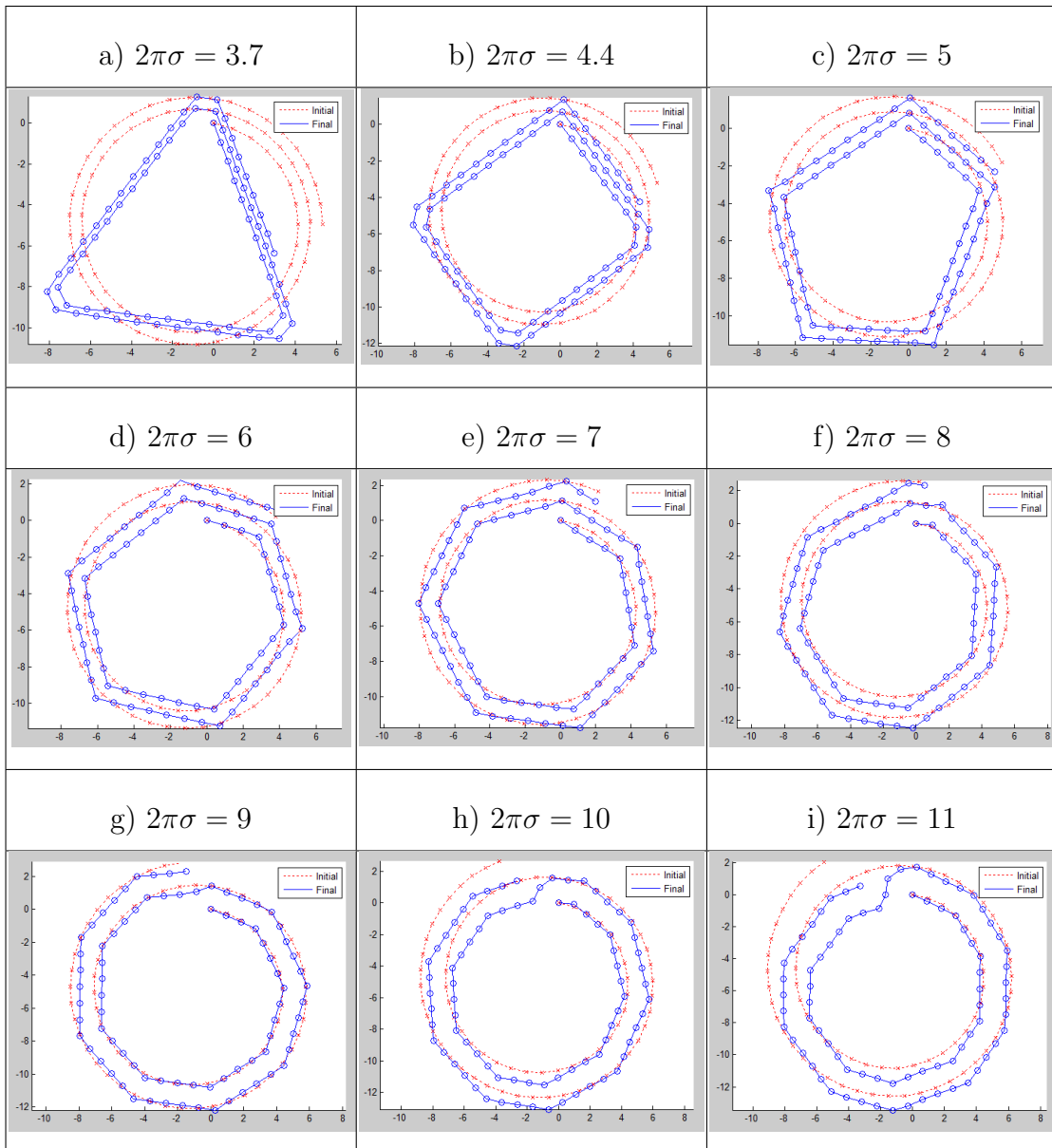
see in the result shown in Table 3.2a. With this assumption, for each pair of edges, we have one more atom in the outer edge than in the inner edge. Therefore, it is reasonable to expect that, for small β , polygonized cross-sections with these features will minimize the total energy and hence that the polygon has $n_2 - n_1$ sides.

The values n_2 and n_1 are well-defined in a for two nested circles; however, their meaning is not so clear in a spiral. Therefore, it may be more beneficial to think of them in terms of a distance. For circles, $n_2 - n_1$ is proportional to the distance between the circles. For spirals, the analogous quantity is the distance between arms of the spiral. Then, the number of corners per revolution should be approximately proportional to the VDW equilibrium distance, σ . With this in mind, we will vary the parameter σ and see how well the solutions match this expectation.

Table 3.3 shows nine plots of initial and final configurations from simulations in which $n = 80$, $\beta = 1$, $\rho = 5$, and $\Delta r = \sigma$, where σ is varied. We see a clear dependence on σ and the number of sides seems to increase approximately linearly with respect to σ for $\sigma = O(1)$. With these parameters we can see convex polygons ranging from triangles to decagons. We can observe that as σ increases, the winding number decreases. Additionally, we can observe that our assumption about the local behavior of the corner is only valid for values of σ within the approximate range of 0.8 to 1.4 for this set of parameters (Table 3.3 c-g); however, the global behavior does not seem to change outside of this range.

Now we investigate the extreme combinations of parameters. First, as suggested by Table 3.3 (a) and (b), $\sigma = \frac{4}{2\pi}$ does not yield a quadrilateral and $\sigma = \frac{3}{2\pi}$

Table 3.3: Results obtained by varying σ : $\sigma < 2$



does not yield a triangle. Here, the ratio of the radius, ρ , to the distance between spiral arms, Δr , is too large to be reasonably approximated by a circle. Then, the difference in spiral arm lengths can no longer be considered to be an integer times 2π . Here, $\sigma = \frac{4}{2\pi}$ is a transitional case between a a triangle and a quadrilateral (see Figure 3.4 a). If the n is increased to 200 and ρ is doubled, we get a quadrilateral (see Figure 3.4 b). We therefore see that this “issue” with the general assumption

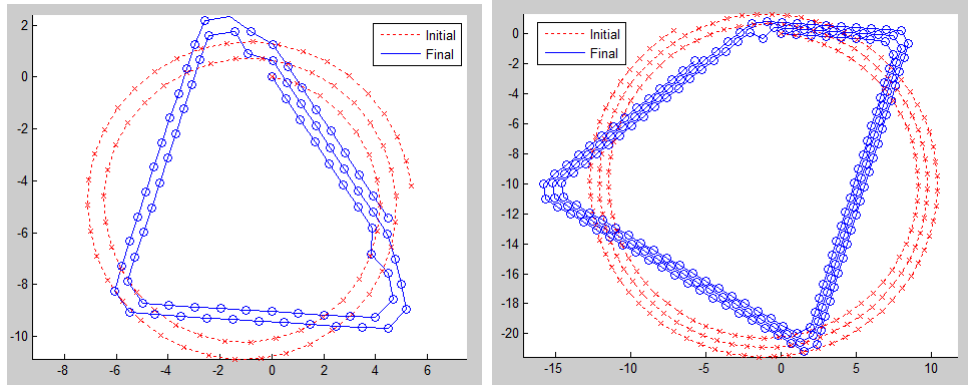


Figure 3.4: Transitional case for $2\pi\sigma = 4$ a) original solution b) doubling ρ and increasing n

only exists for small radii. However, increasing the radius will not yield a triangle for $\sigma = \frac{3}{2\pi}$; an entirely separate problem exists in that case.

A simulation result for $\sigma = \frac{3}{2\pi}$ is shown in Figure 3.6. In this result, the spiral walls intersect and atoms from different arms lie on the same curve. If we set $\sigma = \frac{3}{2\pi}$ and we fix an edge and plot the energy of the interaction of a single atom with a fixed line of atoms (see Figure 3.5), we see that the minimum energy is achieved if the atom lies directly between two atoms within that line. Physically, this would mean two layers of graphene would “fuse together,” and do so without affecting the system

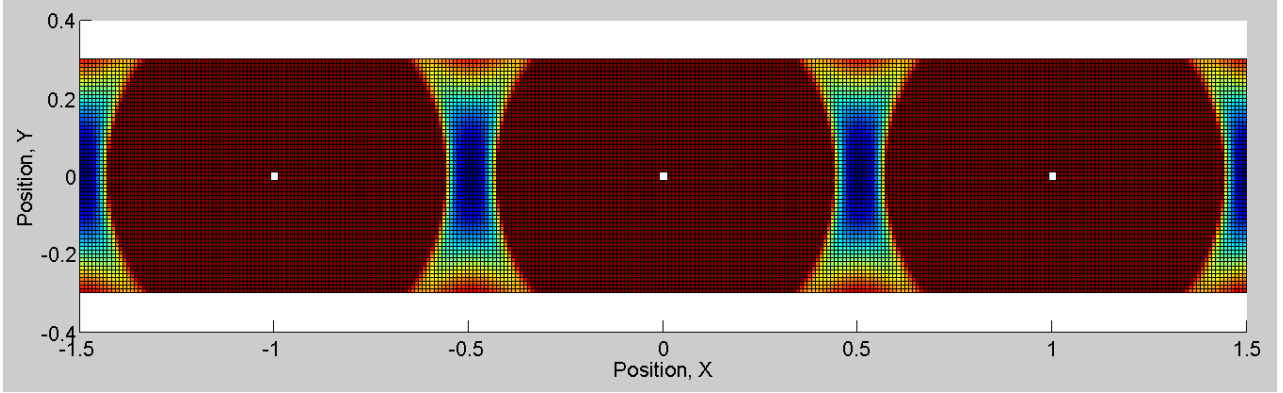


Figure 3.5: Degenerate case for $2\pi\sigma = 3$: energy from free atom with edge

of covalent bonds. This is obviously not realistic. This can happen for $\sigma \leq 0.5$, so we will restrict ourselves to $\sigma > 0.5$. It is worth noting that the covalent bond

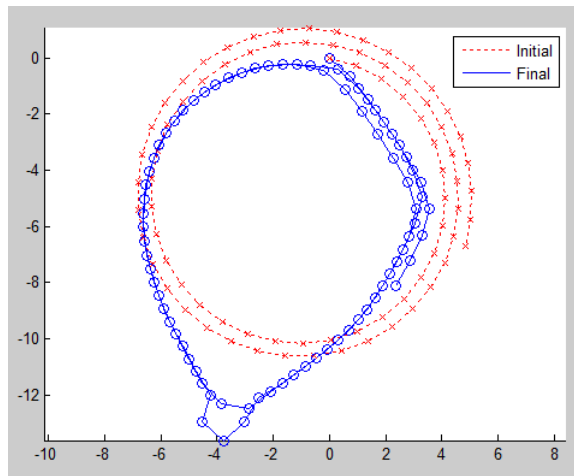


Figure 3.6: Degenerate case for $2\pi\sigma = 3$: simulation result

length between carbon atoms in a graphene sheet is 1.42 \AA and the VDW equilibrium distance is 3.4 \AA , meaning $\sigma_{\text{graphene}} \approx 2.4$ [27]. So, this modeling issue should not be a problem.

We can also see that as σ increases, the final configurations seem to lose their convexity. From Table 3.3i, we see that the convexity is lost for $2\pi\sigma = 11$. In this case, the loss of convexity appears to come from the length of the chain being too short; however, this merely exaggerates a behavior that exists for much larger chains. Figure 3.7 shows the case for 250 atoms. Even after more than tripling the number

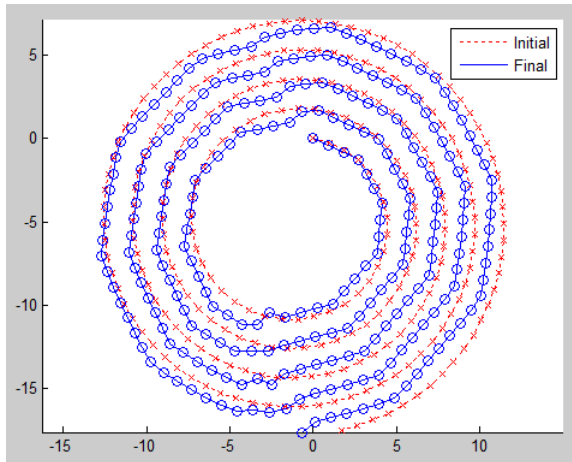


Figure 3.7: Simulation result for $2\pi\sigma = 11$, $n = 250$

of atoms, the configuration is non-convex in the same area. In this area, there is one more layer of atoms on one side of a corner than there is on the other, and so the atoms are attracted towards the “extra layer.”

That explains why this phenomenon occurs, but not why this occurs for larger values of σ rather than smaller values of σ . We offer two reasons. The first reason is, as σ increases, the valleys in the energy contour becomes more shallow, meaning deviating from the VDW minimum is not as undesirable as it was in previous cases. Figure 3.8 shows another energy contour, this time for $2\pi\sigma = 11$; there is a much

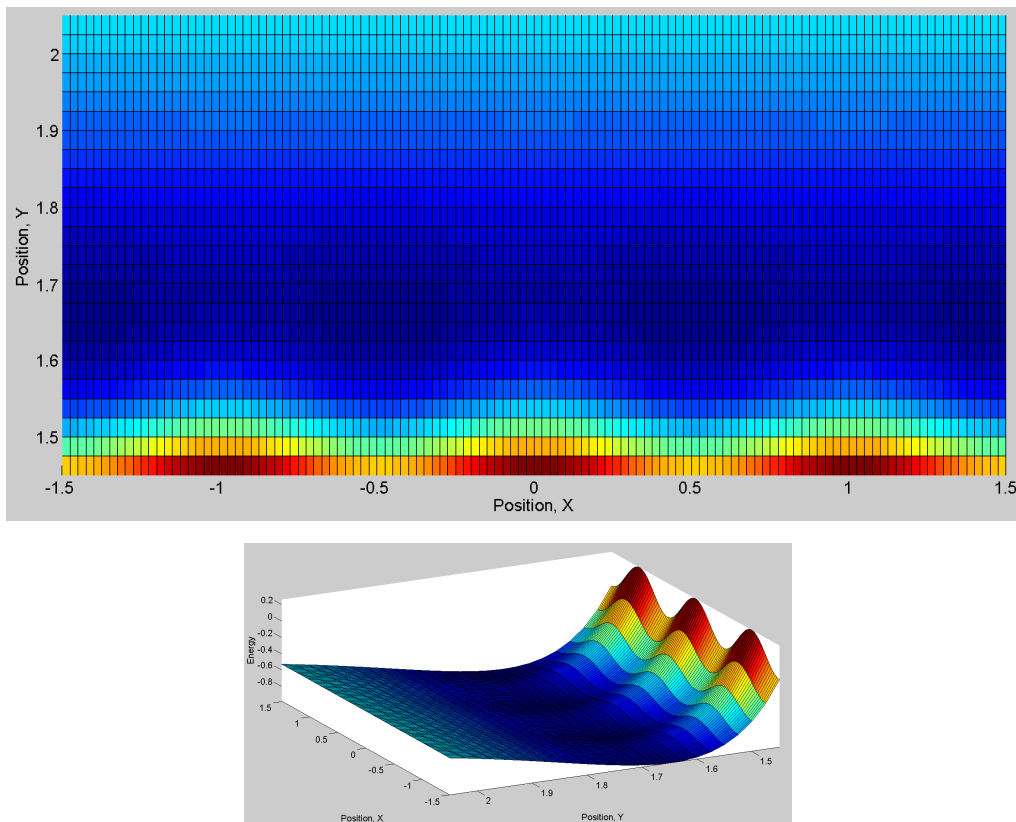
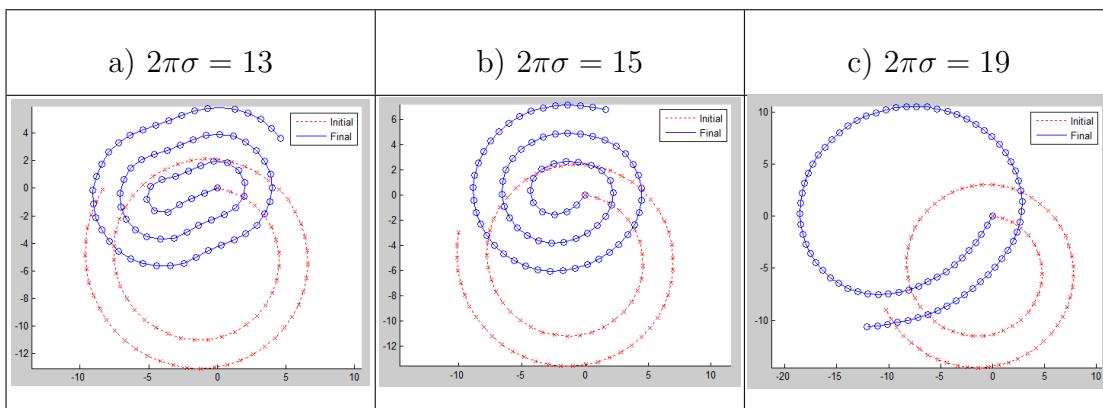


Figure 3.8: $2\pi\sigma = 11$: energy from free atom with edge

smaller penalty for deviating from the stable position here than there is in Figure 3.3. The second reason is, the VDW forces within an edge begin to counteract the VDW forces between edges.

One factor that we have not discussed yet is the effect of VDW forces between atoms that are two bonds apart. Recall that adjacent atoms on a chain are 1 unit apart. Hence, two atoms that are both bonded to a common third atom can be any distance from 0 to 2 from each other. Recall that there are $n_2 - n_1$ corners in a polygonized structure. Then, a regular polygon has angles of size $\frac{\pi}{n_2 - n_1}$. For

Table 3.4: Results obtained by varying σ : $\sigma > 2$



$\sigma < 2 \sin\left(\frac{\pi}{n_2 - n_1}\right)$, VDW forces between atoms that are two bonds apart counteracted bending forces, exaggerating the sharpness of the corners. For $\sigma > 2 \sin\left(\frac{\pi}{n_2 - n_1}\right)$, VDW forces between atoms that are two bonds apart reinforce the bending forces at the corners of a polygonized spiral. This effect caused a buffer to form against the forces from the “extra layer” on only one side of the corner for small σ , but not for large σ . As σ is increased further, the model acts similarly to how it would act if β was increased.

Table 3.4 is an extension of Table 3.3 for $\sigma > 2$. All other parameter values are the same. Here the VDW forces between the next nearest neighbors always act to straighten an arm. We have similar results to those seen in Table 3.2, but with one key difference. Here our energy contour is less steep, giving the edges more freedom to move parallel to one another. In all three cases, the arms start a distance $\Delta r = \sigma$ apart. We established earlier that this is further than the equilibrium distance

between arms. Case (b) shows that the final configuration of the spiral is more tightly wound than the initial configuration for $\sigma = \frac{15}{2\pi}$. Case (c) indicates that increasing σ causes the spiral to start unwinding, much like increasing β would. The remaining case, case (a), is something we have not seen yet. Case (a) begins by winding itself further, much like case (b), but in case (a), two opposite sides collapse inward due to the VDW pull across the center and form a paperclip-like shape. Transitioning to case (b) softens the angles in the semi-circles (the “paperclip” ends), and can also be interpreted as an analog of increasing β . Increasing σ further causes the spiral to entirely unwind into a straight line.

3.4 Setting $\Delta r > \sigma$

In all simulations discussed thus far, Δr was set equal to σ . Setting $\Delta r < \sigma$ gives the same result as $\Delta r = \sigma$ for small β . However, setting $\Delta r > \sigma$ leads to a new type of final configuration. See Figure 3.9. In this case, the inner layers of the spiral move towards the middle of the outer layer. This causes the final configuration to be divided into two sections. The first of these sections contains these inner layers. The second section only has the outer layer. This phenomenon can lead to a number of different final configurations.

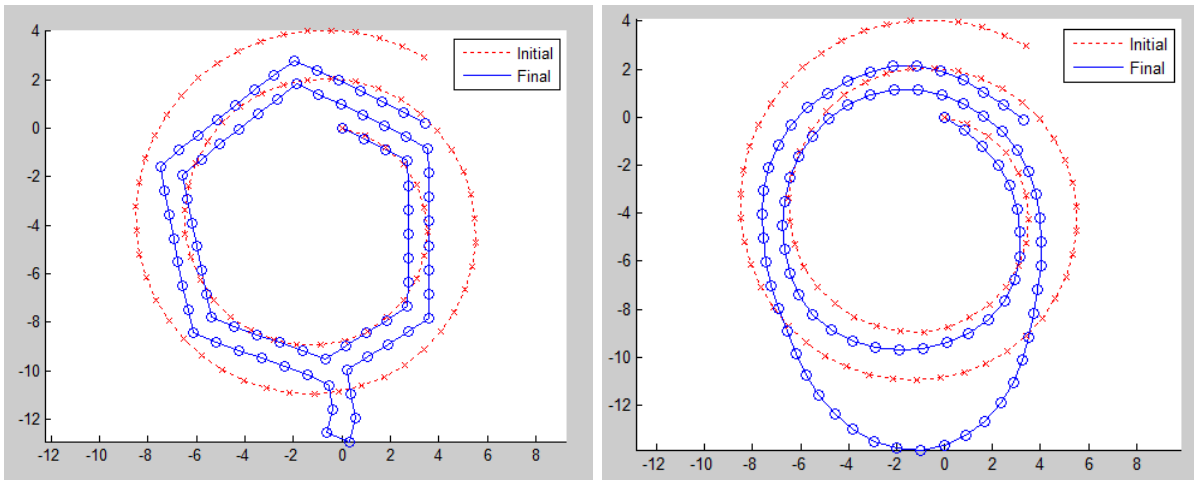


Figure 3.9: Simulation results for $\Delta r = 2$, $\sigma = 1$, a) $\beta = 1$, b) $\beta = 100$

CHAPTER IV

TWO DIMENSIONAL MODEL

4.1 Modeling Assumptions

We are interested in relaxing the assumptions of our one-dimensional model. To do this, we formulate a two-dimensional model. We assume that a CNS can be described by a two-dimensional lattice of atoms with each atom connected to its three nearest neighbors. In a flat, undeformed state, the atoms in the lattice form hexagons. The connection between each of pair of adjacent atoms models covalent bonding. We further assume that there is a trackable “center point” within each of these hexagons. In a flat, undeformed state, the atoms and center points form a triangular lattice (see Figure 4.1).

With this lattice, we associate an energy that consists of three contributions. Nonadjacent vertices interact via VDW forces that tend to keep these atoms at a certain equilibrium distance σ . The covalent bond between adjacent atoms is modeled by an extensional spring that tends to keep these atoms at a certain distance, l ; and, bending forces tend to keep adjacent triangles planar (see Figure 4.2).

We track the position of each atom in \mathbb{R}^3 . As mentioned above, three types of energies are included in the model. The first type is the VDW energy between pairs

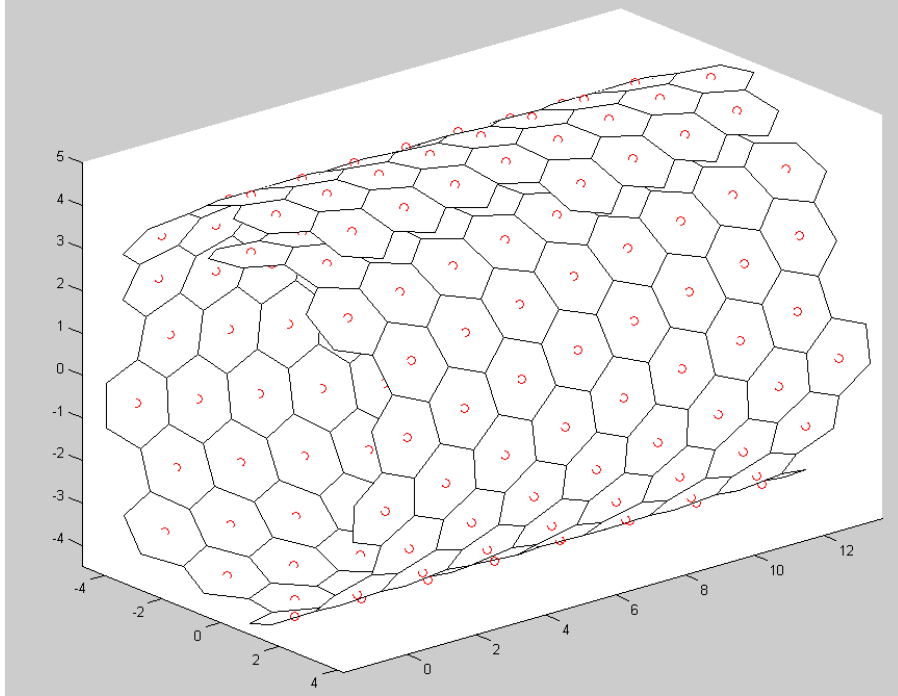


Figure 4.1: Approximated CNS

of bonds that are not directly bonded together. Note that we do not consider the center points when calculating the VDW forces. To represent this energy, we use the Lennard-Jones potential, (see Equation (1.1)). The second contribution to the energy is due to stretching and it increases whenever a bond extends or contracts from its natural length (see Figure 4.3). We assume that the stretching energy between atoms

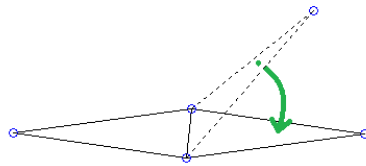


Figure 4.2: Bending force

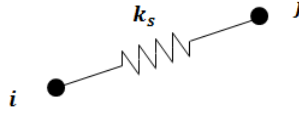


Figure 4.3: Model bond

i and j can be expressed by

$$E_S(l, k_s, \vec{r}_i, \vec{r}_j) = \frac{1}{2}k_s (|\vec{r}_i - \vec{r}_j| - l)^2, \quad (4.1)$$

where \vec{r}_i and \vec{r}_j are the positions of the atoms, k_s is a parameter defining the strength of the spring force relative to the VDW forces, and l is the natural length of the spring. For this type of energy, each center point acts as an atom, bonded to each vertex of its hexagon. The third type of energy is the “penalty” energy stored when adjacent triangles bend about a bond (see Figure 4.4). For this model, we will assume

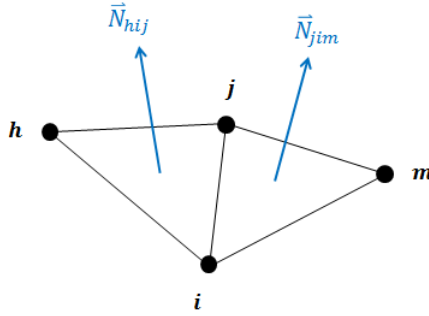


Figure 4.4: Adjacent normals

that the bending energy between two triangles can be expressed as

$$E_B(k_b, \vec{r}_i, \vec{r}_j, \vec{r}_h, \vec{r}_m) = k_b \left(\frac{3}{4} - \vec{N}_{hij} \cdot \vec{N}_{jim} \right), \quad (4.2)$$

where $\vec{N}_{123} = (\vec{r}_1 - \vec{r}_2) \times (\vec{r}_3 - \vec{r}_2)$ is a vector normal to the triangle with atoms 1, 2, and 3, and k_b is a parameter defining the strength of the bending force relative to the VDW forces. The function E_B is even and periodic with respect to the angle between normals; it is 0 when the normals are parallel; and, it has a maximum if the two normals point in opposite directions. We assume that the total energy of the system, E , is the sum of the energy due to VDW interactions, the spring energy, and the bending energy,

$$E = \sum E_{LJ} + \sum E_S + \sum E_B. \quad (4.3)$$

4.2 Gradient Flow Dynamics

We minimize the total energy (4.3) using to gradient flow dynamics. Then, we have the system

$$\frac{d\vec{r}_j^{(i)}}{dt} = -\gamma \frac{\partial E}{\partial \vec{r}_j^{(i)}}, \quad (4.4)$$

where $\vec{r}_j^{(i)}$ is the i^{th} component of the vector \vec{r}_j . Combining equations (4.3) and (4.4), we get

$$\frac{d\vec{r}_j^{(i)}}{dt} = -\gamma \left[\sum \frac{\partial E_{LJ}}{\partial \vec{r}_j^{(i)}} + \sum \frac{\partial E_S}{\partial \vec{r}_j^{(i)}} + \sum \frac{\partial E_B}{\partial \vec{r}_j^{(i)}} \right] \quad (4.5)$$

We need derivatives of the energies defined by (1.1), (4.1) , and (4.2).

We compute the derivative of (1.1) via the chain rule,

$$\frac{\partial E_{LJ}}{\partial \vec{r}_h^{(m)}} = \frac{dE_{LJ}}{d(d_{ij}^2)} \frac{\partial (d_{ij}^2)}{\partial \vec{r}_h^{(m)}}. \quad (4.6)$$

Equation (2.6) gives us the first factor on the right-hand side. We only need to determine the second factor. We can express d_{ij}^2 as $|\vec{r}_i - \vec{r}_j|^2$, so we have the equation

$$\frac{\partial (d_{ij}^2)}{\partial r_h^{(m)}} = \begin{cases} 2 (\vec{r}_i^{(m)} - \vec{r}_j^{(m)}) & \text{if } h = i, \\ 2 (\vec{r}_j^{(m)} - \vec{r}_i^{(m)}) & \text{if } h = j, \\ 0 & \text{otherwise.} \end{cases} \quad (4.7)$$

Now, we can rewrite (4.6) using (2.6) and (4.7) as

$$\frac{\partial E_{LJ}}{\partial r_h^{(m)}} = \begin{cases} \frac{12}{\sigma^2} (\vec{r}_i^{(m)} - \vec{r}_j^{(m)}) \left[\left(\frac{\sigma^2}{|\vec{r}_i - \vec{r}_j|^2} \right)^4 - \left(\frac{\sigma^2}{|\vec{r}_i - \vec{r}_j|^2} \right)^7 \right] & \text{if } h = i, \\ \frac{12}{\sigma^2} (\vec{r}_j^{(m)} - \vec{r}_i^{(m)}) \left[\left(\frac{\sigma^2}{|\vec{r}_i - \vec{r}_j|^2} \right)^4 - \left(\frac{\sigma^2}{|\vec{r}_i - \vec{r}_j|^2} \right)^7 \right] & \text{if } h = j, \\ 0 & \text{otherwise.} \end{cases} \quad (4.8)$$

We find the derivative of (4.1) with the chain rule and equation (4.7),

$$\frac{\partial E_s}{\partial r_h^{(m)}} = \begin{cases} k_s (\vec{r}_i^{(m)} - \vec{r}_j^{(m)}) \left[l - \frac{l}{|\vec{r}_i - \vec{r}_j|} \right] & \text{if } h = i, \\ k_s (\vec{r}_j^{(m)} - \vec{r}_i^{(m)}) \left[l - \frac{l}{|\vec{r}_i - \vec{r}_j|} \right] & \text{if } h = j, \\ 0 & \text{otherwise.} \end{cases} \quad (4.9)$$

Finally, we compute the derivative of (4.2). By repeatedly applying the product rule, we get the equation

$$\begin{aligned} \frac{\partial E_B}{\partial r_p^{(a)}} = & -k_b \left[\frac{\partial (\vec{r}_h - \vec{r}_i)}{\partial r_p^{(a)}} \times (\vec{r}_j - \vec{r}_i) + (\vec{r}_h - \vec{r}_i) \times \frac{\partial (\vec{r}_j - \vec{r}_i)}{\partial r_p^{(a)}} \right] \\ & \cdot \left[\frac{\partial (\vec{r}_j - \vec{r}_i)}{\partial r_p^{(a)}} \times (\vec{r}_m - \vec{r}_i) + (\vec{r}_j - \vec{r}_i) \times \frac{\partial (\vec{r}_m - \vec{r}_i)}{\partial r_p^{(a)}} \right] \end{aligned} \quad (4.10)$$

If we name our basis vectors \hat{e}_1 , \hat{e}_2 , and \hat{e}_3 , then we have the relationship

$$\frac{\partial \vec{r}_p}{\partial \vec{r}_p^{(q)}} = \hat{e}_q.$$

We can use this relationship to rewrite equation (4.10) as

$$\begin{aligned} \frac{\partial E_B}{\partial \vec{r}_i^{(q)}} &= -k_b [\hat{e}_q \times (\vec{r}_i - \vec{r}_j) + (\vec{r}_i - \vec{r}_h) \times \hat{e}_q] \cdot [\hat{e}_q \times (\vec{r}_i - \vec{r}_m) + (\vec{r}_i - \vec{r}_j) \times \hat{e}_q] \\ \frac{\partial E_B}{\partial \vec{r}_j^{(q)}} &= -k_b [(\vec{r}_h - \vec{r}_i) \times \hat{e}_q] \cdot [\hat{e}_q \times (\vec{r}_m - \vec{r}_i)] \\ \frac{\partial E_B}{\partial \vec{r}_h^{(q)}} &= -k_b [\hat{e}_q \times (\vec{r}_j - \vec{r}_i)] \\ \frac{\partial E_B}{\partial \vec{r}_m^{(q)}} &= -k_b [(\vec{r}_j - \vec{r}_i) \times \hat{e}_q] \end{aligned}$$

Simplifying this, we get

$$\begin{aligned} \frac{\partial E_B}{\partial \vec{r}_i^{(q)}} &= -k_b \hat{e}_q \times [(\vec{r}_h - \vec{r}_j) \cdot (\vec{r}_j - \vec{r}_m)] \\ \frac{\partial E_B}{\partial \vec{r}_j^{(q)}} &= -k_b \hat{e}_q \times [(\vec{r}_m - \vec{r}_i) \cdot (\vec{r}_i - \vec{r}_h)] \\ \frac{\partial E_B}{\partial \vec{r}_h^{(q)}} &= -k_b \hat{e}_q \times (\vec{r}_j - \vec{r}_i) \\ \frac{\partial E_B}{\partial \vec{r}_m^{(q)}} &= -k_b \hat{e}_q \times (\vec{r}_i - \vec{r}_j) \end{aligned} \tag{4.11}$$

If we combine equations (4.5), (4.8), (4.9), and (4.11) and rescale t by γ^{-1} , we get an equation for the evolution of the lattice through time. Then, we need only set up an initial condition to run our model.

4.3 Setting up the Initial Condition

Before we can set up an initial configuration of the lattice, we must first create the lattice itself. For simplicity, we create the lattice in a plane and map it to an “initial configuration” later. We begin by triangulating a certain region within this

plane with equilateral triangles. We then associate clusters of triangles as hexagons (see Figure 4.5). We then select a rectangle as an approximate boundary for the

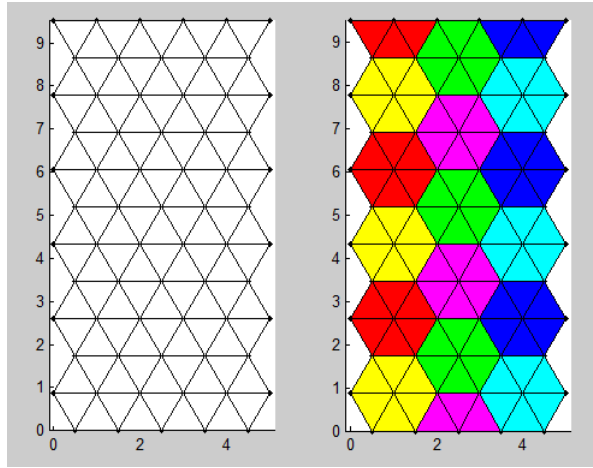


Figure 4.5: Lattice initialization

prospective graphene sheet. We then throw out any hexagons that are not within the rectangle or intersecting the rectangle, (see Figure 4.6).

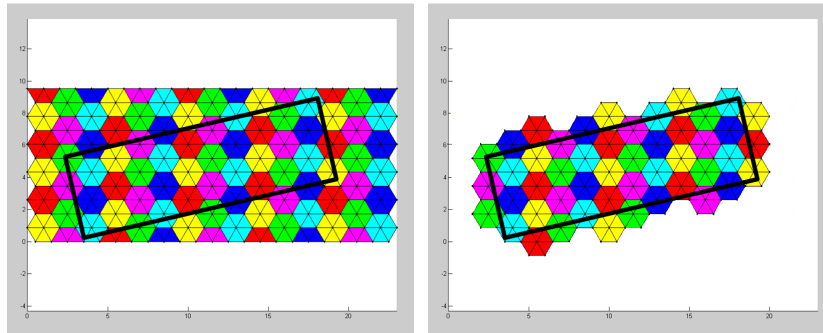


Figure 4.6: Isolating graphene sheet

To do this in practice, we must first define a rectangle. Then, the triangulation domain must be large enough to contain this rectangle with sufficient overlap to

ensure that all hexagons that would cross the border of the rectangle are still included in the triangulation domain. To uniquely define the rectangle, two values must be specified to determine the size of the rectangle and three values must be specified to determine its position and orientation. We define our rectangle with the following five parameters: L (length), W (width), x_0 (the x -position of the lower-left corner), y_0 (the y -position of the lower-left corner), and θ_0 (the rotation about the point (x_0, y_0)), (see Figure 4.7). The angle θ_0 determines the lattice orientation within the sheet and

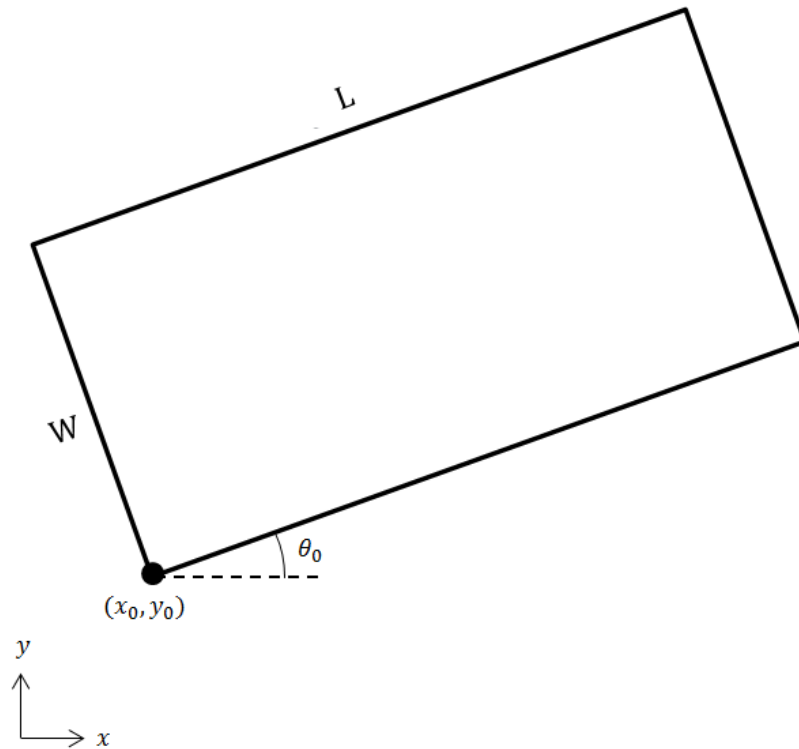


Figure 4.7: Definition of rectangle

therefore determines the sheet's edge geometry. This will correspond to the chirality of the CNS.

We then map this lattice to an “initial configuration” in \mathbb{R}^3 . We wrap this planar lattice into a CNS by translating and rotating the sheet and then bending it. First, we rigidly translate the sheet such that the point (x_0, y_0) moves to the origin. Next, we rigidly rotate the sheet by the angle $-\theta_0$. Finally, we are set up to bend the sheet about the y -axis.

To bend the CNS, we seek to determine how the method of placing secant lines on a 1-D curve can be modified to place secant planes on a 2-D surface. We look for a chain of atoms like we had in the 1-D case. First, we observe that there are three bond orientations within the unbent sheet. Figure 4.8 illustrates this through the use of three colors. We can look at each of these sets of bonds separately (see

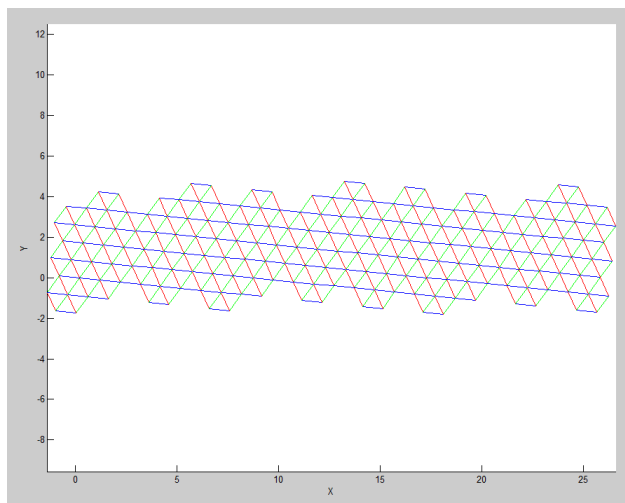


Figure 4.8: Graphene sheet bond orientations

Figure 4.9). The sheet will be bent about the y -axis; so, our chain should be bent about this axis also. Therefore, we should look for a chain that is oriented along the x -axis. If the chirality is “armchair,” we have this; otherwise, we do not.

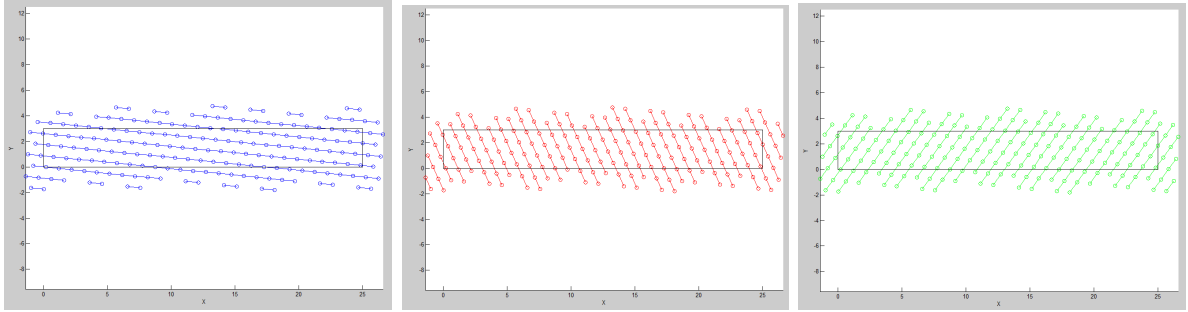


Figure 4.9: Graphene sheet bond orientations: individual orientations

Since we cannot find a chain with the correct orientation, we seek the chain most closely oriented to the x -axis. In the sheet from Figure 4.8, this is the blue set of bonds. We isolate a single chain of these blue atoms and we project this chain onto the x -axis. The projected chain can now be bent similarly to the chain in the 1-D model. We define a spiral in the same way and find secant lines to the curve similarly. Only, this time we create bonds the length of the bond projections.

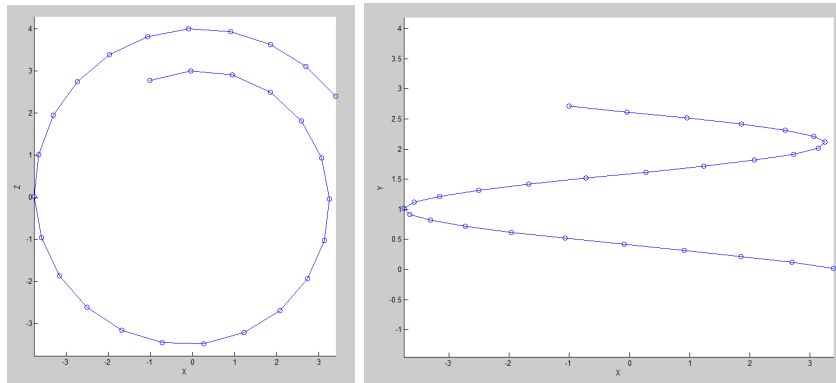


Figure 4.10: 3D chain

yields a bent chain in the xz -plane (see Figure 4.10 (left)). If we perform the same transformation to the original bonds instead of the projected ones, preserving the

position along the y-axis, we get a chain of atoms in 3-D space where each bond has length 1 (see Figure 4.10 (right)).

To expand the transformation of this chain to the entire sheet, we assume that a chain sits along the bottom line of the rectangle defined above with an atom at the lower-left corner. We transform this chain and interpolate all projected atoms onto the curve between the points on the assumed chain. The result is a bent sheet with a spiral-shaped cross section (see Figure 4.11).

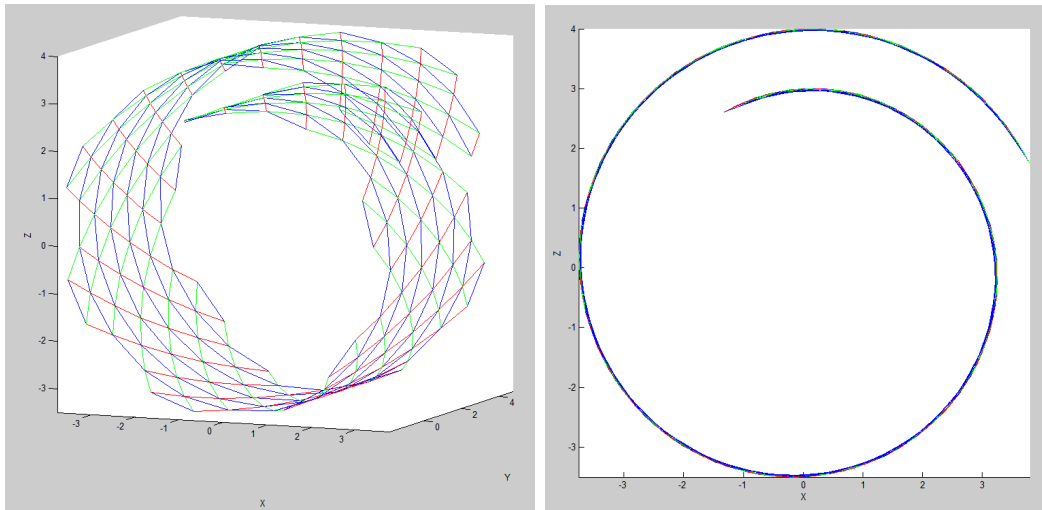


Figure 4.11: 2-D CNS showing bond orientations

Thus, the 2-D model has been set up. This model can now be run to more accurately determine equilibrium configurations for CNS. Additionally, the behavior of a graphene sheet can be investigated by omitting the transformation to a CNS. This model should facilitate a wide variety of results; however, its application is beyond the scope of this thesis.

CHAPTER V

CONCLUSIONS

We have modeled a CNS by a one-dimensional chain of atoms. The model includes VDW forces and bending forces and treats the bonds between adjacent atoms as inextensible. We found equilibrium configurations that locally minimize the total energy by gradient flow dynamics. We assumed an initial configuration of an Archimedean spiral. Finally, we evolved several of these configurations through time while varying the strength of the bending forces and the equilibrium distance of the VDW forces.

By varying the strength of bending forces, we found a variety of final configurations ranging from polygonized to spiral-shaped, to straight. The polygonized shapes correlate with configurations that have been seen experimentally. Similarly, the straight shape corresponds to a graphene sheet. Thus, this model was able to replicate some equilibrium configurations that have been seen experimentally. This could potentially lead to predictive modeling.

By varying the equilibrium distance of the VDW forces, we were able to change the number of sides of a polygon. We were able to see polygons ranging from triangles to decahedrons. This shows that our model can yield a wide variety of polygonal shapes. Overall, this model appears to find the general shapes that are discovered experimentally.

We have also modeled a CNS by a two-dimensional lattice of atoms. This model includes internal forces between bonded atoms, VDW forces, and bending forces. We modeled evolution through time by gradient flow dynamics. Lastly, we discussed issues that arise in creating a spiral initial configuration for the 2D model.

BIBLIOGRAPHY

- [1] C.-H. Kiang, M. Endo, P. M. Ajayan, G. Dresselhaus, and M. S. Dresselhaus, “Size effects in carbon nanotubes,” *Phys. Rev. Lett.*, vol. 81, pp. 1869–1872, Aug 1998.
- [2] F. Y. Wu and H. M. Cheng, “Structure and thermal expansion of multi-walled carbon nanotubes before and after high temperature treatment,” *Journal of Physics D: Applied Physics*, 2005.
- [3] X. Xie, L. Ju, X. Feng, Y. Sun, R. Zhou, K. Liu, S. Fan, Q. Li, and K. Jiang, “Controlled fabrication of high-quality carbon nanoscrolls from monolayer graphene,” *Nano Letters*, vol. 9, no. 7, pp. 2565 – 2570, 2009.
- [4] D. Graber and J. Batten, “Nano enhanced wholesale technologies.” <http://www.nano-enhanced-wholesale-technologies.com/faq/carbon-forms.htm>. Accessed April 4, 2014.
- [5] A. Ito and S. Okamoto, “Molecular dynamics analysis on effects of vacancies upon mechanical properties of graphene and graphite,” *Engineering Letters*, vol. 20, no. 3, pp. 271 – 278, 2012.
- [6] S. Ebrahimi, A. Montazeri, and H. Rafii-Tabar, “Molecular dynamics study of a new mechanism for ripple formation on graphene nanoribbons at very low temperatures based on h₂ physisorption,” *Solid State Communications*, vol. 159, pp. 84 – 87, 2013.
- [7] J. M. Wernik and S. A. Meguid, “Atomistic-based continuum modeling of the nonlinear behavior of carbon nanotubes,” *Acta Mechanica*, vol. 212, no. 1/2, pp. 167 – 179, 2010.
- [8] D. Golovaty and S. Talbott, “Continuum model of polygonization of carbon nanotubes,” *Phys. Rev. B*, vol. 77, p. 081406, Feb 2008.

- [9] M. Endo, T. Hayashi, H. Muramatsu, Y.-A. Kim, H. Terrones, M. Terrones, and M. S. Dresselhaus, “Coalescence of double-walled carbon nanotubes: Formation of novel carbon bicables,” *Nano Letters*, vol. 4, no. 8, pp. 1451 – 1454, 2004.
- [10] B. V. C. Martins and D. S. Galvao, “Curved graphene nanoribbons: structure and dynamics of carbon nanobelts,” *Nanotechnology*, vol. 21, no. 7, p. 075710, 2010.
- [11] X. Shi, N. Pugno, and H. Gao, “Constitutive behavior of pressurized carbon nanoscrolls,” *International Journal of Fracture*, vol. 171, no. 2, pp. 163 – 168, 2011.
- [12] Braga, F. S., Coluci, R. V., Legoas, B. S., Giro, R., Galvao, S. D., Baughman, and H. R., “Structure and dynamics of carbon nanoscrolls,” *Nano Letters*, vol. 4, no. 5, pp. 881 – 884, 2004.
- [13] E. Perim, R. Paupitz, and D. S. Galvo, “Controlled route to the fabrication of carbon and boron nitride nanoscrolls: A molecular dynamics investigation,” *Journal of Applied Physics*, vol. 113, no. 5, p. 054306, 2013.
- [14] H. Y. Song, S. F. Geng, M. R. An, and X. W. Zha, “Atomic simulation of the formation and mechanical behavior of carbon nanoscrolls,” *Journal of Applied Physics*, vol. 113, no. 16, pp. 164305 – 164305–6, 2013.
- [15] Z. Zhang and T. Li, “Carbon nanotube initiated formation of carbon nanoscrolls,” *Applied Physics Letters*, vol. 97, no. 8, p. 081909, 2010.
- [16] Z. Zhang, Y. Huang, and T. Li, “Buckling instability of carbon nanoscrolls,” *Journal of Applied Physics*, vol. 112, no. 6, p. 063515, 2012.
- [17] D. Mantzalis, N. Asproulis, and D. Drikakis, “Enhanced carbon dioxide adsorption through carbon nanoscrolls,” *Physical Review E: Statistical, Nonlinear & Soft Matter Physics*, vol. 84, no. 6-2, pp. 1 – 8, 2011.
- [18] X. Shi, Q. Yin, N. M. Pugno, and H. Gao, “Tunable mechanical behavior of carbon nanoscroll crystals under uniaxial lateral compression,” *Journal of Applied Mechanics*, vol. 81, no. 2, pp. 1 – 6, 2014.
- [19] Y. Huang, “Molecular Mass Transportation Via Carbon Nanoscrolls,” *Journal of Applied Mechanics*, vol. 80, p. 041038, May 2013.

- [20] Y. Cheng, X. Shi, N. M. Pugno, and H. Gao, “Substrate-supported carbon nanoscroll oscillator,” *Physica E Low-Dimensional Systems and Nanostructures*, vol. 44, pp. 955–959, Mar. 2012.
- [21] X. Shi, Y. Cheng, N. M. Pugno, and H. Gao, “A translational nanoactuator based on carbon nanoscrolls on substrates,” *Applied Physics Letters*, vol. 96, no. 5, p. 053115, 2010.
- [22] V. Yannopapas, M. Tzavala, and L. Tsetseris, “Arrays of carbon nanoscrolls as deep subwavelength magnetic metamaterials,” *Phys. Rev. B*, vol. 88, p. 155413, Oct 2013.
- [23] A. L. Chuvilin, V. L. Kuznetsov, and A. N. Obraztsov, “Chiral carbon nanoscrolls with a polygonal cross-section,” *Carbon*, vol. 47, no. 13, pp. 3099 – 3105, 2009.
- [24] C. W. Fan, Y. Y. Liu, and C. Hwu, “Finite element simulation for estimating the mechanical properties of multi-walled carbon nanotubes,” *Applied Physics A: Materials Science & Processing*, vol. 95, no. 3, pp. 819 – 831, 2009.
- [25] X. Shi, N. M. Pugno, and H. Gao, “Tunable core size of carbon nanoscrolls,” *Journal of Computational and Theoretical Nanoscience*, 2010.
- [26] X. Oyharcabal and T. Frisch, “Peeling off an elastica from a smooth attractive substrate,” *Phys. Rev. E*, vol. 71, p. 036611, Mar 2005.
- [27] S. Sarrami-Foroushani and M. Azhari, “Nonlocal vibration and buckling analysis of single and multi-layered graphene sheets using finite strip method including van der waals effects,” *Physica E*, vol. 57, pp. 83 – 95, 2014.

APPENDIX

ONE DIMENSIONAL MODEL CODE

spiral.m

```
%% Mapping
%% Given (j, i) ~ ( tube index, atom index )
%% and an array of angles, y, then
%% y(i + sum(n(1:j-1))) is the angle for the atom
%% at (j, i)
%%%% Parameters
%% m := # of tubes
%% n(j) := ## of atoms on j-th tube
%% beta := bending coefficient
%% len := length between atoms on a tube
%% lambda := load on the tubes
%% epsi := Strength of Van der Waals interaction
%% sigma := Equillibrium distance of Van der Waals.
%% init := initial condition
%% delta(j) := offset of the j-th tube

m = 1;
n = 30;
beta = 10;
len = 1;
lambda = 0; mu = 0;
epsi = 1;
sigma = 1;
delta = 0:1:(m-1);
%%init = zeros(1,sum(n));

checkInterval = 1000;
nhbdUpdate = 500;
nhbdRadius = 6;

%%rng(404530);
%%for i = 2:n*m
```

```

%% init(i) = 0 + (pi/4-0).*rand(1,1);
%% init(i) = init(i-1) + i*pi/3;
%% init(i) = 0;
%%end

init = SpiralGeomInit(n + 1,2,1);

%% Parameter and Output Setup
%%{
if length(n) < m
    temp = n;
    temp_size = length(n);
    temp_end = n(length(n));
    n = zeros(1,m);
    n(1:temp_size) = temp;
    for ps_k = temp_size+1:m
        n(ps_k) = temp_end;
    end
elseif length(n) > m
    n = n(1:m);
end

N = sum(n);

state = struct('m',m,'n',n,'N',N,'beta',beta,'len',len, . +
..
    'lambda',lambda,'mu',mu,'epsi',epsi,'sigma',sigma, . +
..
    'delta',delta,'init',init);

imap = zeros(N,2);
for i = 1:N
    [ imap(i,1), imap(i,2) ] = getAtom(i,n);
end

E = spiralEnergy(init,state,imap,checkInterval,nhbdUpdate +
,nhbdRadius);

options = odeset('RelTol',1e-12,'AbsTol',1e-12);

%%}
%% Solver and Time Domain
state.tend = 10;

```

```
tspan = 0:1:state.tend;
startTime = cputime;
[ t, y ] = ode15s(@spiralForce,tspan,init,options,state,i +
map, ...
    checkInterval,nhbdUpdate,nhbdRadius);
endTime = cputime;
elapsedTime = endTime - startTime;

for j = 2:length(tspan)
    E = [E; spiralEnergy(y(j,:),state,imap,checkInterval, +
nhbdUpdate,nhbdRadius)];
end
state.time = elapsedTime;
```

SpiralGeomInit.m

```
function psi = SpiralGeomInit(n,r,dr)

% This is like Tube init geom but for spirals.
%
% This assumes that the radius linearly increases with re +
spect to the
% angle about the origin.
%
% The "r" term is the initial radius and the "dr" term is +
the change in
% radius of the spiral between layers.

% It determines the atoms' positions in polar coordinates +
and converts them
% to coordinates in terms of azimuths relative to previou +
s points.
%
% The first point of each spiral is assumed to be along t +
he same axis
% intersecting the origin.

Len = length(n); Max = max(n);
psi = zeros(Max-1,Len);

tol = 1e-8; max_it = 10000;

for i = 1:Len
    for j = 1:n(i)
        if j == 1
            R = r(i); DR = dr(i);
            theta_2 = 0; r2 = R;
            y1 = r2; x1 = 0;
            theta_1 = theta_2; r1 = r2;
        else
            a = theta_1; b = theta_1 + 3*pi/2;
            f = @(theta_2)...
                -1 + 2*R^2 * (1 - cos(theta_2 - theta_1)) +
                + ...
                R*DR/pi*(theta_1+theta_2) * (1-cos(theta_ +
                2-theta_1)) + ...
                (DR/(2*pi))^2 * ...
                (theta_1^2+theta_2^2-2*theta_1*theta_2*co +
```



```

        s(theta_2-theta_1));

theta_2 = secant(f,a,b,tol,max_it);
r2 = DR*(theta_2-theta_1)/(2*pi) + r1;
x2 = r2*sin(theta_2); y2 = r2*cos(theta_2);
dx = x2 - x1; dy = y2 - y1;
d = sqrt(dx^2+dy^2);
if 1-d > tol
    fprintf('Spiral #%.1lg, Bond #%.6lg is %4 +
        .4g too short.\n',i,j-1,1-d)
elseif d-1 > tol
    fprintf('Spiral #%.1lg, Bond #%.6lg is %4 +
        .4g too long.\n',i,j-1,d-1)
end
if dx > 0
    if dy > 0
        psi(j-1,i) = atan(dx/dy);
    elseif dy == 0
        psi(j-1,i) = pi/2;
    else
        psi(j-1,i) = pi + atan(dx/dy);
    end
elseif dx == 0
    if dy > 0
        psi(j-1,i) = 0;
    else
        psi(j-1,i) = pi;
    end
else
    if dy < 0
        psi(j-1,i) = pi + atan(dx/dy);
    elseif dy == 0
        psi(j-1,i) = 3*pi/2;
    else
        psi(j-1,i) = 2*pi + atan(dx/dy);
    end
end
x1 = x2; y1 = y2; theta_1 = theta_2; r1 = r2;
end
end
end
end

```

secant.m

```
function [p1,k]=secant(f,p0,p1,tol,max1)

for k=1:max1
    p2=p1-feval(f,p1)*(p1-p0)/(feval(f,p1)-feval(f,p0));
    abserr=abs(p2-p1);
    relerr=2*abserr/(abs(p2)+tol);
    p0=p1;
    p1=p2;
    y=feval(f,p1);
    if (abserr<tol)&(relerr<tol)&(abs(y)<tol),break,end
end
```

getAtom.m

```
function [ j, i ] = getAtom( x, n )
    temp_s = n(1);
    j = 1;
    for k = 2:length(n)
        if x <= temp_s
            temp_s = temp_s - n(k-1);
            break;
        end
        j = k;
        temp_s = temp_s + n(k);
    end
    if j == length(n)
        temp_s = temp_s - n(length(n));
    end
    i = x - temp_s;
end
```

getNghd.m

```
function F = genNghd( y, cs, sn, state, imap, radius )

    len = state.len;
    delta = state.delta;
    r2 = radius*radius;

    k = 1;
    for i = 1:length(y)
        for j = i+1:length(y)
            v = imap(i,1);
            u = imap(i,2);
            g = imap(j,1);
            h = imap(j,2);
            % Rule out the atom itself and it's two immediate neig +
hbors
            if v == g
                if u == h || u + 1 == h || u - 1 == h
                    continue;
                end
            end
            xps = ((delta(v) - delta(g))./len) + sn(i) - sn(j);
            yps = cs(i) - cs(j);
            d = xps.^2 + yps.^2;

            if d <= r2
                C(k,1) = {i};
                C(k,2) = {j}};
                k = k + 1;
            end
        end
    end
    F = cell2mat(C);
end
```

getPos.m

```
function [ x, y ] = getPos( in, state, imap )

    delta = state.delta;
    N = state.N;
    cs = zeros(N,1);
    sn = zeros(N,1);
    for k = 1:length(cs)
        i = imap(k,2);
        if i == 1
            sn(k) = sin(in(k));
            cs(k) = cos(in(k));
        else
            sn(k) = sn(k-1) + sin(in(k));
            cs(k) = cs(k-1) + cos(in(k));
        end
    end
end

x = zeros(N,1);
y = zeros(N,1);

for k = 1:N
    j = imap(k,1);
    x(k) = delta(j) + sn(k);
    y(k) = cs(k);
end

end
```

spiralForce.m

```
function out = spiralForce(t,y,state,imap,icheck,nUpdate, +
nRadius)
    persistent F;
    persistent fcount;
    persistent ltime;
    persistent ptime;

    n = state.n;
    m = state.m;
    beta = state.beta;
    len = state.len;
    lambda = state.lambda;
    mu = state.mu;
    epsi = state.epsi;
    sigma = state.sigma;
    delta = state.delta;
    N = state.N;
        s2 = sigma.^2;
        E = 0;

    if t == 0
        ltime = cputime;
        fcount = 0;
        ptime = 0;
    end
    fcount = fcount + 1;

    % Timing
    if mod(fcount,icheck) == 0
        ttime = cputime;
        elpsd = ttime - ltime;
        ptime = ptime + elpsd;
        pcnt = 100 * (t / state.tend);
        cmpltTime = ( ( ptime * 100 / pcnt ) - ptime ) / 60;
        fprintf('%.2f %%%, %.5f s, %.5f m; %.5f m \n',pcnt +
,elpsd,ptime/60,cmpltTime);
        ltime = cputime;
    end

    %% Construct Cosine and Sine lookup tables
    cs = zeros(N,1);
    sn = zeros(N,1);
    for k = 1:length(cs)
```

```

i = imap(k,2);
if i == 1
    sn(k) = sin(y(k));
    cs(k) = cos(y(k));
else
    sn(k) = sn(k-1) + sin(y(k));
    cs(k) = cs(k-1) + cos(y(k));
end
end

%% Update Neighborhood
if mod(fcount,nUpdate) == 0 || fcount == 1
    F = genNghd(y,cs,sn,state,imap,nRadius);
end

out = zeros(length(y),1);

%% V.W. /w Substrate + Bending + Load
%%%%{
z = length(y);
for j = m:-1:1
    for i = n(j):-1:1

        %% Bending
        %%{
        if i == 1
            out(z) = 0; %%out(z) - beta*(y(z) - y(z+1)) - beta*y( +
z);
        elseif i == n(j)
            out(z) = out(z) + beta*(sin(y(z-1) - y(z)));
        else
            out(z) = out(z) - beta*(sin(y(z) - y(z+1))) ...
                + beta*(sin(y(z-1) - y(z)));
        end
        %%}

        %% Bending
        %%%%{
        if i == 1
            out(z) = 0; %%out(z) - beta*(y(z) - y(z+1)) - beta*y( +
z);
            elseif i == 2
                out(z) = out(z) - beta*(sin((y(z) - y(z+1) +
))/2)./(cos((y(z) - y(z+1))/2).^3));

```

```

elseif i == n(j)
    out(z) = out(z) + beta*(sin((y(z-1) - y(z))/2)./(cos( +
(y(z-1) - y(z))/2).^3));
else
    out(z) = out(z) - beta*(sin((y(z) - y(z+1))/2)./(cos( +
(y(z) - y(z+1))/2).^3)) ...
        + beta*(sin((y(z-1) - y(z))/2)./(cos((y(z-1) - +
y(z))/2).^3));
    end
    if i > 2
        E = E + beta*tan((y(i)-y(i-1))/2)^2;
    end
%%}}

%% Bending Quadratic
%%{
mc = 0;
cp = 0;
if i ~= 1 && y(z-1) - y(z) >= 2*pi
    mc = y(z-1) - y(z) - 2*pi;
elseif i ~= 1 && y(z-1) - y(z) <= 0
    mc = y(z-1) - y(z) + 2*pi;
end

if i ~= n(j) && y(z) - y(z+1) >= 2*pi
    cp = y(z) - y(z+1) - 2*pi;
elseif i ~= n(j) && y(z) - y(z+1) <= 0
    cp = y(z) - y(z+1) + 2*pi;
end

if i == 1
    out(z) = 0; %%out(z) - beta*(y(z) - y(z+1)) - beta*y( +
z);
elseif i == n(j)
    out(z) = out(z) + beta*mc;
    %%E = E + (beta/2) * (y(z-1) - y(z)).^2;
else
    out(z) = out(z) - beta*cp + beta*mc;
    %%E = E + (beta/2)*(y(z) - y(z+1)).^2 ...
    %%      + (beta/2)*(y(z-1) - y(z)).^2;
end
%%}}

z = z - 1;

```

```

    end
end
%%}}

%% V.W. atom-to-atom
%%%%{
for h = 1:length(F)
    a = F(h,1);
    b = F(h,2);
    j = imap(a,1);
    i = imap(a,2);
    r = imap(b,1);
    k = imap(b,2);

    xps = sn(a) - sn(b);
    yps = cs(a) - cs(b);
    d2 = xps.^2 + yps.^2;
        s2d2 = s2./d2; s2d22 = s2d2.^2; s2d24 = s2d22.^2;
        LenJ = 12./s2.*(s2d24.*s2d22.*s2d2-s2d24);
        E = E + s2d24.*s2d22 - 2*s2d22.*s2d2;

        temp = 0;

        s = a - i + 1;
    for q = s:a
        atemp = (xps*cos(y(q)) - yps*sin(y(q)));
        out(q) = out(q) + LenJ*atemp;
    end

    s = b - k + 1;
    for q = s:b
        btemp = (yps*sin(y(q)) - xps*cos(y(q)));
        out(q) = out(q) + LenJ*btemp;
    end
        end
%% end
%%}}
if mod(fcount,icheck) == 0
    fprintf('      E = %%6.6g\n',E)
end
end

```


spiralEnergy.m

```
function E = spiralEnergy(y, state, imap, icheck, nUpdate, nRa +
dius)
    persistent F;
    persistent fcount;
    persistent ltime;
    persistent ptime;
n = state.n;
    m = state.m;
    beta = state.beta;
    len = state.len;
    lambda = state.lambda;
    mu = state.mu;
    epsi = state.epsi;
    sigma = state.sigma;
    delta = state.delta;
    N = state.N;
    s2 = sigma.^2;
    E = 0;

    fcount = 1;
    % Construct Cosine and Sine lookup tables
    cs = zeros(N,1);
    sn = zeros(N,1);
    for k = 1:length(cs)
        i = imap(k,2);
        if i == 1
            sn(k) = sin(y(k));
            cs(k) = cos(y(k));
        else
            sn(k) = sn(k-1) + sin(y(k));
            cs(k) = cs(k-1) + cos(y(k));
        end
    end

    % Update Neighborhood
    if mod(fcount,nUpdate) == 0 || fcount == 1
        F = genNghd(y,cs,sn,state,imap,nRadius);
    end

    out = zeros(length(y),1);

    % V.W. /w Substrate + Bending + Load
```

```

%%{
z = length(y);
for j = m:-1:1
    for i = n(j):-1:1

        % Bending
        %{
            if i == 1
                out(z) = 0; %out(z) - bet +
a*(y(z) - y(z+1)) - beta*y(z);
            elseif i == n(j)
                out(z) = out(z) + beta*(s +
in(y(z-1) - y(z)));
            else
                out(z) = out(z) - beta*(s +
in(y(z) - y(z+1))) ...
+ beta*(sin(y(z-1) - y(z)));
            end
        %}

        % Bending
        %%{
%           if i == 1
%               out(z) = 0; %out(z) - b +
eta*(y(z) - y(z+1)) - beta*y(z);
%               elseif i == 2
%                   out(z) = out(z) - beta*(sin((y(z) - y(z +
+1))/2)./(cos((y(z) - y(z+1))/2).^3));
%                   elseif i == n(j)
%                       out(z) = out(z) + beta* +
(sin((y(z-1) - y(z))/2)./(cos((y(z-1) - y(z))/2).^3));
%                       else
%                           out(z) = out(z) - beta* +
(sin((y(z) - y(z+1))/2)./(cos((y(z) - y(z+1))/2).^3)) ...
%
%                   end
                if i >2
                    E = E + beta*tan((y(i)-y(i-1))/2)^2;
                end
            %}

        % Bending Quadratic
        %{

```

```

                                mc = 0;
                                cp = 0;
                                if i ~= 1 && y(z-1) - y(z) >= 2*pi
                                    mc = y(z-1) - y(z) - 2*pi;
                                elseif i ~= 1 && y(z-1) - y(z) <= +
0
                                    mc = y(z-1) - y(z) + 2*pi;
                                end

                                if i ~= n(j) && y(z) - y(z+1) >= +
2*pi
                                    cp = y(z) - y(z+1) - 2*pi;
                                elseif i ~= n(j) && y(z) - y(z+1) +
<= 0
                                    cp = y(z) - y(z+1) + 2*pi;
                                end

                                if i == 1
                                    out(z) = 0; %out(z) - bet +
a*(y(z) - y(z+1)) - beta*y(z);
                                elseif i == n(j)
                                    out(z) = out(z) + beta*mc;
                                    %E = E + (beta/2) * (y(z-
1) - y(z)).^2;
                                else
                                    out(z) = out(z) - beta*cp +
+ beta*mc;
                                %E = E + (beta/2)*(y(z) - y(z+1)).^2 ...
                                %      + (beta/2)*(y(z-1) - y(z)).^2;
                                end
                                %}

                                z = z - 1;

                                end
                                end
                                %}

                                % V.W. atom-to-atom
                                %%{
                                for h = 1:length(F)
                                    a = F(h,1);
                                    b = F(h,2);
                                    j = imap(a,1);

```

```

        i = imap(a,2);
        r = imap(b,1);
        k = imap(b,2);

        xps = sn(a) - sn(b);
        yps = cs(a) - cs(b);
        d2 = xps.^2 + yps.^2;
s2d2 = s2./d2; s2d22 = s2d2.^2; s2d24 = s2d22.^2;
LenJ = 12./s2.*(s2d24.*s2d22.*s2d2-s2d24);
E = E + s2d24.*s2d22 - 2*s2d22.*s2d2;

temp = 0;

% s = a - i + 1;
%     for q = s:a
%         atemp = (xps*cos(y(q)) - yps*sin(y(q)));
%         out(q) = out(q) + LenJ*atemp;
%     end
%
%     s = b - k + 1;
%     for q = s:b
%         btemp = (yps*sin(y(q)) - xps*cos(y(q)));
%         out(q) = out(q) + LenJ*btemp;
%     end
% end
% }
if mod(fcount,icheck) == 0
    fprintf('    E = %6.6g\n',E)
end
end

```

R2_Space.m

```
function[x,y] = R2_Space(n,psi)

% This function finds the R2 space of the atoms. The orig +
in is the first
% point of the spiral.

x = zeros(n,1); y = x;

for j = 2:n
    x(j) = x(j-1) + sin(psi(j-1));
    y(j) = y(j-1) + cos(psi(j-1));
end
end
```

plotter.m

```
for i=1:length(t)
    if mod(i,1)==0
[X,Y]=R2_Space(n-1,y(i,2:n-1));
figure(1)
pause(0.001)
clf(figure(1))
subplot(2,1,1)
hold on
plot(X,Y,'black')
plot(X,Y,'blueo')
axis equal
subplot(2,1,2)
plot(t(1:i),E(1:i)), title('Energy v. time'), axis([0,t(e +
nd), floor(min(E(1:i))), floor(max(E(1:i)))+1])
fprintf('t = %5.5g\n',t(i))
    end
end
```

plotter_2.m

```
for i=1:500:length(t)
[X,Y]=R2_Space(n,y(i,2:n));
figure(1)
pause(0.001)
clf(figure(1))
hold on
plot(X,Y,'blueo')
axis equal
fprintf('t = %5.5g\n',t(i))
end
```

plot_init.m

```
[X,Y]=R2_Space(n,y(1,2:n));
figure(1)
clf(figure(1))
hold on
plot(X,Y,'black')
plot(X,Y,'blueo')
axis equal
```

plot_both.m

```
figure(1)
clf(figure(1))
hold on

[X,Y]=R2_Space(n,y(1,2:n));
subplot(1,2,1),plot(X,Y,'black',X,Y,'blacko'),title('Init +
ial'), axis equal

[X,Y]=R2_Space(n,y(end,2:n));
subplot(1,2,2),plot(X,Y,'red',X,Y,'redo'),title('Final'), +
axis equal

fprintf('t_final = %5.5g\n',t(end))
```

plot_both_alt.m

```
figure(1)
clf (figure(1))
hold on

[X,Y]=R2_Space(n,y(1,2:n));
plot(X,Y,'red:'), axis equal

[X,Y]=R2_Space(n,y(end,2:n));
plot(X,Y,'blue'), axis equal

[X,Y]=R2_Space(n,y(1,2:n));
plot(X,Y,'redx'), axis equal

[X,Y]=R2_Space(n,y(end,2:n));
plot(X,Y,'blueo'), axis equal

legend('Initial','Final')
fprintf('t_final = %5.5g\n',t(end))
```

plot_both_alt2.m

```
figure(1)
clf (figure(1))
hold on

[X,Y]=R2_Space(n,y(1,2:n));
plot(X,Y,'red',X,Y,'redo'), axis equal

figure(2)
clf (figure(2))
hold on

[X,Y]=R2_Space(n,y(end,2:n));
plot(X,Y,'blue',X,Y,'blueo'), axis equal

fprintf('t_final = %5.5g\n',t(end))
```