2021

A Molecular Dynamic Study on the Piezoelectric Properties of Bulk ZnS And Nanobelts

Rui Xie

A MOLECULAR DYNAMIC STUDY ON THE PIEZOELECTRIC PROPERTIES OF BULK ZnS AND NANOBELTS

A thesis submitted to
the Graduate College of
Marshall University
In partial fulfillment of
the requirements for the degree of
Master of Science
In
Mechanical Engineering
by
Rui Xie
Approved by
Dr. Iyad Hijazi, Committee Chairperson
Dr. Gang Chen
Dr. Yousef Sardahi

Marshall University
December 2021

APPROVAL OF THESIS

We, the faculty supervising the work of Rui Xie, affirm that the thesis, "*A Molecular Dynamic Study On The Piezoelectric Properties Of Bulk ZnS And Nanobelts*", meets the high academic standards for original scholarship and creative work established by the College of Engineering and the Computer Science of the Master of Science in Mechanical Engineering. This work also conforms to the editorial standards of our discipline and the Graduate College of Marshall University. With our signatures, we approve the manuscript for publication.

Dr. Iyad Hijazi, Weisberg Division of Engineering	Committee Chairperson	Date
		10/8/2021
Dr. Gang Chen, Weisberg Division of Engineering	Committee Member	Date
		10/8/2021
Dr. Yourself Sardahi, Weisberg Division of Engineering	Committee Member	Date
		10-08-2021

ACKNOWLEDGMENTS

I would like to express my gratitude to all those who helped me during the writing of this thesis. My deepest gratitude goes first and foremost to Professor Iyad Hijazi, my supervisor, for his constant encouragement and guidance. He has walked me through all the stages of the writing of this thesis. Without his consistent and illuminating instruction, this thesis could not have reached its present form. Furthermore, He has spent much time reading through each draft and provided me with inspiring advice. Without him patient instruction, insightful criticism and expert guidance, the completion of this thesis would not have been possible. Also, I would like to thank all the professors from Weisberg Division of Engineering and the Marshall University library who supported me during my thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In this thesis we proved the feasibility of using classical atomic simulations, namely molecular dynamics and molecular statics, to study the piezoelectric properties of bulk and nanobelts ZnS structures, by utilizing the core-shell atomic potential model. Based on the verification of bulk and nanobelts ZnO piezoelectric constants, utilizing LAMMPS scripts and the Nyberg et al. core-shell potential, we reported the bulk ZnS piezoelectric constants calculated using three different classical interatomic core-shell ZnS potentials; the Wright and Jackson (1995) potential, the Wright and Gale (2004) potential, and the Namsani et al. (2015) potential. The simulation results showed that the Wright and Gale (2004) ZnS potential, which includes a four-body bonded term, is the most reliable potential to be used for large-scale atomic simulation of piezoelectric response of the bulk ZnS structures. Utilizing the Wright and Gale (2004) potential, we further studied the effect of size scale effect on the piezoelectric response of ZnS nanobelts by conduction molecular dynamics simulations for six ZnS nanobelts with length of 91.75 Å and transverse size of 22.94 - 42.06 Å. The results showed that, as with the ZnO nanobelts, the change of piezoelectric constant decreased with the increase of the size of the ZnS nanobelts structures.

CHAPTER 1

INTRODUCTION

Piezoelectric materials have long attracted the attention of scientists and engineers because they can convert mechanical strain into available electrical energy [1-3]. This characteristic is termed piezoelectric. Basically, it is the accumulation of charges within certain solid materials, such as crystals or ceramics, when exposed to mechanical stress. Several research communities have been interested in metal oxide [4-15] and metal sulfide [16-23] because of their ability to convert mechanical strain energy into electrical energy in nano/micro devices.

Among the metal oxide and metal sulfide, zinc oxide (ZnO) and zinc sulfide (ZnS) have attracted the attention of many researchers [11-23]. Both ZnO and ZnS have hexagonal wurtzite (WZ) and cubic zinc blende (ZB) structures. ZnO can also exist as rock salt, with the WZ structure being the most stable phase at ambient conditions [11, 12], while for ZnS, the ZB structure, also called sphalerite, is the more stable cubic form [18]. Sphalerite can easily transform into wurtzite at 1013 ℃ or 1031 ℃ temperatures, depending on sulfur activity [23]. In the tetrahedral bonded semiconductors, ZnO has the highest piezoelectric tensor, which makes it an important technical material in many practical applications, such as mechanical actuators and piezoelectric sensors [5, 7]. Similarly, in wide bandgap semiconductors, ZnS is an important material for designing high-performance optoelectronic devices, including light-emitting diodes and laser diodes in blue or ultraviolet regions [16]. In addition, by combining the anionic alloying of sulfur (S) and ZnO, Zn (O, S) mixed crystals can be prepared for the development of more efficient solar cells [25].

Over the recent years, several experimental and numerical studies have been carried out to understand the experimental, mechanical, and piezoelectric properties of ZnO [1-7, 11, 13]. Agrawal et al. presented an experimental and computational method to clearly quantify the size effect on young's modulus of ZnO nanowires and explain the origin of scaling [1]. Zhao et al. used piezo-response force microscopy (PFM) to measure the effective piezoelectric coefficient of a single (0001) surface dominated ZnO nanobelt lying on a conductive surface [2]. Noel et al. conducted an ab initio study of the spontaneous polarization and piezoelectric constants of bulk ZnO and BeO [3]. Corso et al. demonstrated the feasibility of ab initio studies of piezoelectricity within an all-electron scheme with a focus on wurtzite ZnO and compared the results with BeO and ZnS [5]. Hill et al. presented a first-principles study of the relationship between stress, temperature, and electronic properties in piezoelectric bulk ZnO [7]. Dai et al. demonstrated the feasibility of using molecular dynamics (MD) and molecular statics (MS), to study the piezoelectric properties of bulk ZnO using core-shell interatomic potentials [6]. Momeni et al. investigated the size scale effect on the piezoelectric properties of bulk ZnO and nanobelts by using classical MD simulation [11]. Wang et al. used the MD method to investigate the tensile behavior of ultra-thin ZnO nanowires in <0001> orientation with three different diameters [13].

Furthermore, much research studied the synthesis, mechanical and piezoelectric properties of bulk ZnS and nanobelts [5, 16-18, 20, 35]. Wang et al. synthesized single-crystalline and pure ZnS nanobelts with periodically modulated thickness through a controllable and simple one-step chemical vapor deposition (CVD) method [16]. Li et al. measured the mechanical properties of ZnS nanobelts at room temperature by direct nanoindentation experiment [17]. Corso et al. conducted an ab initio study to calculate the theoretical piezoelectric tensor component of bulk ZnS and compared to the experimental values [5].

Ferahtia et al. used ab initio calculation to investigate the structure, elasticity, and piezoelectric properties of wurtzite ZnS and ZnSe [18]. Catti et al. also used ab initio calculation to calculate the complete piezoelectric tensor of ZnO and ZnS wurtzite and sphalerite polymorphs [20]. Using MD simulation, Khalkhali et al. studied the structural evolution of 1-5nm independent ZnS nanoparticles with sphalerite and wurtzite crystal structures [35].

Ab initio calculations are reliable simulation methods, but they are not feasible for the simulation of most systems containing a few hundreds of atoms due to their high computational cost and do not include finite temperature, or thermal effects [3]. Molecular dynamics (MD) simulations can avoid the temperature and length scale problems related to ab initio calculations by providing a more practical method of simulating a more realistic system size of more atoms [33].

A number of researchers have developed interatomic potentials for ZnS [21-23, 28-30, 34]. Khalkali et al. reviewed five ZnS empirical potentials and tested their performance in predicting different ZnS properties. The properties tested in their work include the mechanical and structural properties, phonon dispersion relation, surface energy and structure, behavior under pressure, thermal expansion, and energy hypersurface [34]. The author concluded that reliability of a particular empirical potentials and the choice of potential is highly dependent on the application that the molecular mechanic simulation aims for, and therefore each of these potentials is designed to reproduce some specific ZnS properties [34]. In this thesis, we investigated three classical interatomic core-shell potentials and report their accuracy in predicting the piezoelectric response to the bulk ZnS and the influence of size scale effect on the piezoelectric response to ZnS nanobelts through MD simulations.

CHAPTER 2

METHODS

In this work, three ZnS potentials based on the Buckingham-type potential and labeled PT1, PT2, and PT3, were considered. PT1 is the Wright and Jackson potential [21], PT2 is the Wright and Gale potential [22], and PT3 is the Namsani et al. potential [23]. The three potentials can be represented as either a traditional core-only (rigid ion) potential, or a core–shell potential, in which each point ion consists of a core of charge *X* and a shell of charge *Y* such that the sum of the core and shell charges *(X + Y)* equals the formal charge of the ion. The Wright and Jackson potential (PT1) combines a Buckingham interaction with a shell model for both Zn and S**,** plus a bonded harmonic three-body term for the S-Zn-S bond. The Buckingham (short range) + Coulomb (long-range) has the form

$$U_{ij} = \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}} + \sum_{ij} A_{ij} exp\left[\frac{rij}{\rho ij}\right] - C_{ij} r_{ij}^{-6} \quad (1)$$

And a three-body term is given by

$$U_{ijk}^{b} = \frac{1}{2} k_\theta \left(\theta_{ijk} - \theta_0\right)^2 \quad (2)$$

The potential parameters were derived by empirical fitting to crystal properties and are presented in Table 1, along with experimental lattice constants, where the internal parameter u defined as the anion-cation bond length along the [0001] axis in units of c.

Table 1. Parameters for PT1, PT2, and PT3 ZnS potentials. PT1 is the Wright and Jackson potential, PT2 is the Wright and Gale potential, and PT3 is the Namsani et al. potential [21-23].

	Species	A (eV)	ρ (Å)	C (eV Å^{-6})	Cut-off (Å)
PT1	**Zn core – S core**	528.889	0.411	0.000	12.000
	S core – S core	1200.000	0.149	120.000	
PT2	**Zn core – S shell**	672.288	0.39089	0.00	12.000
	S shell – S shell	1200.0	0.149	0.0	
PT3	**Zn core – S shell**	580.84615	0.400505	0.00	12.000
	S shell – S shell	1199.78975	0.148604	0.0	
	Three-body potential	**k(eV/rad^{-2})**	**θ_0(°)**	**ρ1/ρ2(Å)**	**Cut-off (Å)**
PT1	**S core – Zn core – S core**	0.713	109.47		12.0
PT2	**S shell – Zn core – S shell**	9428340	109.47	0.3	6.0
PT3	**S shell – Zn core – S shell**	7000859.729	109.47	0.3	12.0
	Torsional potential	**K_t (eV)**	**m / n**	**r_{min} (Å)**	**r_{max} (Å)**
PT2	**Zn core - S shell – Zn core - S shell**	0.005	+1/ +3	2.5	3.0
	Charge (e)	**Zn core**	**Zn shell**	**S core**	**S shell**
	PT1	+2.00	0.00	-2.00	0.00
	PT2	+2.00	0.00	+1.03061	-3.03061
	PT3	+2.00	0.00	+1.087526	-3.087526
	Lattice Constants	***a* (Å)**	***c* (Å)**	$\boldsymbol{u = \frac{\lvert z(O) - z(Zn)\rvert}{c}}$	
		3.8227	6.2607	0.3748	

Wright and Gale ZnS core-shell potential model (PT2) includes in addition to the two-body potential and three-body potentials in Equations (1) & (5), higher order four-body potential to allow for torsional effects. The form of the torsion potential is shown in Equation (3) & (4) and includes a taper term to limit the spatial extent of the interaction smoothly, again to prevent discontinuous behavior as the coordination geometry varies. The potential parameter for ZnS given by

$$U_{ijkl}^{t} = k_t[1 + m\cos(n\varphi - \varphi_0)]f(r_{ij})f(r_{jk})f(r_{kl}) \quad (3)$$

Where the taper function $f(r)$ takes the form,

$$\mathrm{r} < \mathrm{r}_{min}\ f(r_{ij}) = 1$$

$$r_{min} < r < r_{max}\ f(r) = \frac{1}{2}\left\{1 + \cos\left(\pi\frac{(r - r_{min})}{r_{max} - r_{min}}\right)\right\}$$

$$r_{max} < r\,f(r) = 0 \quad (4)$$

Namsani et al. ZnS core-shell potential model (PT3) combines the above Buckingham two-body potential in Equation (1) with a harmonic three-body potential. Their three-body potential contains exponential decay terms, as shown in Equation (5), to avoid discontinuous behavior of the potential with respect to the atom movement in the coordination shells.

$$U_{ijk}^{b} = \frac{1}{2} k_{\theta} \left(\theta_{ijk} - \theta_0\right)^2 \exp\left(-\frac{r_{ij}}{\rho_1}\right) \exp\left(-\frac{r_{ik}}{\rho_2}\right) \quad (5)$$

Their potentials fitting parameters are listed in Table 1. As it can be seen in the Table 1, the Zn ion was taken to be rigid (i.e., no shell) with a fixed formal charge of 2+, while the S ion was assumed to be more polarizable and so described by a core and shell.

As a test for the reliability of the present potentials, the bulk modulus for the bulk ZnS was calculated and compared to the experimental results. Volumes at various pressures and temperatures of 1 K and 300 K were calculated. Using a numerical differentiation method, the bulk modulus was calculated from the pressure-volume data and the relation

$$B = -V \frac{\partial P}{\partial V} \quad (6)$$

where P is pressure, V is volume, and $\frac{\partial P}{\partial V}$ denotes the partial derivative of pressure with respect to volume. Table 2 lists the calculated bulk modulus values obtained from the present three potentials with core-core and core-shell MD simulations. From Table 2 it can be seen that our calculated values using PT2 and PT3 potentials are in good agreement with the experimental value and those obtained by PT2 and PT3. The largest deviation from the experimental value resulted from using the PT1 ZnS potential. Figure 1 shows the pressure vs volume curve for ZnS bulk modulus using the PT2 potential.

Table 2. Comparison of calculated bulk modulus for ZnS wurtzite structure with experimental data and other works.

Bulk modulus	Temperature (K)	This work PT1 (GPa)	This work PT2 (GPa)	This work PT3 (GPa)
Core-core	1	69.49	73.66	72.32
Core-core	300	69.70	74.08	71.55
Core-shell	1	-	73.36	72.05
Core-shell	300	-	72.23	70.91
Other works	-	-	76.4 [3]	70.91 [4]
Exp.		74.0 [22]		

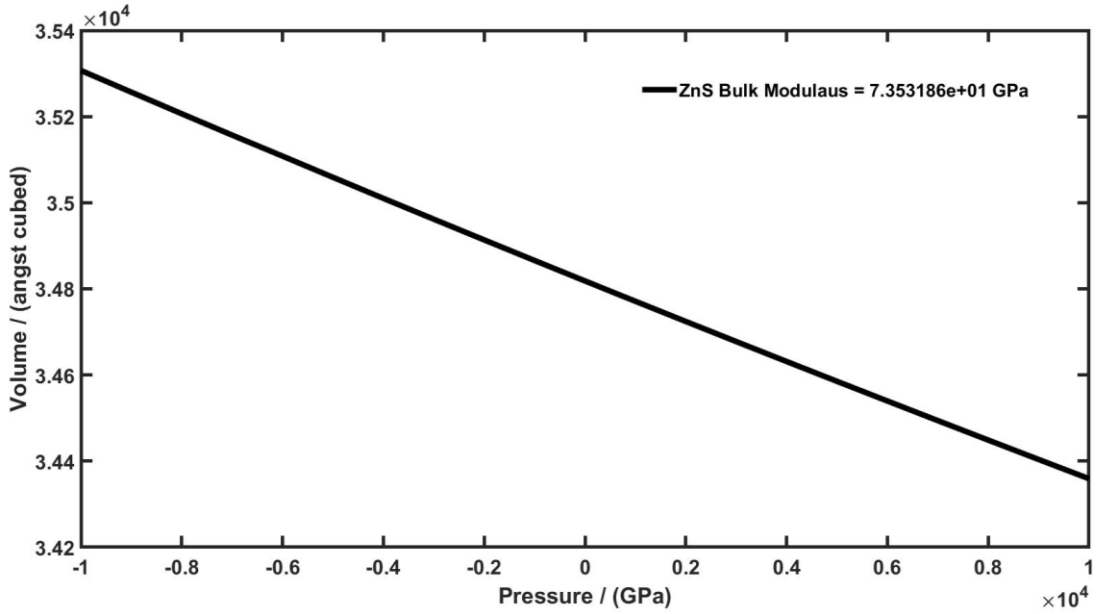


Figure 1. Calculated ZnS bulk modulus using the PT2 potential [22].

CHAPTER 3

BULK ZINC OXIDE CORE-CORE AND CORE-SHELL SIMULATIONS

3.1 Bulk Zinc Oxide Core-Core Simulations

Dai et al. calculated the piezoelectric constants for bulk ZnO [6]. In their MD simulations, they considered two classical core-shell potentials, that of Binks et al. [37] and Nyberg et al. [38], where both potentials take the Buckingham-type form shown in Equation (1) and can be used as either a traditional core-only (rigid ion) potential, or a core-shell potential. To validate our LAMMPS MD simulation code that will be utilized to calculate the piezoelectric coefficients for bulk ZnS, we first utilized the script code to run both core-core and core-shell MD simulations using the Buckingham potential to calculate the piezoelectric coefficients for bulk ZnO. The Nyberg potential parameters that were used in the simulations are listed in Table 3. Following the modeling procedure of Dai et al. [6], we run MD simulations by considering

Table 2. Buckingham & Core-shell parameters with electric charges and Lattice parameters for the Nyberg et al. for ZnO [38].

Nyberg	Species	A (eV)	ρ (Å)	C (eV Å^6)
	$O^{2-} - O^{2-}$	22764.0	0.1490	27.88
	$Zn^{2+} - O^{2-}$	499.6	0.3595	0.0
	$Zn^{2+} - Zn^{2+}$	0.0	0.0	0.0
Charges	**Zn core**	**O core**	**Zn shell**	**O shell**
	-0.05	0.00	2.05	-2.00
Lattice Constants	a (Å)	c (Å)	$u = \frac{\lvert z(O) - z(Zn)\rvert}{c}$	
	3.2495	5.2069	0.375	

a wurtzite ZnO structure with 6 × 6 × 6-unit cells for a total of 1728 atoms (see Figure 3). Periodic boundary conditions were imposed in all three coordinate directions to mimic a bulk ZnO crystal. MD annealing simulations with a Nose-Hoover NPT ensemble from 500 K to 0.1 K in 100 picoseconds were performed during each run. All classical atomistic simulations were

performed using the open-source molecular simulation code LAMMPS. Each NPT annealing simulation was followed by non-equilibrium MD (NEMD) simulation with Nose-Hoover NVT ensemble at 0.1K. Therefore, during each NVT run the volume of the simulation box was changed by applying strain rates of 10^8 s^{-1} in tension and compression along the z-direction in 100 picoseconds resulting in small strains between −1 and 1%. During the deformation process, the strain and resulting ZnO lattice parameters and volume were saved to a text file at various time steps. The structure atomic types and positions were also saved to an XMol XYZ file format. Utilizing the MD simulation data, the piezoelectric constants were calculated using Equations (7) - (9).

$$e_{ij} = e_{ij}^{(0)} + \mathrm{e}_{\mathrm{ij}}^{\mathrm{int}} \tag{7}$$

$$e_{ij}^{(0)} = \left.\frac{\partial P_i}{\partial \varepsilon_j}\right|_u \tag{8}$$

$$e_{ij}^{int} = \left.\frac{\partial P_i}{\partial u}\right|_{\varepsilon_j} \frac{\partial u}{\partial \varepsilon_j} \tag{9}$$

where $e_{ij}^{(0)}$ is the clamped ion term, which represents the influence of electron delocalization on the piezoelectric properties, while e_{ij}^{int} is the internal piezoelectric constant, which represents the contribution to the piezoelectricity of the change in position of atoms due to applied strain, u is the fractional crystallographic coordinate along the *k*th direction in the unit cell, P_i is the electric dipole moment vector, and ε_j is the strain in Voigt notation. It should be noted that e_{ij} is a 3 × 6 matrix.

To calculate the piezoelectric response associated with the c-axis the polarization was determined from

$$P_3 = P_3\left(u(\varepsilon_3), \varepsilon_3\right) = P_3^{elec}(\varepsilon_3) + P_3^{dis}\big(u(\varepsilon_3)\big) \tag{10}$$

where $P_3^{elec}(\varepsilon_3)$ is the polarization due to the relative displacement of electrons and core, and $P_3^{dis}\big(u(\varepsilon_3)\big)$ represents the polarization due to the relative displacement between zinc and sulfur atoms. From Equation (9) the piezoelectric constants, e_{33}, can be calculated as

$$e_{33} = \frac{\partial P_3}{\partial \varepsilon_3} = \frac{\partial P_3^{elec}(\varepsilon_3)}{\partial \varepsilon_3} + \frac{\partial P_3^{dis}(u(\varepsilon_3))}{\partial \varepsilon_3},$$

$$= \frac{\partial P_3^{elec}(\varepsilon_3)}{\partial \varepsilon_3} + \frac{\partial P_3^{dis}(u(\varepsilon_3))}{\partial \varepsilon_3} \frac{\mathrm{d}u(\varepsilon_3)}{\mathrm{d}\varepsilon_3},$$

$$= \left.\frac{\partial P_3}{\partial \varepsilon_3}\right|_u + \left.\frac{\partial P_3}{\partial u(\varepsilon_3)}\right|_{\varepsilon_3} \frac{\partial u(\varepsilon_3)}{\partial \varepsilon_3},$$

$$= e_{33}^{(0)} + \mathrm{e}_{33}^{\mathrm{int}} \tag{11}$$

The polarization P_3 is calculated as:

$$P_3 = e_{31}\epsilon_1 + e_{32}\epsilon_2 + e_{33}\epsilon_3 = \frac{\sum_i r_i q_i}{V_{cell}} = \frac{-4qu}{\sqrt{3}a^2(1+\epsilon_1)(1+\epsilon_2)} \tag{12}$$

where $e_{31} = e_{32}$, $\varepsilon_1 = \varepsilon_2 = \varepsilon = \frac{a-a_0}{a_0}, \varepsilon_3 = \frac{(c-c_0)}{c_0}$

and e_{33} is giving by

$$\tilde{\mathrm{e}}_{33} = \frac{\partial P_3}{\partial \varepsilon_3} = e_{33} = \frac{-4q}{\sqrt{3}a_0^2}\frac{\mathrm{d}u}{\partial \varepsilon_3} \tag{13}$$

In our ZnO MD simulations, we first considered only the core-core interaction, and the effect of polarization between the nucleus and electron cloud was neglected, which resulted in the elimination of the clamped ion term, i.e., $P_3^{elec}(\varepsilon_3) = 0$ and $e_{ij}^{(0)} = 0$ from Equations (7) to (11). The piezoelectric constants $\tilde{\mathrm{e}}_{33}$ calculated using the Nyberg potential are summarized in Table 3, including the two terms comprising the internal strain term, along with the proper and

improper piezoelectric constants are tabulated. Where the improper piezoelectric tensor is calculated as

$$e_{ijk} = \frac{\partial P_i}{\partial \varepsilon_{jk}} \tag{14}$$

Table 3. ZnO Results of e_{33} piezoelectric constants for core-core and core-shell.

	This work		Dai et al. [6]			
Methods / **Potentials**	**No shell Nyberg**	**Shell Nyberg**	**No shell Nyberg**	**Shell Nyberg**	**DFT [6]**	**Exp. [5, 44]**
$\boldsymbol{a}$	3.2305	3.2304	3.2303	3.2303		3.2495
$\boldsymbol{c}$	5.0772	5.0769	5.0767	5.0767		5.2069
$\boldsymbol{u}$	0.389	0.389	0.389	0.389		0.382
$\boldsymbol{P_3}$	-2.7589	-2.7598	-2.7587	-2.7587		
$\boldsymbol{P_3^{theo}}$	-2.6281	-2.6281	-2.6281	-2.6281		
$\boldsymbol{P_{eq} = P_3 - P_3^{theo}}$	-0.1308	-0.1317	-0.099	-0.099	-0.029→ -0.057	
$\boldsymbol{\partial u/\partial \varepsilon_3}$	-0.182	-0.193	-0.182	-0.211	-0.21→ -0.254	
$\boldsymbol{\partial P_3/\partial u}$	-7.09	-7.09	-7.09	-7.09	-7.10→ -7.60	
$\tilde{e}_{33}^{int} = \frac{\partial P_3}{\partial u}\frac{\partial u}{\partial \varepsilon_3}$	1.29	1.369	1.29	1.496		
$e_{33}^{int} = \frac{\partial P_3}{\partial \varepsilon_3}$	1.29	1.369	1.29	1.496		
$e_{33}^{(0)}$	0	-0.084	0	-0.22	-0.45→ -0.73	
$e_{33} = e_{33}^{(0)} + e_{33}^{int}$	1.29	1.285	1.29	1.27	0.89→ 1.31	0.79→1.18

From Table 4, it can be seen that our results for the core-only lattice constants, e_{33} piezoelectric constant, and related data obtained using the Nyberg potential are in complete agreements with those obtain by Dai et al. [6].

The value for the core-only polarization P_3 calculated at equilibrium or zero strain is given in Table 4 and is in perfect agreement with the value obtained by Dai et al. [6]. Table 4. also shows the spontaneous polarization P_{eq} calculations for the core-only bulk ZnO structure, where P_{eq} is defined as the difference between the theoretical polarization P_3^{theo} (i.e., when u = 0.375) and the minimum energy polarization P_3, which appears when u = 0.389 using the

Nyberg potential. Our P_{eq} value is in good agreement with those obtained by Dai et al., but as with Dai et al. our value is also larger than those obtained by DFT calculations [6].

Figure 2. shows the plots resulting from our MD simulations from which the core-only values in Table 4 were obtained by performing linear fits to the data in Figure 2. The proper piezoelectric constant e_{33} was obtained by evaluating the e_{33}^{int} term in Equation (11), which correspond to multiplying the two slops value of the $\frac{\partial P_3}{\partial u}$ and $\frac{\partial u}{\partial \varepsilon_3}$ plots in Figure 2. From Figure 2. it can be seen that the $P_3^{dis}\big(u(\varepsilon_3)\big)$ polarization values obtained using the fraction coordinates and the volume average utilizing Equation (11) are in perfect agreement. We note the slop $\frac{\partial P_3}{\partial u}$ can also be calculated analytically.

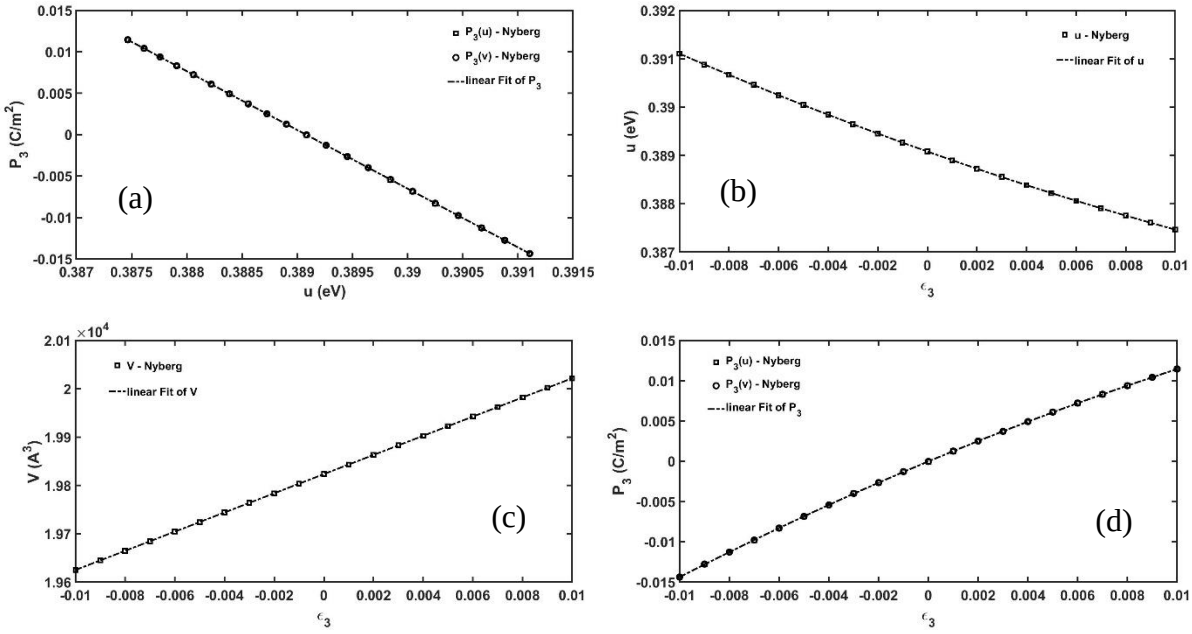


Figure 2. Plots utilized to calculate the proper and improper piezoelectric constant e_{33}^{int} without shell. (a) the polarization P_3 as a function of the fractional coordinate u without shell; (b) fractional coordinate u as a function of the strain ε_3; (c) the volume as a function of strain ε_3; (d) the polarization P_3 as a function of the strain ε_3 without shell.

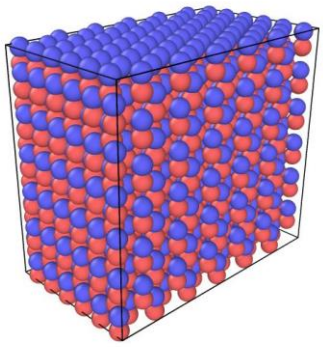
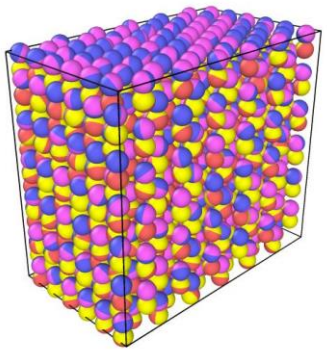
(a)  (b) 

Figure 3. 6x6x6 bulk ZnO structure for (a) core-only and (b) core-shell.

3.2 Bulk Zinc Oxide Core-Shell Simulations

In our core-shell ZnO MD code validation simulations, the interaction between the nucleus and electron cloud was taken into consideration, and the clamped ion terms, $P_3^{elec}(\varepsilon_3)$ and $e_{ij}^{(0)}$ from Equations (7) to (11) were included in the calculations of the polarization P_3 and piezoelectric constants e_{33}. Table 4 lists the results for the core-shell lattice constants, minimum energy polarization P_3, spontaneous polarization P_{eq}, and e_{33} piezoelectric constant and related data obtained using the Nyberg potential. As with the core-only simulation results our core-shell results are in excellent agreements with those obtain by Dai et al. [6].

Figure 4 shows the graphs resulting from our MD simulations from which the core-shell values in Table 4 were obtained by performing linear fits to the data in Figure 4. When the shell is considered, the $\frac{\partial u}{\partial \varepsilon_3}$ term becomes more negative due to increased relaxation effects, and thus becomes more accurate as compared to the DFT results. The proper (core-core) piezoelectric constant e_{33}^{int} was obtained by evaluating Equation (9), which correspond to multiplying the two slops value of the $\frac{\partial \mathrm{P}_3}{\partial \mathrm{u}}$ and $\frac{\partial \mathrm{u}}{\partial \varepsilon_3}$ plots in Figure 4(a) and 4(b) respectively. Figure 4(d) shows the P_3

= P_3^{dis} (core-core) polarization values alone with $P_3 = P_3^{elec}\ (core - shell) + P_3^{dis}\ (core - core)$ for the strain range from -0.01 to 0.01. The value of clamped ion term $e_{33}^{(0)}$ = -0.084 was obtained by taken the difference between the slops of the two figures.

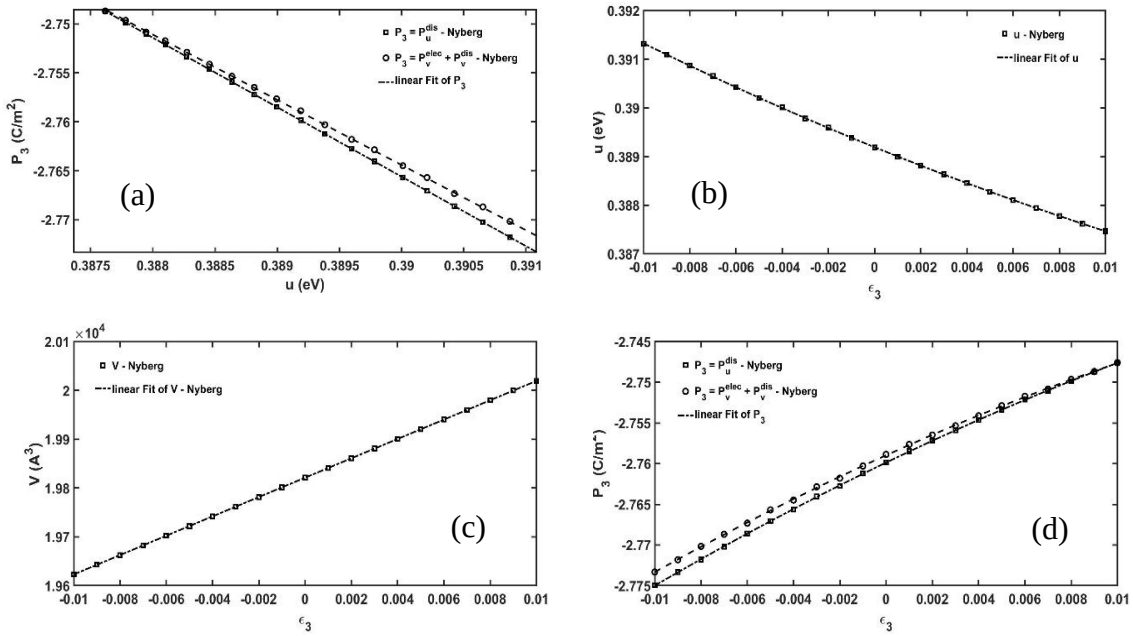


Figure 4. Plots utilized to calculate the proper and improper piezoelectric constant e_{33}^{int} with shell, (a) the polarization P_3 as a function of the fractional coordinate u with shell; (b) fractional coordinate u as a function of the strain ε_3; (c) the volume as a function of strain ε_3; (d) the polarization P_3 as a function of the strain ε_3 with shell.

CHAPTER 4

BULK ZINC SULFIDE CORE-CORE AND CORE-SHELL SIMULATIONS

4.1 Bulk Zinc Sulfide Core-Core Simulations

After validating our LAMMPS core-only MD simulation code in the calculation of the piezoelectric constants $\tilde{e}_{33}$ for the bulk ZnO, we utilized the code to conducted MD simulations using the three ZnS potentials; PT1, PT2, and PT3 potentials to calculate the core-only bulk ZnS piezoelectric constants. Since the PT2 and PT3 potentials include a harmonic three-body potential with truncation terms, shown in Equation (5), which is not part of LAMMPS library of potentials, we had to provide our own implementation for this potential. The harmonic three-body potential has the general form:

$$U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = A(\theta_{jik}) S(r_{ij}) S(r_{ik}) \tag{15}$$

where $A(\theta_{jik})$ is a purely angular function and $S(r)$is a screening or truncation function with all function arguments are scalars.

$$A(\theta_{jik}) = k(\theta_{jik} - \theta_0)^2 \tag{16}$$

$$S(r_{ij}) = exp\left(-\frac{r_{ij}}{\rho_1}\right) \tag{17}$$

$$S(r_{ik}) = exp\left(-\frac{r_{ik}}{\rho_2}\right) \tag{18}$$

Appendix B shows the equations derivation and LAMMPS C++ implementation. To extract the ZnS structural information, which includes the number of atoms, angles, and dihedrals, LAMMPS data file needs to run a simulation, we implemented our own extraction code utilizing MATLAB, included in Appendix B. Table 5 lists the structural information for ZnS with 6 × 6 × 6 unit cells used in our MD simulations. For each potential the long-range Coulombic forces and

energies were calculated using both the Ewald summation and Wolf summation methods [39], where the Wolf summation methods is given by:

$$E_{total}(r_{ij}) = \sum_{i=1}^{N}\sum_{j\neq i} A\ exp\left(-\frac{r_{ij}}{\rho}\right) - \frac{C}{r_{ij}^{6}} + E_{long}(r_{ij}) \quad (19)$$

$$E_{long} = \frac{1}{2}\sum_{j\neq i}\frac{q_i q_j erfc(\alpha r_{ij})}{r_{ij}} + \frac{1}{2}\sum_{j\neq i}\frac{q_i q_j erf(\alpha r_{ij})}{r_{ij}} \quad (20)$$

Table 4. The structural data for ZnS with 6x6x6 unit cells and periodic boundaries.

Type	Data
Atoms	1728
Angles	5184
Dihedrals	31104

From Table 6, it can be seen that the proper and improper e_{33} piezoelectric constants obtained using the PT2 and PT3 potentials with the Ewald summation method are in excellent agreement with the experimental data. From the table it can also be seen that the values are in closer agreement with the experimental data than the values obtained using the Wolf summation method. The largest deviation from experimental value was obtained using the PT1 potential.

Table 5. Summary of piezoelectric constants e_{33} calculated using classical interatomic potentials for bulk ZnS with core-only. In each potential the first and second data are obtained from Ewald summation and Wolf summation methods respectively [39].

Potentials	Methods	$\frac{\partial P_3}{\partial u}$	$\frac{\partial u}{\partial \varepsilon_3}$	$\tilde{e}_{33}^{int} = \frac{\partial P_3}{\partial u}\frac{\partial u}{\partial \varepsilon_3}$	$e_{33}^{int} = \frac{\partial P_3}{\partial \varepsilon_3}$	e_{33} (Exp)
PT1	**Ewald**	-5.0785	-0.0932	0.473	0.473	
	Wolf	-5.0272	-0.1006	0.505	0.505	
PT2	**Ewald**	-5.0074	-0.0512	0.256	0.256	0.265[a],0.27[b] 0.34[c],0.347[d] 0.43[e]
	Wolf	-4.9570	-0.0594	0.294	0.294	
PT3	**Ewald**	-5.0627	-0.0522	0.264	0.264	
	Wolf	-5.0131	-0.0630	0.315	0.315	

[a] Reference [40]., [b] Reference [5]., [c] Reference [9]., [d] Reference [41]., [e] Reference [42].

Figure 5. shows the graphs resulting from the MD simulations from which the values in Table 6 were obtained by performing linear fits to the data in Figure 5. The proper piezoelectric constant e_{33} was obtained by evaluating the e_{33}^{int} term in Equation (11), which correspond to multiplying the two slops value of the $\frac{\partial P_3}{\partial u}$ and $\frac{\partial u}{\partial \varepsilon_3}$ plots in Figure 5. From Figure 5. it can be seen that the $P_3^{dis}\big(u(\varepsilon_3)\big)$ polarization values obtained using the fraction coordinates and the volume average utilizing Equation (12) are in perfect agreement. We note that the slop $\frac{\partial P_3}{\partial u}$ can also be calculated analytically.

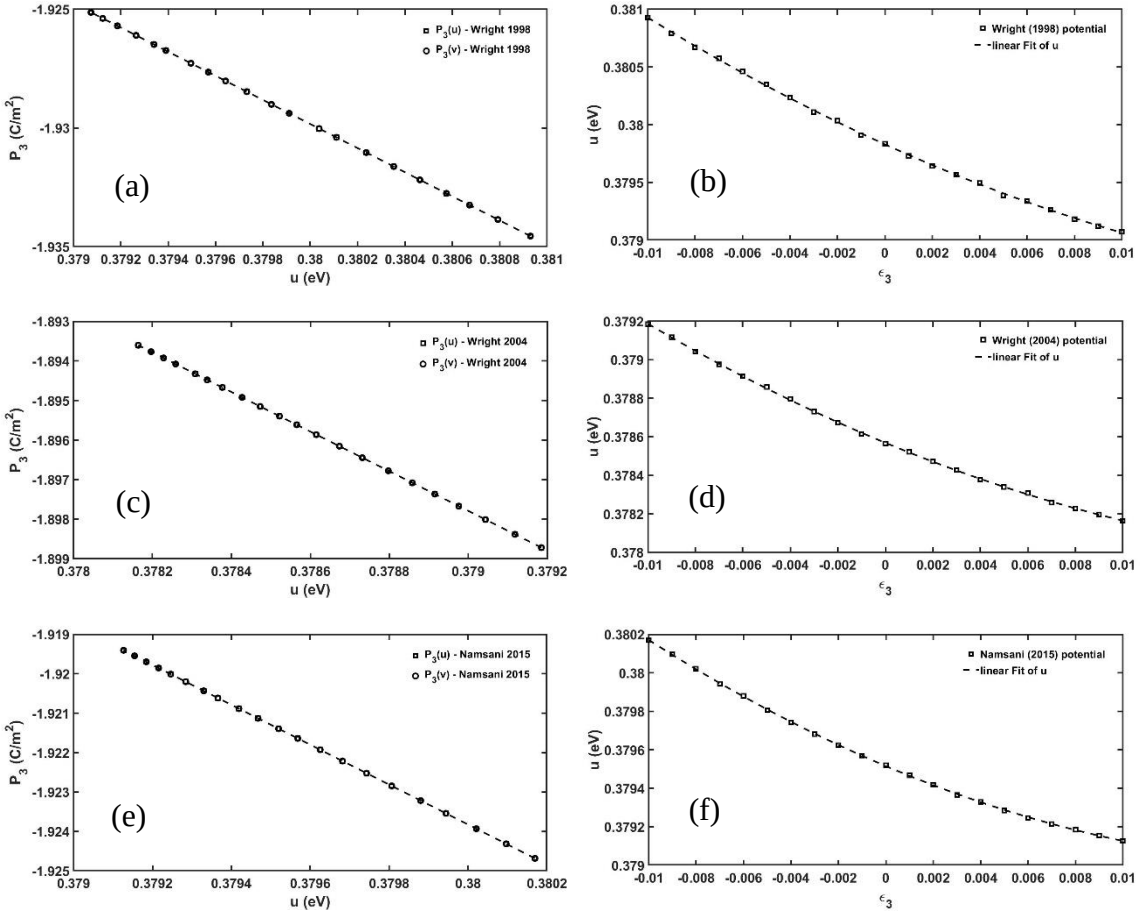


Figure 5. Plots utilized to calculate the proper piezoelectric constant e_{33}. (a) and (b) using PT1 potential; (c) and (d) using PT2 potential; (e) and (f) using PT3 potential. Left side: the polarization P_3 as a function of the fractional coordinate u, Right side: fractional coordinate u as a function of the strain ε_3 [21-23].

From the expression

$$\left.\frac{\partial P_3}{\partial u}\right|_{\varepsilon_3} = \frac{-4q}{\sqrt{3}{a_0}^2} \tag{21}$$

Which can be substituted in the e_{33}^{int} term in Equation (8) to obtain the expression

$$e_{33} = \frac{-4q}{\sqrt{3}{a_0}^2}\frac{\partial u(\varepsilon_3)}{\partial \varepsilon_3} \tag{22}$$

Figure 6. shows the graphs for the volume versus strain and P_3 versus strain. The improper piezoelectric constants e_{33} were obtained from the slops of the right plots by applying Equation (14). It can be seen from the plots on the right side of Figure 6. that the improper piezoelectric constants e_{33} values obtained using the volume average and the fraction coordinates utilizing the two expressions in Equation (12) are also in perfect agreement. In addition, as shown in Table 6 the proper and improper piezoelectric values are also in perfect agreement.

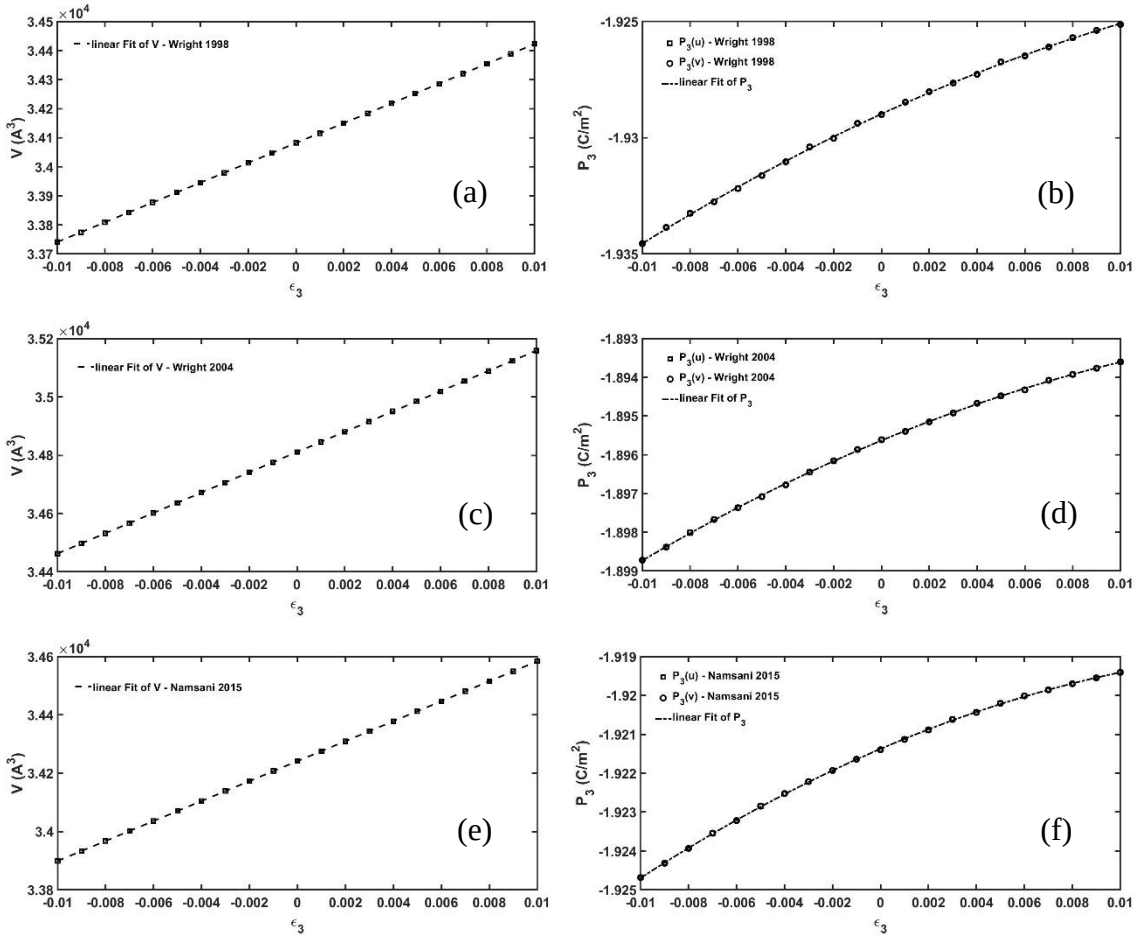


Figure 6. Plots utilized to calculate the improper piezoelectric constant e_{33}. (a) and (b) using PT1 potential; (c) and (d) using PT2 potential; (e) and (f) using PT3 potential. Left side: the volume as a function of strain ε_3., Right side: the polarization P_3 as a function of the strain ε_3 [21-23].

4.2 Bulk Zinc Sulfide Core-Shell Simulations

After validating our core-shell LAMMPS MD simulation code in the calculation of the piezoelectric constants e_{33} for the bulk ZnO, we utilized the code to conduct MD simulations using PT2 and PT3 core-shell potentials to calculate bulk ZnS piezoelectric constants.

Table 7 lists the results for the core-shell lattice constants, minimum energy polarization P_3, spontaneous polarization P_{eq}, and related data obtained using the PT2 and PT3 potentials. Our calculated lattice constants from the MD core-shell simulations are in good agreement with the

experiment values and in better agreement than those obtained by Catti et al. [20]. Our minimum energy polarization P_3 and spontaneous polarization P_{eq} results are also in excellent agreements with those obtain by Catti et al. [20]. Our $\frac{\partial P_3}{\partial u}$ values are also in excellent agreement with the DFT value of -5.034 obtained by Catti et al. [20]. In addition, the values for $\frac{\partial P_3}{\partial u}$ calculated from the slops of Figures 7(a) and 7(c) are in perfect agreement with the values obtained analytically using the Equation (21).

As with the bulk ZnO calculations, when the shell is considered, the $\frac{\partial u}{\partial \varepsilon_3}$ term in Equation (13) for both potentials became more negative due to increased relaxation effects, and thus became more accurate but underestimated as compared to the available DFT data. Since the internal strain component $\tilde{e}_{33}^{int}$ is the product $\frac{\partial P_3}{\partial u}\frac{\partial u}{\partial \varepsilon_3}$, the values for $\tilde{e}_{33}^{int}$ from calculation also became more accurate but underestimated as compared to the available DFT calculations [5].

In the case of the core-only calculations using Equation (11), the clamped ion term $e_{33}^{(0)} = 0$, and the piezoelectric constant e_{33} = $\tilde{e}_{33}^{\mathrm{int}}$ = $\frac{\partial \mathrm{P}_3}{\partial \mathrm{u}}\frac{\partial \mathrm{u}}{\partial \varepsilon_3}$ = 0.254 and 0.264 using the PT2 and PT3 potentials respectively. Since the calculated e_{33} values are in excellent agreement with the experimental value of 0.27, and the $\frac{\partial P_3}{\partial u}$ values are also in excellent agreement with the DFT data, we can conclude that the $\frac{\partial u}{\partial \varepsilon_3}$ values should be accurate [5].

In the case of core-shell calculations, the internal strain component $\tilde{e}_{33}^{int} = \frac{\partial P_3}{\partial u}\frac{\partial u}{\partial \varepsilon_3}$ values are underestimated in comparison to the DFT data, due to the fact that the $\frac{\partial u}{\partial \varepsilon_3}$ term are overestimated [5]. Since the $\frac{\partial u}{\partial \varepsilon_3}$ term depends on the fractional coordinate $u = \frac{|z(S)-z(Zn)|}{c}$ and the strain $\varepsilon_3 = \frac{(c-c_0)}{c_0}$, it can be concluded that the overestimation in the $\partial u/\partial \varepsilon_3$ values are due to

the fractional coordinate u term not being fully relaxed. This could be attributed to the spring constant K values used in Equation (16), for the harmonically coupled core-shell, not being fitted to the piezoelectric constant data for bulk ZnS. As a results of the incomplete relaxation of the fractional coordinate u term, it can be seen from Table 7 that the clamped ion term $e_{33}^{(0)}$ also tends to be underestimated compared with the DFT data. Since $e_{33} = e_{33}^{(0)} + e_{33}^{int}$, our calculated values for core-shell piezoelectric constants e_{33} are also underestimated in comparison with our core-only values and available DFT and experimental data.

Table 6. Summary of piezoelectric constants e_{33} calculated using classical interatomic potentials for bulk ZnS both with and without shell [22, 23].

Methods	**No shell**	**Shell**	**No shell**	**Shell**	**DFT**	**Exp.**
Potentials	**This work PT2**	**This work PT2**	**This work PT3**	**This work PT3**		
$\boldsymbol{a}$	3.8443	3.8445	3.8232	3.8237	3.982[c]	3.8227[f]
$\boldsymbol{c}$	6.2960	6.2965	6.2615	6.2623	6.500[c]	6.2607[f]
$\boldsymbol{u}$	0.3786	0.3775	0.3795	0.3789	0.377[c]	0.3748[f]
$\boldsymbol{P_3}$	-1.8956	-1.8898	-1.9214	-1.9180	-1.9092[c]	
$\boldsymbol{P_3^{theo}}$	-1.8980	-1.8980	-1.8980	-1.8980	-1.8980[c]	
$\boldsymbol{P_{eq} = P_3 - P_3^{theo}}$	0.0024	0.0082	-0.0234	-0.02	-0.0112[c]	
$\boldsymbol{\partial P_3/\partial u}$	-5.0074	-5.0067	-5.0627	-5.0613	-5.26[b], -5.034[c]	
$\boldsymbol{\partial u/\partial \varepsilon_3}$	-0.0512	-0.1055	-0.0522	-0.0731	-0.19[b], -0.147[c]	
$\tilde{e}_{33}^{int} = \frac{\partial P_3}{\partial u}\frac{\partial u}{\partial \varepsilon_3}$	0.256	0.528	0.264	0.369	1.01[b], 0.74[c]	
$e_{33}^{int} = \frac{\partial P_3}{\partial \varepsilon_3}$	0.256	0.528	0.264	0.369	1.01[b], 0.74[c]	
$\boldsymbol{e_{33}^{(0)}}$	0	-1.186	0	-1.29	-0.76[b], -0.59[c]	
$\boldsymbol{e_{33} = e_{33}^{(0)} + e_{33}^{int}}$	0.256	-0.6583	0.264	-0.9216	0.24[b], 0.15[c]	0.265[a],0.27[b] 0.34[c], 0.347[d] 0.43[e]

[a] Reference [40]., [b] Reference [5]., [c] Reference [9]., [d] Reference [41]., [e] Reference [42].

[f] Reference [43].

Figures 7. and 8. shows the plots resulting from MD simulations from which the proper $\tilde{e}_{33}^{int}$ and improper e_{33}^{int} values in Table 7 were obtained by performing linear fits to the

data in the two figures. It can be seen that the $P_3^{dis}\left(u(\varepsilon_3)\right)$ polarization values obtained using the fraction coordinates and the volume average utilizing Equation (12) are also in perfect agreement.

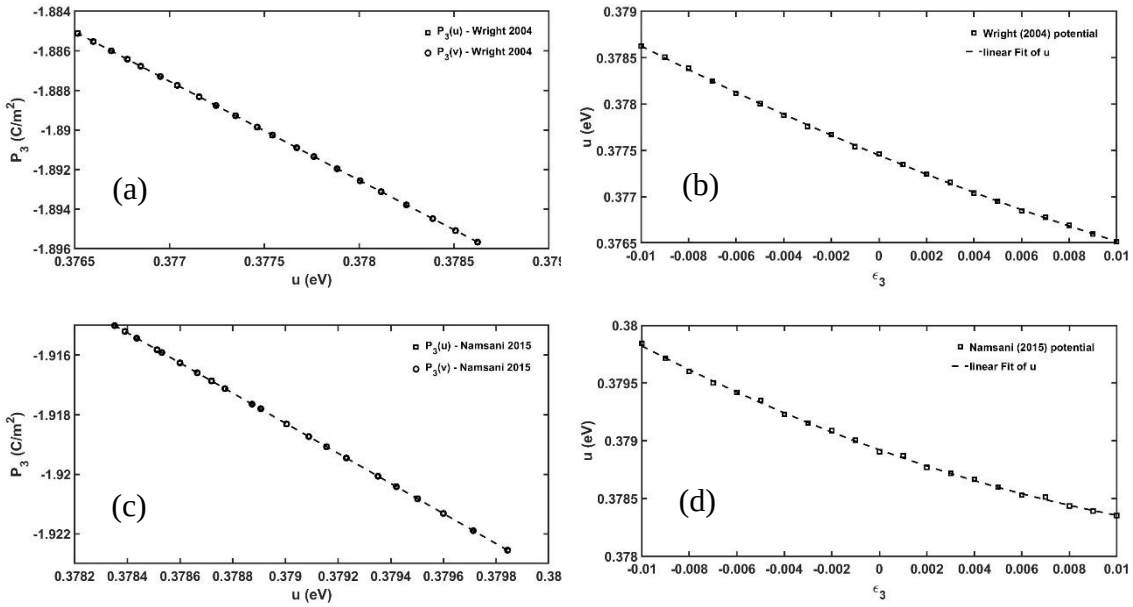


Figure 7. Plots utilized to calculate the proper piezoelectric constant e_{33}. (a) and (b) using PT2 potential; (c) and (d) using PT3 potential. Left side: the polarization P_3 as a function of the fractional coordinate u, Right side: fractional coordinate u as a function of the strain ε_3 [22, 23].

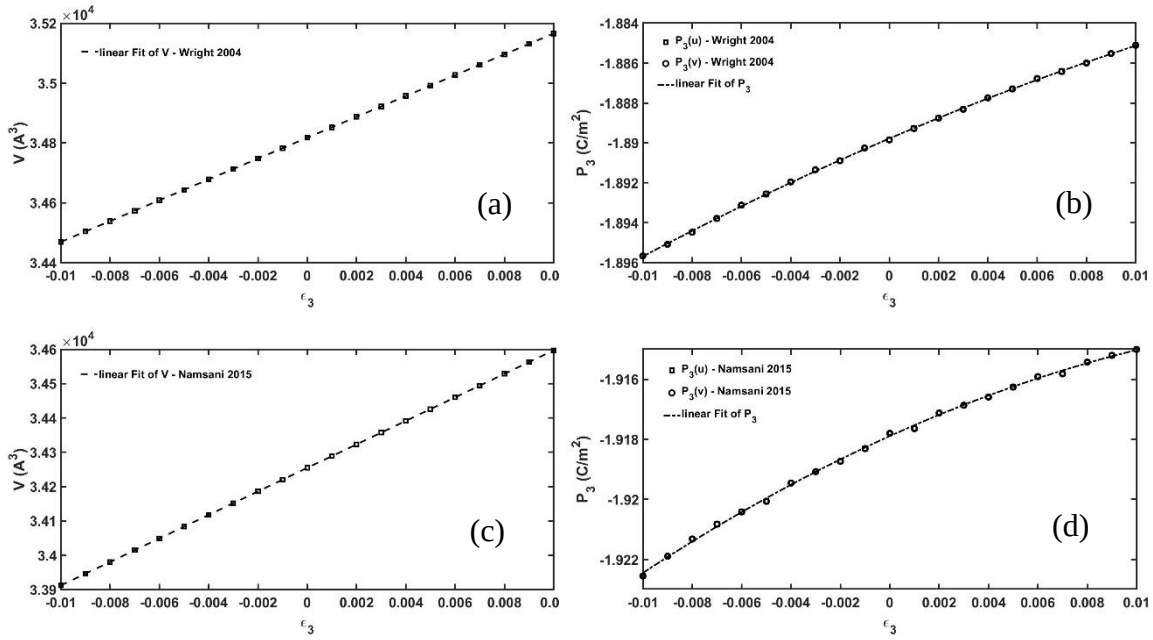


Figure 8. Plots utilized to calculate the improper piezoelectric constant e_{33}. (a) and (b) using PT2 potential; (c) and (d) using PT3 potential. Left side: the volume as a function of strain ε_3., Right side: the polarization P_3 as a function of the strain ε_3 [22, 23].

CHAPTER 5

ZINC OXIDE AND ZINC SULFIDE NANOBELTS

5.1 Zinc Oxide Nanobelts

Using classical MD simulation techniques and rigid ion approximations, Momeni et al. determined the size scale effect on the piezoelectric response of bulk ZnO and ZnO nanobelts [11]. Based on their study, they found that the piezoelectric coefficient of ZnO NBs decreased with increasing lateral dimensions, and converged to the bulk value as the lateral dimension approached 40 Å. In their MD simulations they neglected the presence of the shell polarization (i.e., rigid ion approximation) while modeling zinc-oxygen interaction using Binks et al. inter-atomic potential [37]. The Binks potential utilizes the Buckingham potential in Equation (1) with the parameters listed in Table 8. To validate our LAMMPS MD simulation code that will be utilized to calculate the piezoelectric coefficient of ZnS NBs, we utilized the code to calculate the piezoelectric coefficient of ZnO NBs, following the modeling procedure of Momeni et al. [11]. The ZnO NBs had a wurtzite crystalline structure with optimized lattice parameters as listed in Table 8. Each NBs structure was created with three

Table 7. Buckingham & Core-shell parameters with electric charges and Lattice parameters for the Binks et al. for ZnO [37].

Binks	**Species**	$\boldsymbol{A}$ **(eV)**	$\boldsymbol{\rho}$ (Å)	$\boldsymbol{C}$ **(eV** Å$^{\boldsymbol{6}}$**)**
	$O^{2-} - O^{2-}$	9547.96	0.21916	32.0
	$Zn^{2+} - O^{2-}$	529.70	0.3581	0.0
	$Zn^{2+} - Zn^{2+}$	0.0	0.0	0.0
Charges	**Zn core**	**O core**	**Zn shell**	**O shell**
	0.00	0.04	2.00	-2.04
Lattice Constants	$\boldsymbol{a}$ **(Å)**	$\boldsymbol{c}$ **(Å)**	$\boldsymbol{u = \frac{\lvert z(O) - z(Zn)\rvert}{c}}$	
	3.2495	5.2069	0.375	

separate regions: bottom, center, and top (see Figure 12). The coordinate locations along the longitudinal axis for the boundary atoms, belonging to the bottom and top regions, where within a 2.08 nm distance from both ends of the NBs and were assigned fixed motions for simulated axial deformation of the NBs. Initial equilibrium of the NB structures was achieved at 0.1 K for 300 ps. Momeni et al. indicated that the smallest ZnO NB (8.13 Å), which has a thickness below the critical thickness of 13 Å, underwent significant reconfiguration [11]. Their results are consistent with previous reports on the reconfiguration of ZnO nanowires (NWs) with diameters smaller than 13 Å [35, 45]. Following equilibration, the atoms at the bottom boundary remained fixed while the atoms at the upper boundary displaced by 0.151 Å in tension. According to the existing literatures, most of the stretching speeds range from 1m/s to 1000 m/s [7]. Therefore, the deformation process of the nanobelts was conducted in the z-axial direction with a strain rate of 10^8 s^{-1}, which corresponds to a stretching speed of 1.5 m/s. In Momeni et al. work, the effect of polarization between the nucleus and electron cloud was neglected, which resulted in elimination of the clamped ion term, i.e., $P_3^{elec}(\epsilon_3) = 0$ and $e_{ij}^{(0)} = 0$ [11]. The volume of the ZnO NBs was calculated as the volume of the continuum medium enclosing their structures, and the volume average polarization vector was calculated using Equation (12). The six ZnO NBs that were considered are listed in Table 9 along with their structure dimensions and total number of atoms. The polarization vectors as a function of the fractional coordinate for the six NB systems are shown Figure 9. The data points in the figure were derived from the MD simulations. The plot lines in the figure correspond to the least square fit to each data set. The data indicate that the NBs display linear piezoelectric behavior in the small strain regime. The piezoelectric coefficients were calculated from the slopes of the linear curve fits and are listed in Table 9.

From the table it can be seen that our results are in excellent agreement with piezoelectric coefficients value obtained by Momeni et al. [11].

Table 8. ZnO Nanobelts dimensions along different direction, number of atoms, and piezoelectric coefficient using the Binks et al. potential [37].

Index	Dimensions (Å)			No. of atoms	$\tilde{e}_{33} = e_{33}$
	x_1 (a)	x_2 (b)	x_3 (c)		(C m^{-2})
NB 1	6.50	9.38	151	2436	2.7128
NB 2	16.25	15.01	151	3828	2.4158
NB 3	19.50	20.64	151	6032	2.1542
NB 4	22.75	26.27	151	8700	2.0124
NB 5	29.25	26.27	151	11020	1.9703
NB 6	32.50	31.90	151	14616	1.9097

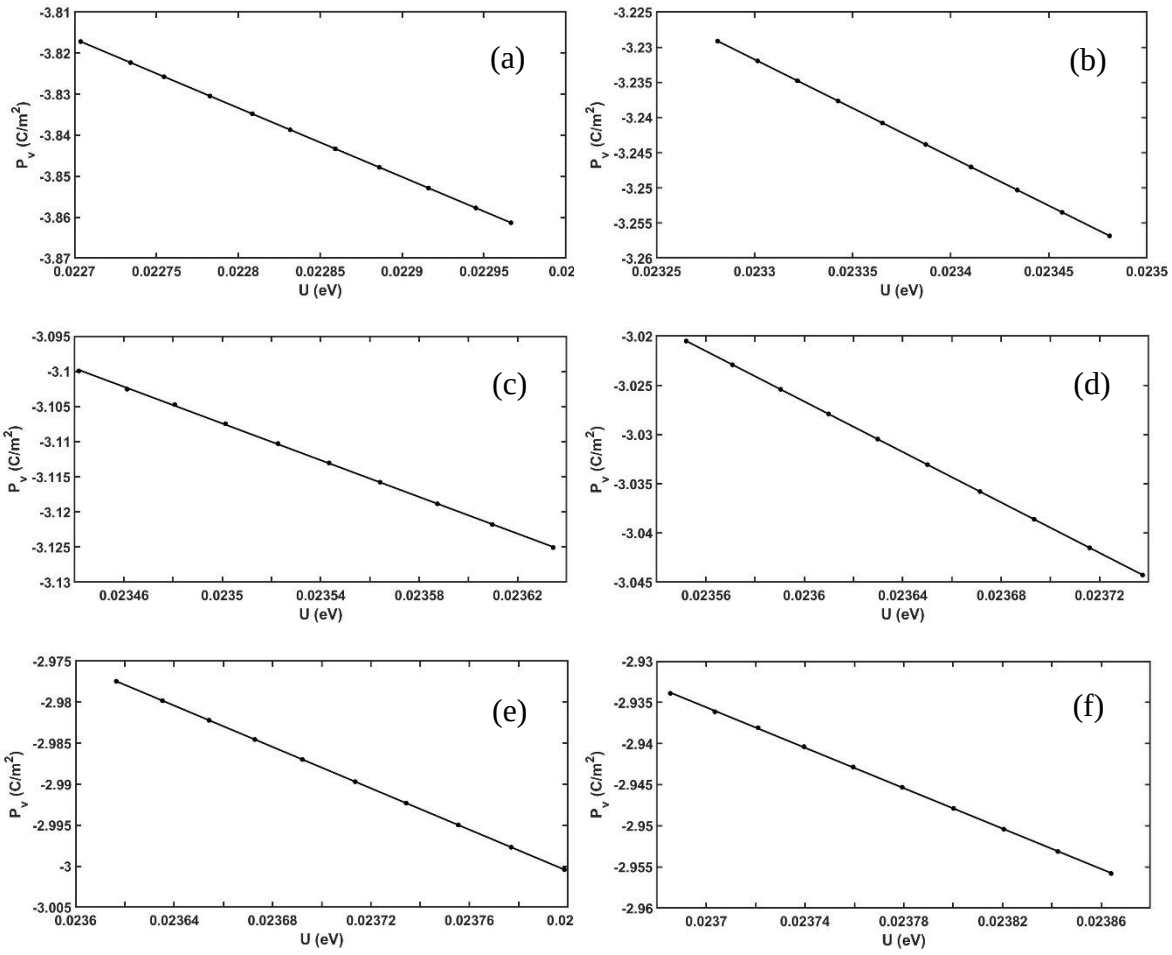


Figure 9. The average polarization vectors (∂Pv) vs. u calculated along the z-axis for ZnO NBs with lateral dimensions (a) 6.50 Å, (b) 16.25 Å, (c) 19.50 Å, (d) 22.75 Å, (e) 29.25 Å, and (f) 32.50 Å.

Figure 10 shows the fractional coordinate u as a function of the strain ε_3 calculated along the z-axis for six NBs with lateral dimensions.

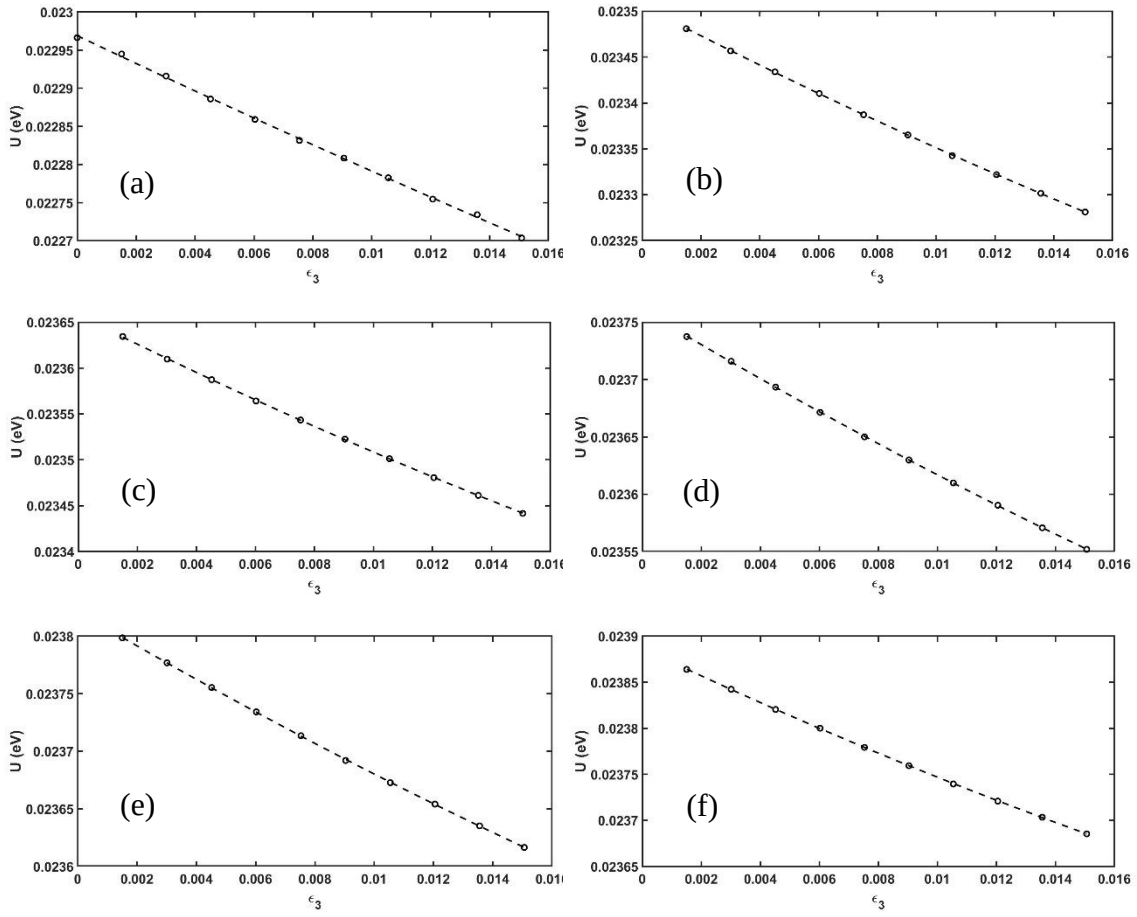


Figure 10. The fractional coordinate u as a function of the strain ε_3 calculated along the z-axis for ZnO NBs with lateral dimensions (a) 6.50 Å, (b) 16.25 Å, (c) 19.50 Å, (d) 22.75 Å, (e) 29.25 Å, and (f) 32.50 Å.

The piezoelectric coefficients from our work and those from Momeni et al. are plotted in Figure 11 for ZnO NBs with six lateral dimensions [11]. The plot also shows the calculated piezoelectric coefficient of bulk ZnO along the [0001] crystallographic direction: 1.27 C/m^2 [6], which is reasonably close to the experimental value from literature (1–1.55 C/m^2) [11]. From the figure, it is clear that the piezoelectric constant values for our ZnO NBs decreases with increasing size, from 2.7128 C/m^2 for the 6.50 Å NB to 1.9097 C/m^2 for the 32.50 Å NB. Our result for the 6.50 Å ZnO NB is about 53%. larger than the values predicted for bulk ZnO. These results are in excellent agreement with the piezoelectric coefficient values for the ZnO NBs from

Momeni et al. work [11], which also decreases with increasing size, from 2.322 C/m^2 for the 8.13 Å NB to 1.639 C/m^2 for the 37.37 Å NB as can be seen in Figure 11.

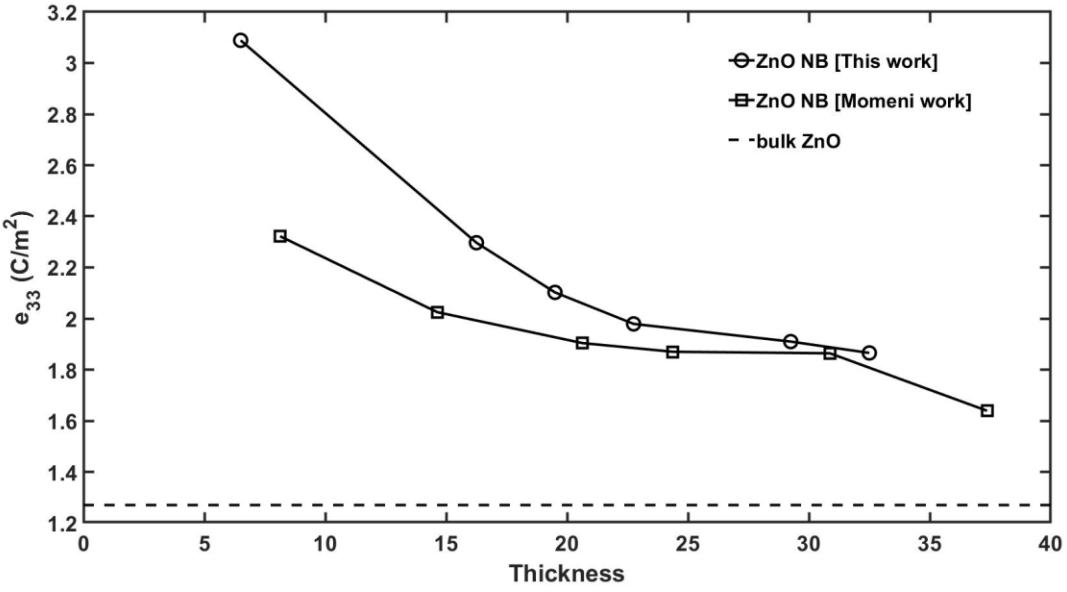


Figure 11. Piezoelectric coefficient of ZnO as a function of the NB lateral dimension. The dashed line shows the piezoelectric coefficient of bulk ZnO calculated using a classical MD approach and rigid ion approximation [11].

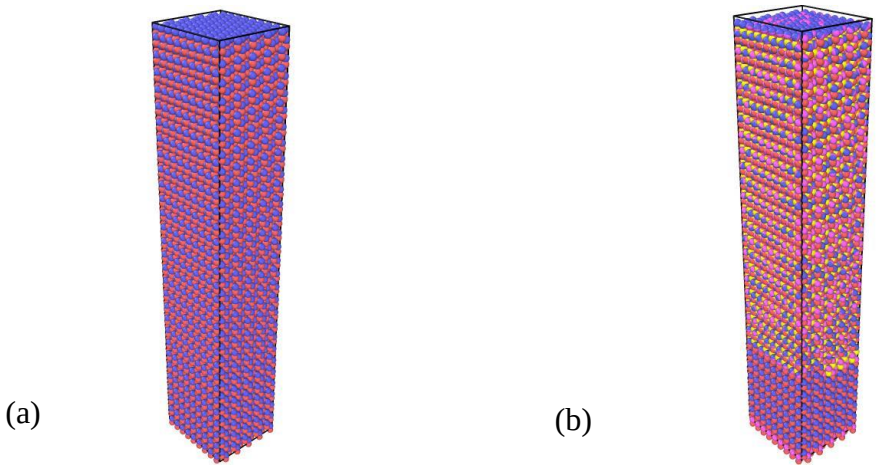


Figure 12. 6x6x6 ZnO Nanobelts structure for (a) core-only and (b) core-shell.

5.2 Zinc Sulfide Nanobelts

After validating our LAMMPS MD simulation code by calculate the piezoelectric coefficient for the six ZnO NBs, we run MD simulations to calculate the piezoelectric coefficient for six ZnS NB structures. The ZnS NBs were created with wurtzite crystalline structures with the experimental lattice parameters listed in Table 1. During our MD simulations, the NB structures maintained better structural stability when using the PT2 potential than when utilizing the PT3 potential, possibly due to the inclusion of the four-body bonded term in Equation (3). Therefore, in our NB simulations we only utilized the PT2 potential. Wang et al. found that the structural stability field for wurtzite ZnS nanobelts was below the critical thickness of 74 Å [25]. In our simulations, all wurtzite ZnS NBs, sizes were less than 43 Å, which indicates that the created structures are within the stable field. The six ZnS NB structure sizes that were investigated in this work and their corresponding number of atoms, are listed in Table 10.

Table 9. ZnS Nanobelts dimensions along different directions, number of atoms, and calculated piezoelectric coefficient using PT2 potential [22].

Index	Dimensions (Å)			No. of atoms	$\tilde{e}_{33} = e_{33}$
	x_1 (a)	x_2 (b)	x_3 (c)		(C m^{-2})
NB 1	22.94	24.28	91.75	3068	0.6038
NB 2	26.76	30.90	91.75	4425	0.4834
NB 3	30.59	37.53	91.75	6018	0.4355
NB 4	38.23	37.53	91.75	7434	0.3599
NB 5	34.41	44.15	91.75	7847	0.3545
NB 6	42.06	57.39	91.75	12213	0.2732

As with ZnO NBs, after equilibration the bottom boundary atoms were fixed and the upper ones were displaced 0.091 Å in tension. Figure 13. shows the polarization as a function of strain for the six ZnS NBs with the plot lines correspond to least squares fits to each data set. The slope of each line gives the corresponding piezoelectric coefficient for that particular ZnS NBs.

The average polarization vector was calculated considering the volume of the NB as the sum of the volumes of individual atoms.

In Figure 14. the piezoelectric coefficients are plotted as a function of NB size. From the figure it is clear that as with ZnO NBs the piezoelectric constant values for ZnS NBs also decreases with increasing size, from 0.6038 C/m^2 for the 22.94 Å NB to 0.2732 C/m^2 for the 42.06 Å NB. Our results of 22.94 Å ZnS NB is almost 58%. larger than the 0.256 C/m^2 value calculated for bulk ZnS plotted as dashed line in Figure 14.

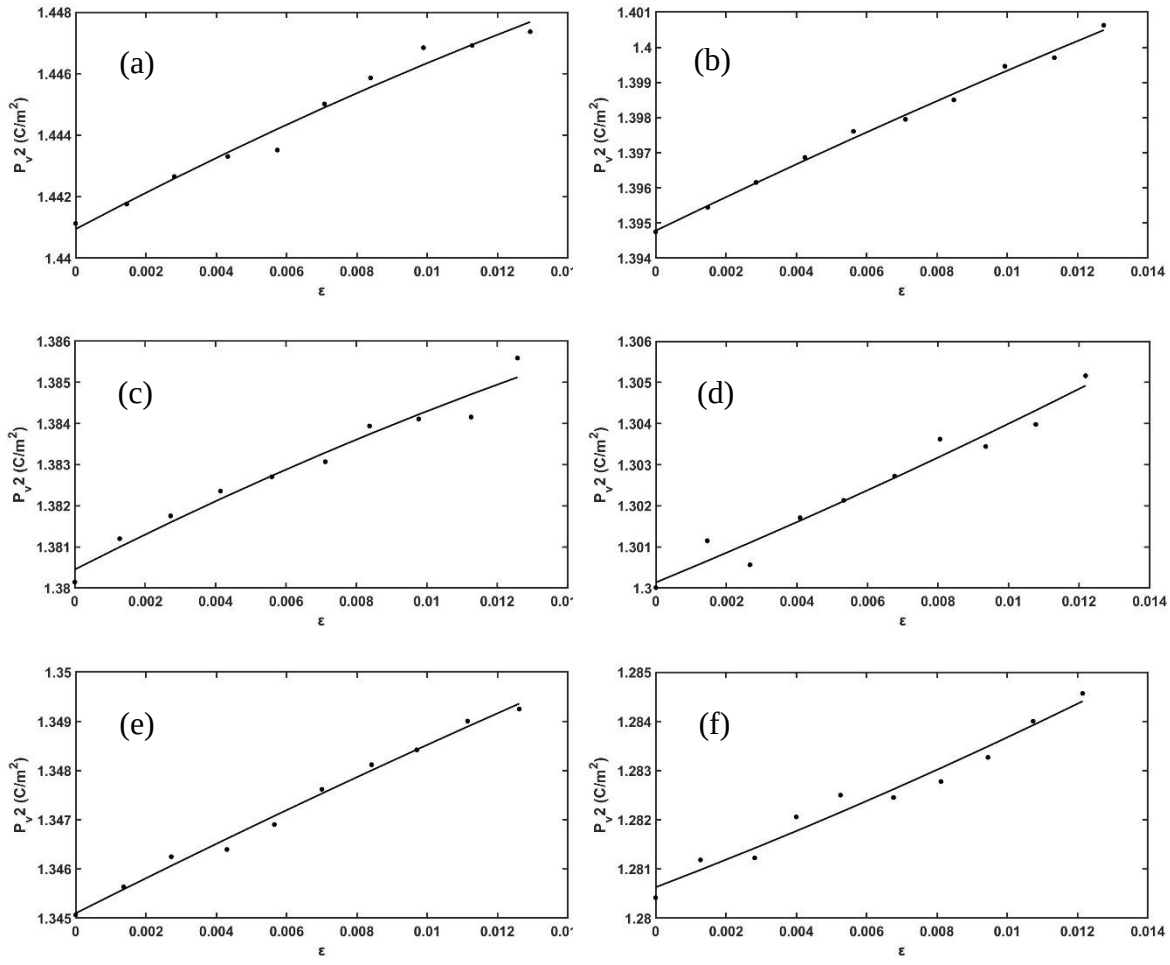


Figure 13. The average polarization vectors (∂Pv) vs. strain calculated along the z-axis for ZnS NBs with lateral dimensions (a) 22.94 Å, (b) 26.76 Å, (c) 30.59 Å, (d) 38.23 Å, (e) 34.41 Å, and (f) 42.06 Å.

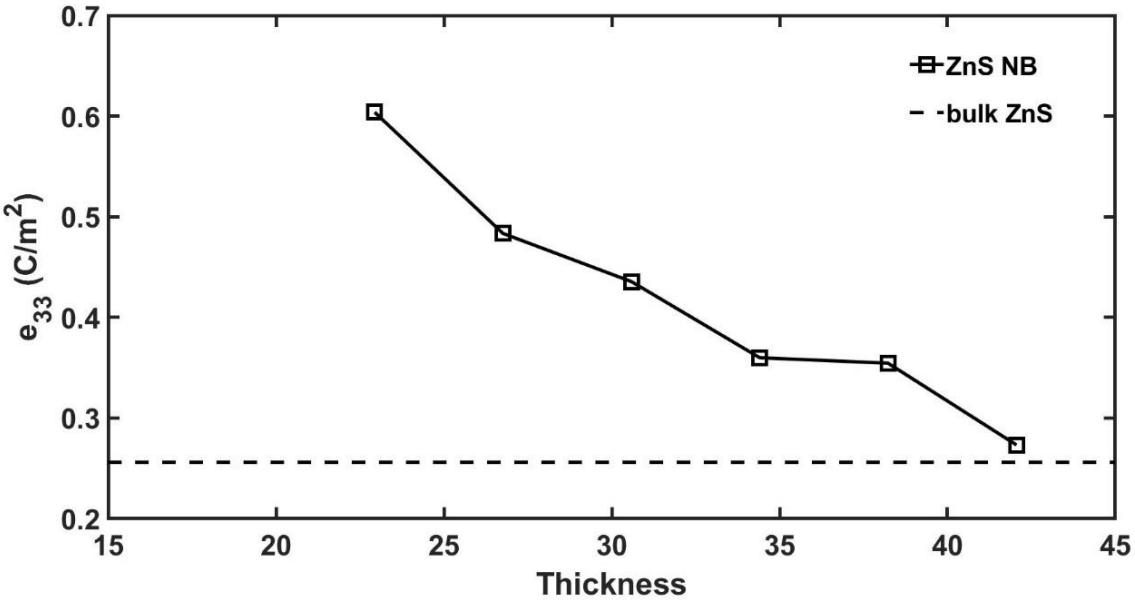


Figure 14. Piezoelectric coefficient of ZnS as a function of the NB lateral dimension. The dashed line shows the piezoelectric coefficient of bulk ZnS calculated using a classical MD approach and rigid ion approximation.

CHAPTER 6

CONCLUSION

In this thesis we performed molecular dynamics simulations using the open source LAMMPS software to calculate the piezoelectric constants for bulk zinc sulfides and nanobelts structures utilizing three core-shell interatomic ZnS potentials. The reliability of the three potentials was first tested by calculating the bulk modulus for the bulk ZnS using core-only and core-shell MD simulations. The bulk modulus calculated values using PT2 and PT3 potentials were in excellent agreement (70.91-74.08 GPa) with the experimental value of 74.0 GPa, and the largest deviation resulted from using the PT1 potential. The results from MD simulations for the bulk ZnS core-only also showed that the polarization P_3 (-1.8956, -1.9214 C/m^2), the spontaneous polarization P_{eq} (0.0024, -0.0234 C/m^2), and the proper and improper e_{33} piezoelectric constants (0.256, 0.264 C/m^2) are obtained using the PT2 and PT3 potentials with the Ewald summation method are in excellent agreement with the available DFT and experimental data, and the largest deviation from experimental value was also obtained using the PT1 potential. Our calculated values for the P_3, P_{eq} from the core-shell simulations were also excellent agreement with available DFT and experiment data, however the e_{33} piezoelectric constant values were underestimated in comparison with our e_{33} core-only values and the DFT and experimental data. The underestimation of the e_{33} core-shell values can be attributed to the spring constant K values for the harmonically coupled core-shell not being fitted to the piezoelectric constant data for bulk ZnS. For the ZnS NBs MD simulations, utilizing the PT2 potential resulted in better structural stability than that using the PT1 and PT3 potentials, which might be due to the inclusion of the four-body bonded term (torsional) potential. The ZnS NBs core-only MD simulation results utilizing the PT2 potential showed that piezoelectric constant

values decrease with increasing lateral size (0.6038-0.2732 C/m^2 for 22.94-57.39 Å). The result for 22.94 Å ZnS NB is almost 58%. larger than the 0.256 C/m^2 value calculated for bulk ZnS. These results are consistent with those obtained for ZnO NBs. In conclusion, utilizing the PT2 potential developed by Wright and Gale (2004), in core-only MD simulation was the most reliable potential in predicting the lattice constants, minimum energy polarization P_3, spontaneous polarization P_{eq}, and the piezoelectric constant e_{33} for both bulk and nanobelts ZnS structure.

REFERENCE

[1] Agrawal, R., Peng, B., Gdoutos, E. E., and Espinosa, H. D., 2008, "Elasticity Size Effects in ZnO Nanowires-A Combined Experimental-Computational Approach," *Nano Letters,* 8(11), pp. 3668-3674. https://pubs.acs.org/doi/10.1021/nl801724b.

[2] Zhao, M., Wang, Z., and Mao, S. X., 2004, "Piezoelectric Characterization of Individual Zinc Oxide Nanobelt Probed by Piezoresponse Force Microscope," *Nano Letters*, 4(4), pp. 587-590.

[3] Noel, Y., Llunell, M., Orlando, R., D'Arco, P., and Dovesi, R., 2002, "Performance of Various Hamiltonians in The Study of The Piezoelectric Properties of Crystalline Compounds: The Case of BeO and ZnO," *Physical Review B,* 66, pp. 214107. https://journals.aps.org/prb/abstract/10.1103/PhysRevB.66.214107.

[4] Serrà, A., Artal, R., García-Amorós, J., Sepúlveda, B., Gómez, E., Nogués, J., and Philippe, L., 2020, "Hybrid Ni@ZnO@ZnS-Microalgae for Circular Economy: A Smart Route to The Efficient Integration of Solar Photocatalytic Water Decontamination and Bioethanol Production," *Advanced Science,* 7, pp. 1902447. https://doi.org/10.1002/advs.201902447.

[5] Corso, A. D., Posternak, M., Resta, R., and Baldereschi, A., 1994, "Ab Initio Study of Piezoelectricity and Spontaneous Polarization in ZnO," *Physical Review B,* 50(15), pp. 10715-10721.

[6] Dai, S., Dunn, M. L., and Park, H. S., 2010, "Piezoelectric Constants for ZnO Calculated Using Classical Polarizable Core–Shell Potentials," *Nanotechnology,* 21, pp. 445707. https://iopscience.iop.org/article/10.1088/0957-4484/21/44/445707.

[7] Hill, N. A., and Waghmare, U., 2000, "First-Principles Study of Strain-Electronic Interplay in ZnO: Stress and Temperature Dependence of The Piezoelectric Constants," *Physical Review B,* 62(13), pp. 8802-8810.

[8] Cinthia, J., A., Sudhapriyanga, G., Rajeswarapalanichamy, R., and Santhosh, M., 2014, "Structural, Electronic and Elastic Properties of ZnO and CdO: A First-Principles Study," *Procedia Materials Science,* 5, pp. 1034-1042. https://doi.org/10.1016/j.mspro.2014.07.394.

[9] Kalay, M., Kart, H. H., Kart, S. Ö., and Çağın, T., 2009, "Elastic Properties and Pressure Induced Transitions of ZnO Polymorphs from First-Principle Calculations," *Journal of Alloys and Compounds,* 484(1), pp. 431-438. https://doi.org/10.1016/j.jallcom.2009.04.116.

[10] Lee, W. J., Chang, J. G., Ju, S. P., Weng, M. H., and Lee, C. H., 2011, "Structure-Dependent Mechanical Properties of Ultrathin Zinc Oxide Nanowires," *Nanoscale Research Letters,* 6, pp. 352. https://doi.org/10.1186/1556-276X-6-352.

[11] Momeni, K., Odegard, G. M., and Yassar, R. S., 2012, "Finite Size Effect on The Piezoelectric Properties of ZnO Nanobelts: A Molecular Dynamics Approach," *Acta Materialia,* 60, pp. 5117-5124. https://doi.org/10.1016/j.actamat.2012.06.041.

[12] Wang, S., Fan, Z., Koster, R. S., Fang, C., Huis, M. A., Yalcin, A. O., and Vlugt, T. J., 2014, "New Ab Initio Based Pair Potential for Accurate Simulation of Phase Transitions in ZnO," *Journal of Physical Chemistry,* 118, pp. 11050-11061.

[13] Wang, W., Pi, Z., Lei, F., and Lu, Y., 2016, "Understanding the Tensile Behaviors of Ultra-Thin ZnO Nanowires Via Molecular Dynamics Simulations," *AIP Advance,* 6, pp. 035111. https://doi.org/10.1063/1.4944499.

[14] Fernandes, C. D., Ferrer, M. M., Raubach, C. W., Moreira, E. C., Gularte, L. T., Cava, S., Piotrowski, M. J., Jardim, P. L. G., Carvalho, R. D., and Moreira, M. L., 2020, "An

Investigation of The Photovoltaic Parameters of ZnS Grown on ZnO (1011)," *New Journal Chemistry*, 44, pp. 20600. https://doi.org/10.1039/D0NJ041.

[15] Bera, B., and Sarkar, M. D., 2016, "Piezoelectric Effect, Piezotronics and Piezophototronics: A Review," *Journal of Interdisciplinary Research*, 2(11), pp. 1407-1410.

[16] Wang, X., Li, J., Zhang, W., Guo, S., Zhang, Y., Zou, B., and Liu, R., 2014, "Synthesis and Characterization of Zinc Sulfide Nanobelts with Periodically Modulated Thickness," *Materials Letters*, 132(1), pp. 224-227. https://doi.org/10.1016/j.matlet.2014.06.102.

[17] Li, X., Wang, X., Xiong, Q., and Eklund, P. C., 2005, "Mechanical Properties of ZnS Nanobelts," *Nano Letters*, 5(10), pp. 1982-1986.

[18] Ferahtia, S., Saib, S., Bouarissa, N., and Benyettou, S., 2014, "Structural Parameters, Elastic Properties and Piezoelectric Constants of Wurtzite ZnS and ZnSe Under Pressure," *Superlattices and Microstructures*, 67, pp. 88-96. https://doi.org/10.1016/j.spmi.2013.12.021.

[19] Wright, K., Watson, G. W., Parker, S. C., and Vaughan, D. J., 1998, "Simulation of The Structure and Stability of Sphalerite (ZnS) Surfaces," *American Mineralogist*, 83, pp. 141-146.

[20] Catti, M., Noel, Y., and Dovesi, R., 2003, "Full Piezoelectric Tensors of Wurtzite and Zinc Blende ZnO and ZnS by First-Principles Calculations," *Journal of Physics and Chemistry of Solids*, 64(11), pp. 2183-2190. https://doi.org/10.1016/S0022-3697(03)00219-1.

[21] Wright, K., and Jackson, R. A., 1995, "Computer Simulation of The Structure and Defect Properties of Zinc Sulfide," *Journal of Materials Chemistry*, 5(11), pp. 2037-3040.

[22] Wright, K., and Gale, J. D., 2004, "Interatomic Potentials for The Simulation of The Zinc-Blende and Wurtzite Forms of ZnS and CdS: Bulk Structure, Properties, and Phase Stability. *Physical Review B*, 70, pp. 035211. https://doi.org/10.1103/PhysRevB.70.035211.

[23] Namsani, S., Nair, N. N., and Singh, J. K., 2015, "Interaction Potential Models for Bulk ZnS, ZnS Nanoparticle, and ZnS Nanoparticle-PMMA from First-Principles," *Journal of Computational Chemistry*, 36, pp. 1176-1186.

[24] Güler, E., and Güler, M., 2015, "A Theoretical Investigation Oof The Effect of Pressure on The Structural, Elastic and Mechanical Properties of ZnS Crystals," *Brazilian Journal of Physics*, 45, pp. 296-301. https://doi.org/10.1007/s13538-015-0320-4.

[25] Wang, Z., Daemen, L. L., Zhao, Y., Zha, C. S., Downs, R. T., Wang, X., Wang, Z. L., and Hemley, R. J., 2005, "Morphology-Tuned Wurtzite-Type ZnS Nanobelts," *Nature Materials*, 4, pp. 922-927.

[26] Lopez-Garcia, A. J., Bauer, A., Rubio, R. F., Payno, D., Li-Kao, Z. J., Kazim, S., Hariskos, D., Izquierdo-Roca, V., Saucedo, E., and Pérez-Rodríguez, A., 2020, "UV-Selective Optically Transparent Zn(O,S)-Based Solar Cells," *Advanced Science News,* 4, pp. 2000470.

[27] Zheng, E., Wang, Y., Song, J., Wang, X. F., Tian, W., Chen, G., and Miyasaka, T., 2018, "ZnO/ZnS Core-Shell Composites for Low-Temperature-Processed Perovskite Solar Cells," *Journal of Energy Chemistry*, 27(5), pp. 1461-1467. https://doi.org/10.1016/j.jechem.2017.09.026.

[28] Hamad, S., Cristol, S., and Catlow, C. R., 2002, "Surface Structures and Crystal Morphology of ZnS: Computational Study," *Journal of Physical Chemistry B,* 106, pp. 11002-11008.

[29] Grünwald, M., Zayak, A., Neaton, J. B., Geissler, P. L., and Rabani, E., 2012, "Transferable Pair Potentials for CdS and ZnS Crystals," *Journal of Chemical Physics* 136, pp. 234111. https://doi.org/10.1063/1.4729468.

[30] Benkabou, F., Aourag, H., and Certier, M., 2000, "Atomistic Study of Zinc-Blende CdS, CdSe, ZnS, and ZnSe from Molecular Dynamics," *Materials Chemistry and Physics,* 66(1), pp. 10-16.

[31] Güler, M., and Güler, E., 2017, "Theoretical Analysis of Elastic, Mechanical and Phonon Properties of Wurtzite Zinc Sulfide Under Pressure," *Crystals*, 7(6), pp. 161. https://doi.org/10.3390/cryst7060161.

[32] Bilge, M., Kart, S. Ö., Kart, H. H., and Cagin, T., 2008, "Mechanical and Electronical Properties of ZnS Under Pressure," *Journal of Achievements in Materials and Manufacturing Engineering,* 31(1), pp. 29-34.

[33] Vashishta, P., Kalia, R. K., and Nakano, A., 2003, "Multimillion Atom Molecular Dynamics Simulations of Nanostructures on Parallel Computers," *Journal of Nanoparticle Research,* 5, pp. 119-135. https://doi.org/10.1023/A:1024459800821.

[34] Khalkhali, M., Liu, Q., and Zhang, H., 2014, "A Comparison of Different Empirical Potentials in ZnS," *Modelling and Simulation in Materials Science and Engineering,* 22, pp. 085014. http://dx.doi.org/10.1088/0965-0393/22/8/085014.

[35] Khalkhali, M., Liu, Q., Zeng, H., and Zhang, H., 2015, "A Size-Dependent Structural Evolution of ZnS Nanoparticles," *Scientific Reports,* 5, pp. 14267. https://doi.org/10.1038/srep14267.

[36] Pan, Y., Qu, S., Dong, S., Cui, Q., Zhang, W., Liu, X., Liu, J., Liu, B., Gao, C., and Zou, G., 2002, "An Investigation on The Pressure-Induced Phase Transition of Nanocrystalline ZnS," *Journal of Physics: Condensed Matter,* 14, pp. 10487-10490.

[37] Binks, D. J., and Grimes, R. W., 1994, "The Non-Stoichiometry of Zinc and Chromium Excess Zinc Chromite," *Solid State Communications,* 89(11), pp. 921-924. https://doi.org/10.1016/0038-1098(94)90351-4.

[38] Nyberg, M., Nygren, M. A., Pettersson, G. M., Gray, D. H., and Rohl, A. L., 1996, "Hydrogen Dissociation on Reconstructed ZnO Surfaces," *Journal of Physical Chemistry,* 100, pp. 9054-9063.

[39] Wolf, D., Keblinski, P., Phillpot, S. R., and Eggebrecht, J., 1999, "Exact Method for The Simulation of Coulombic Systems by Spherically Truncated Pairwise r-1 Summation," *Journal of Chemical Physics,* 110, pp. 8254.

[40] Dan, k. I. A., Kobyakov, I. B., and Dabydov, S. Yu., 1982, *Soviet Physics Solid State*, 24, pp. 2058.

[41] Zhdanov, V. A. and Brysneva, L. A., 1974, *Soviet Physics Crystallography*, 19, pp. 223.

[42] Firsova, M. M., 1974, *Soviet Physics Solid State,* 16, pp. 350.

[43] Kisi, E. H. and Elcombe, M. M., 1989, "u Parameters for The Wurtzite Structure of ZnS and ZnO Using Powder Neutron Diffraction," *Acta Crystallographica*, 45, pp. 1867-1870.

[44] Nyberg, M., Nygren, M. A., and Pettersson, L. G., 1996, "Hydrogen Dissociation on Reconstructed ZnO Surfaces," *Journal of Physical Chemistry,* 100, pp. 9054-9063.

[45] Zhang, L. and Huang, H., 2007, "Structural Transformation of ZnO Nanostructures," *Applied Physics Letters*, 90, pp. 023115.

[46] De K. J., 1967, "Elastic Constants of α-ZnS," *Journal Physics Chemistry Solids*, 28, pp. 1831-1837.

[47] Desgreniers, S., Beaulieu, L., and Lepage, I., 2000, "Pressure-Induced Structural Changes in ZnS," *Physical Review B,* 61, pp. 8726-8733.

[48] Cline, C. F., 1967, "Elastic Constants of Hexagonal BeO, ZnS, and CdSe," *Journal Applied Physics*, 38, pp. 1944.

[49] Uchida, N., 1972, "Elastic and Photoelastic Constants of α-ZnS," *Journal Applied Physics*, 43, pp. 971.

[50] Haynes, W. M., Lide, D. R., and Bruno, T. J., 2013, *CRC handbook of chemistry and physics, 94th edition.* Boca Raton, FL: CRC Press.

[51] Sumit, W., Forester, T. R., Todorov, I. T., and Leslie, M., 2013, "The DLPOLY Molecular Simulation Package," Daresbury, Warrington, U.K. Retrieved Sep 24, 2013, http://www.ccp5.ac.uk/DL_POLY/.

APPENDIX A:

INSTITUTIONAL REVIEW BOARD LETTER

MARSHALL UNIVERSITY
www.marshall.edu

Office of Research Integrity

September 13, 2021

Rui Xie
1050 Enclave Blvd., Apt 304 A
Edwardsville, IL 62025

Dear Rui:

This letter is in response to the submitted thesis abstract entitled *"A Molecular Dynamic Study of the Piezoelectric Properties for Bulk ZnS and Nanobelts."* After assessing the abstract, it has been deemed not to be human subject research and therefore exempt from oversight of the Marshall University Institutional Review Board (IRB). The Code of Federal Regulations (45CFR46) has set forth the criteria utilized in making this determination. Since the information in this study does not involve human subjects as defined in the above referenced instruction, it is not considered human subject research. If there are any changes to the abstract you provided then you would need to resubmit that information to the Office of Research Integrity for review and a determination.

I appreciate your willingness to submit the abstract for determination. Please feel free to contact the Office of Research Integrity if you have any questions regarding future protocols that may require IRB review.

Sincerely,

Bruce F. Day, ThD, CIP
Director

APPENDIX B:

CODE USED TO CALCULATE DATA IN THIS THESIS

SCREENED HARMONIC ANGLE POTENTIAL DERIVATION

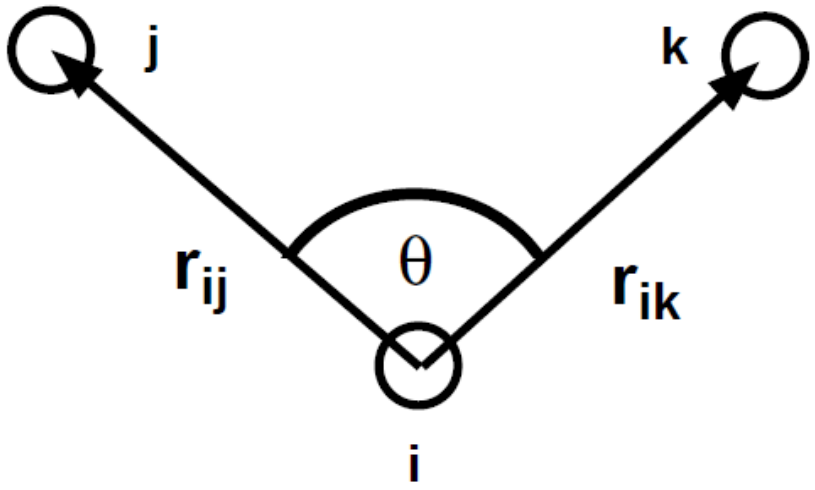


Figure 15. The screened harmonic angle and associated vectors.

In Figure 15. the screened harmonic angle potentials describe the bond bending terms between the specified atoms [51]. In DL POLY Classic the most general form for the screened harmonic angle potentials can be written as:

$$U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = A(\theta_{jik})S(r_{ij})S(r_{ik}) \quad (1)$$

where $A(\theta_{jik})$ is a purely angular function and $S(r)$is a screening or truncation function. All the function arguments are scalars.

$$A(\theta_{jik}) = k(\theta_{jik} - \theta_0)^2 \quad (2)$$

$$S(r_{ij}) = exp\left(-\frac{r_{ij}}{\rho_1}\right) \quad (3)$$

$$S(r_{ik}) = exp\left(-\frac{r_{ik}}{\rho_2}\right) \quad (4)$$

$$\frac{\partial A(\theta_{jik})}{\partial \theta_{jik}} = 2k\left(\theta_{jik} - \theta_0\right) \tag{5}$$

$$\frac{\partial S(r_{ij})}{\partial r_{ij}} = -\frac{1}{\rho_1} exp\left(-\frac{r_{ij}}{\rho_1}\right) \tag{6}$$

$$\frac{\partial S(r_{ik})}{\partial r_{ik}} = -\frac{1}{\rho_2} exp\left(-\frac{r_{ik}}{\rho_2}\right) \tag{7}$$

$$-\frac{\partial U(\theta_{jik} \cdot r_{ij} \cdot r_{ik})}{\partial_{r_\ell^\alpha}} = -\frac{\partial}{\partial_{r_\ell^\alpha}} A\left(\theta_{jik}\right) \tag{8}$$

$$-\frac{\partial A(\theta_{jik})}{\partial_{r_\ell^\alpha}} = \left\{\frac{1}{\sin(\theta_{jik})}\right\} \frac{\partial}{\partial \theta_{jik}} A\left(\theta_{jik}\right) \frac{\partial}{\partial_{r_\ell^\alpha}} \left\{\frac{\underline{r}_{ij} \cdot \underline{r}_{ik}}{r_{ij} r_{ik}}\right\}$$

With atomic label ℓ being one of i, j, k and α indicating the x, y, z component. The derivative is

$$-\frac{\partial}{\partial_{r_\ell^\alpha}} U\left(\theta_{jik} \cdot r_{ij} \cdot r_{ik}\right) =$$

$$-S\left(r_{ij}\right) S(r_{ik}) \frac{\partial}{\partial_{r_\ell^\alpha}} A\left(\theta_{jik}\right) - A\left(\theta_{jik}\right) S(r_{ik}) \left(\delta_{\ell j} - \delta_{\ell i}\right) \frac{r_{ij}^\alpha}{r_{ij}} \frac{\partial}{\partial_{r_{ij}}} S\left(r_{ij}\right) - A\left(\theta_{jik}\right) S\left(r_{ij}\right) (\delta_{\ell k} -$$

$$\delta_{\ell i}) \frac{r_{ik}^\alpha}{r_{ik}} \frac{\partial}{\partial_{r_{ik}}} S(r_{ik}) \tag{9}$$

$$\frac{\partial}{\partial_{r_\ell^x}} \left\{\frac{\underline{r}_{ij} \cdot \underline{r}_{ik}}{r_{ij} r_{ik}}\right\} = \left(\delta_{\ell j} - \delta_{\ell i}\right) \frac{r_{ik}^\alpha}{r_{ij} r_{ik}} + \left(\delta_{\ell k} - \delta_{\ell i}\right) \frac{r_{ij}^\alpha}{r_{ij} r_{ik}} - \cos\left(\theta_{jik}\right) \left\{\left(\delta_{\ell j} - \delta_{\ell i}\right) \frac{r_{ij}^\alpha}{r_{ij}^2} +\right.$$

$$\left.\left(\delta_{\ell k} - \delta_{\ell i}\right) \frac{r_{ik}^\alpha}{r_{ik}^2}\right\} \tag{10}$$

With ℓ = i, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_i^x}} \left\{\frac{\underline{r}_{ij} \cdot \underline{r}_{ik}}{r_{ij} r_{ik}}\right\} = \left(\delta_{ij} - \delta_{ii}\right) \frac{r_{ik}^x}{r_{ij} r_{ik}} + \left(\delta_{ik} - \delta_{ii}\right) \frac{r_{ij}^x}{r_{ij} r_{ik}} - \cos\left(\theta_{jik}\right) \left\{\left(\delta_{ij} - \delta_{ii}\right) \frac{r_{ij}^x}{r_{ij}^2} +\right.$$

$$\left.\left(\delta_{ik} - \delta_{ii}\right) \frac{r_{ik}^x}{r_{ik}^2}\right\} \frac{\partial}{\partial_{r_i^x}} \left\{\frac{\underline{r}_{ij} \cdot \underline{r}_{ik}}{r_{ij} r_{ik}}\right\} = (0-1) \frac{r_{ik}^x}{r_{ij} r_{ik}} + (0-1) \frac{r_{ij}^x}{r_{ij} r_{ik}} - \cos\left(\theta_{jik}\right) \left\{(0-1) \frac{r_{ij}^x}{r_{ij}^2} +\right.$$

$$\left.(0-1) \frac{r_{ik}^x}{r_{ik}^2}\right\}$$

$$\frac{\partial}{\partial_{r_i^x}} \left\{\frac{\underline{r}_{ij} \cdot \underline{r}_{ik}}{r_{ij} r_{ik}}\right\} = -\frac{r_{ik}^x}{r_{ij} r_{ik}} - \frac{r_{ij}^x}{r_{ij} r_{ik}} - \cos\left(\theta_{jik}\right) \left\{-\frac{r_{ij}^x}{r_{ij}^2} - \frac{r_{ik}^x}{r_{ik}^2}\right\} \tag{10a}$$

With ℓ = i, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_i^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = -\frac{r_{ik}^y}{r_{ij}r_{ik}} - \frac{r_{ij}^y}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{-\frac{r_{ij}^y}{r_{ij}^2} - \frac{r_{ik}^y}{r_{ik}^2}\right\} \tag{10b}$$

With ℓ = i, and α = z; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_i^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = -\frac{r_{ik}^z}{r_{ij}r_{ik}} - \frac{r_{ij}^z}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{-\frac{r_{ij}^z}{r_{ij}^2} - \frac{r_{ik}^z}{r_{ik}^2}\right\} \tag{10c}$$

With ℓ = j, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_j^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = (\delta_{jj} - \delta_{ji})\frac{r_{ik}^x}{r_{ij}r_{ik}} + (\delta_{jk} - \delta_{ji})\frac{r_{ij}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{(\delta_{jj} - \delta_{ji})\frac{r_{ij}^x}{r_{ij}^2} + (\delta_{jk} - \delta_{ji})\frac{r_{ik}^x}{r_{ik}^2}\right\} \tag{11}$$

$$\frac{\partial}{\partial_{r_j^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = (1-0)\frac{r_{ik}^x}{r_{ij}r_{ik}} + (0-0)\frac{r_{ij}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{(1-0)\frac{r_{ij}^x}{r_{ij}^2} + (0-0)\frac{r_{ik}^x}{r_{ik}^2}\right\}$$

$$\frac{\partial}{\partial_{r_j^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ik}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ij}^x}{r_{ij}^2}\right\} \tag{11a}$$

With ℓ = j, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_j^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ik}^y}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ij}^y}{r_{ij}^2}\right\} \tag{11b}$$

With ℓ = j, and α = z; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_j^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ik}^z}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ij}^z}{r_{ij}^2}\right\} \tag{11c}$$

With ℓ = k, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_k^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = (\delta_{kj} - \delta_{ki})\frac{r_{ik}^x}{r_{ij}r_{ik}} + (\delta_{kk} - \delta_{ki})\frac{r_{ij}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{(\delta_{kj} - \delta_{ki})\frac{r_{ij}^x}{r_{ij}^2} + (\delta_{kk} - \delta_{ki})\frac{r_{ik}^x}{r_{ik}^2}\right\} \tag{12}$$

$$\frac{\partial}{\partial_{r_k^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = (0-0)\frac{r_{ik}^x}{r_{ij}r_{ik}} + (1-0)\frac{r_{ij}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{(0-0)\frac{r_{ij}^x}{r_{ij}^2} + (1-0)\frac{r_{ik}^x}{r_{ik}^2}\right\}$$

$$\frac{\partial}{\partial_{r_k^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ij}^x}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ik}^x}{r_{ik}^2}\right\} \tag{12a}$$

With ℓ = k, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_k^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ij}^y}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ik}^y}{r_{ik}^2}\right\} \tag{12b}$$

With ℓ = k, and α = z; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$\frac{\partial}{\partial_{r_k^x}}\left\{\frac{\underline{r}_{ij}\cdot\underline{r}_{ik}}{r_{ij}r_{ik}}\right\} = \frac{r_{ij}^z}{r_{ij}r_{ik}} - \cos(\theta_{jik})\left\{\frac{r_{ik}^z}{r_{ik}^2}\right\} \tag{12c}$$

$$-\frac{\partial}{\partial_{r_\ell^\alpha}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_\ell^\alpha}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(\delta_{\ell j}-\delta_{\ell i})\frac{r_{ij}^\alpha}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(\delta_{\ell k}-\delta_{\ell i})\frac{r_{ik}^\alpha}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \tag{13}$$

With ℓ = i, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_i^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_i^x}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(\delta_{ij}-\delta_{ii})\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(\delta_{ik}-\delta_{ii})\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \tag{14}$$

$$-\frac{\partial}{\partial_{r_i^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_i^x}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(0-1)\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(0-1)\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik})$$

$$-\frac{\partial}{\partial_{r_i^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_i^x}}A(\theta_{jik}) + A(\theta_{jik})S(r_{ik})\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) +$$

$$A(\theta_{jik})S(r_{ij})\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \tag{14a}$$

With ℓ = i, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_i^y}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_i^y}}A(\theta_{jik}) + A(\theta_{jik})S(r_{ik})\frac{r_{ij}^y}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) +$$

$$A(\theta_{jik})S(r_{ij})\frac{r_{ik}^y}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \quad (14b)$$

With ℓ = i, and α = z; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_i^z}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_i^y}}A(\theta_{jik}) + A(\theta_{jik})S(r_{ik})\frac{r_{ij}^z}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) +$$

$$A(\theta_{jik})S(r_{ij})\frac{r_{ik}^z}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \quad (14c)$$

With ℓ = j, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_j^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_j^x}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(\delta_{jj}-\delta_{ji})\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(\delta_{jk}-\delta_{ji})\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik}) \quad (15)$$

$$-\frac{\partial}{\partial_{r_j^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_j^x}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(1-0)\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(0-0)\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}}S(r_{ik})$$

$$-\frac{\partial}{\partial_{r_j^x}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_j^x}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) \quad (15a)$$

With ℓ = j, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_j^y}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_j^y}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})\frac{r_{ij}^y}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) \quad (15b)$$

With ℓ = j, and α = z; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_j^z}}U(\theta_{jik}\cdot r_{ij}\cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_j^z}}A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})\frac{r_{ij}^z}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}}S(r_{ij}) \quad (15c)$$

With ℓ = k, and α = x; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_k^x}} U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_k^x}} A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(\delta_{kj} - \delta_{ki})\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}} S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(\delta_{kk} - \delta_{ki})\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}} S(r_{ik}) \quad (16)$$

$$-\frac{\partial}{\partial_{r_k^x}} U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_k^x}} A(\theta_{jik}) - A(\theta_{jik})S(r_{ik})(0 - 0)\frac{r_{ij}^x}{r_{ij}}\frac{\partial}{\partial_{r_{ij}}} S(r_{ij}) -$$

$$A(\theta_{jik})S(r_{ij})(1 - 0)\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}} S(r_{ik})$$

$$-\frac{\partial}{\partial_{r_k^x}} U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_k^x}} A(\theta_{jik}) - A(\theta_{jik})S(r_{ij})\frac{r_{ik}^x}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}} S(r_{ik}) \quad (16a)$$

With ℓ = k, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_k^y}} U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_k^y}} A(\theta_{jik}) - A(\theta_{jik})S(r_{ij})\frac{r_{ik}^y}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}} S(r_{ik}) \quad (16b)$$

With ℓ = k, and α = y; $\delta_{ab} = 1$ if a = b; $\delta_{ab} = 0$ if a ≠ b

$$-\frac{\partial}{\partial_{r_k^z}} U(\theta_{jik} \cdot r_{ij} \cdot r_{ik}) = -S(r_{ij})S(r_{ik})\frac{\partial}{\partial_{r_k^z}} A(\theta_{jik}) - A(\theta_{jik})S(r_{ij})\frac{r_{ik}^z}{r_{ik}}\frac{\partial}{\partial_{r_{ik}}} S(r_{ik}) \quad (16c)$$

LAMMPS C++ CODE

```
LAMMPS - Large-scale Atomic/Molecular Massively Parallel Simulator
  http://lammps.sandia.gov, Sandia National Laboratories
  Steve Plimpton, sjplimp@sandia.gov

  Copyright (2003) Sandia Corporation.  Under the terms of Contract
  DE-AC04-94AL85000 with Sandia Corporation, the U.S. Government retains
  certain rights in this software.  This software is distributed under
  the GNU General Public License.

  See the README file in the top-level LAMMPS directory.
------------------------------------------------------------------------- */

#include <cmath>
#include <cstdlib>
#include "angle_harmonic_decay.h"
#include "atom.h"
#include "neighbor.h"
#include "domain.h"
#include "comm.h"
#include "force.h"
#include "math_const.h"
#include "memory.h"
#include "error.h"

using namespace LAMMPS_NS;
using namespace MathConst;

#define SMALL 0.001

/* ---------------------------------------------------------------------- */

AngleHarmonicDecay::AngleHarmonicDecay(LAMMPS *lmp) : Angle(lmp)
{
  k = NULL;
  theta0 = NULL;
  rho1 = NULL;
  rho2 = NULL;
}

/* ---------------------------------------------------------------------- */

AngleHarmonicDecay::~AngleHarmonicDecay()
{
  if (allocated && !copymode) {
```

```
    memory->destroy(setflag);
    memory->destroy(k);
    memory->destroy(theta0);
    memory->destroy(rho1);
    memory->destroy(rho2);
  }
}

/* ---------------------------------------------------------------------- */

void AngleHarmonicDecay::compute(int eflag, int vflag)
{
  int i1,i2,i3,n,type;
  double delx1,dely1,delz1,delx2,dely2,delz2;
  double eangle,f1[3],f3[3];
  double dtheta,tk;
  double rsq1,rsq2,r1,r2,c,s,a,a11,a12,a22,s1,s11,s22;

  eangle = 0.0;
  if (eflag || vflag) ev_setup(eflag,vflag);
  else evflag = 0;

  double **x = atom->x;
  double **f = atom->f;
  int **anglelist = neighbor->anglelist;
  int nanglelist = neighbor->nanglelist;
  int nlocal = atom->nlocal;
  int newton_bond = force->newton_bond;

  for (n = 0; n < nanglelist; n++) {
    i1 = anglelist[n][0];
    i2 = anglelist[n][1];
    i3 = anglelist[n][2];
    type = anglelist[n][3];

    // 1st bond

    delx1 = x[i1][0] - x[i2][0];
    dely1 = x[i1][1] - x[i2][1];
    delz1 = x[i1][2] - x[i2][2];

    rsq1 = delx1*delx1 + dely1*dely1 + delz1*delz1;
    r1 = sqrt(rsq1);

    // 2nd bond
delx2 = x[i3][0] - x[i2][0];
```

```
dely2 = x[i3][1] - x[i2][1];
delz2 = x[i3][2] - x[i2][2];

rsq2 = delx2*delx2 + dely2*dely2 + delz2*delz2;
r2 = sqrt(rsq2);

// angle (cos and sin)

c = delx1*delx2 + dely1*dely2 + delz1*delz2;
c /= r1*r2;

if (c > 1.0) c = 1.0;
if (c < -1.0) c = -1.0;

s = sqrt(1.0 - c*c);
if (s < SMALL) s = SMALL;
s = 1.0/s;

// force & energy

dtheta = acos(c) - theta0[type];
tk = k[type] * dtheta;

if (eflag) eangle = tk*dtheta*exp(-r1/rho1[type])*exp(-r2/rho2[type]);

a = -2.0 * tk * exp(-r1/rho1[type])*exp(-r2/rho2[type]) * s;
a11 = a*c / rsq1;
a12 = -a / (r1*r2);
a22 = a*c / rsq2;

s1 = -tk*dtheta*exp(-r1/rho1[type])*exp(-r2/rho2[type]);
s11 = -s1/(rho1[type]*r1);
s22 = -s1/(rho2[type]*r2);

f1[0] = a11*delx1 + a12*delx2 + s11*delx1;
f1[1] = a11*dely1 + a12*dely2 + s11*dely1;
f1[2] = a11*delz1 + a12*delz2 + s11*delz1;

f3[0] = a22*delx2 + a12*delx1 + s22*delx2;
f3[1] = a22*dely2 + a12*dely1 + s22*dely2;
f3[2] = a22*delz2 + a12*delz1 + s22*delz2;

// apply force to each of 3 atoms

if (newton_bond || i1 < nlocal) {
  f[i1][0] += f1[0];
```

```
      f[i1][1] += f1[1];
      f[i1][2] += f1[2];
    }

    if (newton_bond || i2 < nlocal) {
      f[i2][0] -= f1[0] + f3[0];
      f[i2][1] -= f1[1] + f3[1];
      f[i2][2] -= f1[2] + f3[2];
    }

    if (newton_bond || i3 < nlocal) {
      f[i3][0] += f3[0];
      f[i3][1] += f3[1];
      f[i3][2] += f3[2];
    }

    if (evflag) ev_tally(i1,i2,i3,nlocal,newton_bond,eangle,f1,f3,
                         delx1,dely1,delz1,delx2,dely2,delz2);
  }
}

/* ---------------------------------------------------------------------- */

void AngleHarmonicDecay::allocate()
{
  allocated = 1;
  int n = atom->nangletypes;

  memory->create(k,n+1,"angle:k");
  memory->create(theta0,n+1,"angle:theta0");
  memory->create(rho1,n+1,"angle:rho1");
  memory->create(rho2,n+1,"angle:rho2");
  memory->create(setflag,n+1,"angle:setflag");
  for (int i = 1; i <= n; i++) setflag[i] = 0;
}

/* ----------------------------------------------------------------------
   set coeffs for one or more types
------------------------------------------------------------------------- */

void AngleHarmonicDecay::coeff(int narg, char **arg)
{
  if (narg != 5) error->all(FLERR,"Incorrect args for angle coefficients");
  if (!allocated) allocate();
int ilo,ihi;
  force->bounds(FLERR,arg[0],atom->nangletypes,ilo,ihi);
```

```
  double k_one = force->numeric(FLERR,arg[1]);
  double theta0_one = force->numeric(FLERR,arg[2]);
  double rho1_one = force->numeric(FLERR,arg[3]);
  double rho2_one = force->numeric(FLERR,arg[4]);

  // convert theta0 from degrees to radians

  int count = 0;
  for (int i = ilo; i <= ihi; i++) {
    k[i] = k_one;
    theta0[i] = theta0_one/180.0 * MY_PI;
    rho1[i]= rho1_one;
    rho2[i]= rho2_one;
    setflag[i] = 1;
    count++;
  }

  if (count == 0) error->all(FLERR,"Incorrect args for angle coefficients");
}

/* ---------------------------------------------------------------------- */

double AngleHarmonicDecay::equilibrium_angle(int i)
{
  return theta0[i];
}

/* ----------------------------------------------------------------------
   proc 0 writes out coeffs to restart file
------------------------------------------------------------------------- */

void AngleHarmonicDecay::write_restart(FILE *fp)
{
  fwrite(&k[1],sizeof(double),atom->nangletypes,fp);
  fwrite(&theta0[1],sizeof(double),atom->nangletypes,fp);
  fwrite(&rho1[1],sizeof(double),atom->nangletypes,fp);
  fwrite(&rho2[1],sizeof(double),atom->nangletypes,fp);
}

/* ----------------------------------------------------------------------
   proc 0 reads coeffs from restart file, bcasts them
------------------------------------------------------------------------- */
void AngleHarmonicDecay::read_restart(FILE *fp)
{
  allocate();
```

```
  if (comm->me == 0) {
    fread(&k[1],sizeof(double),atom->nangletypes,fp);
    fread(&theta0[1],sizeof(double),atom->nangletypes,fp);
    fread(&rho1[1],sizeof(double),atom->nangletypes,fp);
    fread(&rho2[1],sizeof(double),atom->nangletypes,fp);
  }
  MPI_Bcast(&k[1],atom->nangletypes,MPI_DOUBLE,0,world);
  MPI_Bcast(&theta0[1],atom->nangletypes,MPI_DOUBLE,0,world);
  MPI_Bcast(&rho1[1],atom->nangletypes,MPI_DOUBLE,0,world);
  MPI_Bcast(&rho2[1],atom->nangletypes,MPI_DOUBLE,0,world);

  for (int i = 1; i <= atom->nangletypes; i++) setflag[i] = 1;
}

/* ----------------------------------------------------------------------
   proc 0 writes to data file
------------------------------------------------------------------------- */

void AngleHarmonicDecay::write_data(FILE *fp)
{
  for (int i = 1; i <= atom->nangletypes; i++)
    fprintf(fp,"%d %g %g %g %g\n",i,k[i],theta0[i]/MY_PI*180.0,rho1[i],rho2[i]);
}

/* ---------------------------------------------------------------------- */

double AngleHarmonicDecay::single(int type, int i1, int i2, int i3)
{
  double **x = atom->x;

  double delx1 = x[i1][0] - x[i2][0];
  double dely1 = x[i1][1] - x[i2][1];
  double delz1 = x[i1][2] - x[i2][2];
  domain->minimum_image(delx1,dely1,delz1);
  double r1 = sqrt(delx1*delx1 + dely1*dely1 + delz1*delz1);

  double delx2 = x[i3][0] - x[i2][0];
  double dely2 = x[i3][1] - x[i2][1];
  double delz2 = x[i3][2] - x[i2][2];
  domain->minimum_image(delx2,dely2,delz2);
  double r2 = sqrt(delx2*delx2 + dely2*dely2 + delz2*delz2);
  double c = delx1*delx2 + dely1*dely2 + delz1*delz2;
  c /= r1*r2;
  if (c > 1.0) c = 1.0;
  if (c < -1.0) c = -1.0;
```

```
 double dtheta = acos(c) - theta0[type];
 double tk = k[type] * dtheta * exp(-r1/rho1[type])*exp(-r2/rho2[type]);
 return tk*dtheta;

}
```

MATLAB CODE FOR CORE-ONLY DATA FILE

```
%*************************************************************
% Reading atom coordinates and boundary
%*************************************************************
function Core_only_data_file
       clc

        r = dlmread('ZnS_3x3x3_sorted.xyz',' ',2,0);

         N = 6;
         a    = 3.8227  * N;
         b    = 6.62111 * N;
         c    = 6.2607  * N;

        Write_lAMMPS_CoreShell(r)

 end

%*************************************************************
% Setting conditions for bonds generation
%*************************************************************
 function [bond, nbonds] = find_bonds(atom,a,b,c,PERIODIC)

     n = length(atom(:,1));
     nbonds = 0;
     for i = 1:n

         atomA(i,:) = atom(i,:);

         for j = 1:n

             atomB(j,:) = atom(j,:);
```

```
            if atomA(i,1) ~= atomB(j,1) && atomA(i,1) == 2
&& atomB(j,1) == 1

                xl = abs(atomA(i,2) - atomB(j,2));
                yl = abs(atomA(i,3) - atomB(j,3));
                zl = abs(atomA(i,4) - atomB(j,4));

                if(PERIODIC)
                    if (xl > a / 2.0)
                        xl = a - xl;

                    end
                    if (yl > b / 2.0)
                        yl = b - yl;
                    end
                    if (zl > c / 2.0)
                        zl = c - zl;
                    end
                end

                d = sqrt(xl * xl + yl * yl + zl * zl);

                if (d <= 2.37 + 0.1 && d >= 1.25)
                     nbonds = nbonds+1;
                     bond(nbonds,:) = [nbonds  1  i  j];
                end

            end

        end
    end

 end

%*************************************************************
% Setting conditions for angles generation
%*************************************************************

function [angle,nangle] = find_angles(bond)
     n = length(bond(:,1));
     nangle = 1;
     angle(nangle,:) = [0 0 0];
     for i = 1:n
         for j = 1:n
```

```
            if  bond(i,1) ~= bond(j,1) && bond(i,4) ==
bond(j,4)
                temp(1,:)  = [bond(i,3) bond(j,4)
bond(j,3)];
                temp2(1,:) = [bond(j,3) bond(j,4)
bond(i,3)];
                if ismember(temp2,angle,'rows') == false
                    angle(nangle,:) = [temp];
                    nangle = nangle+1;
                end
            end
        end
    end
    nangle = nangle - 1;
end

%************************************************************
% Output to LAMMPS data file
%************************************************************

function Write_lAMMPS_CoreShell(p)

    n = length(p(:,1));
    N = 6;

     a    = 3.8227  * N;
     b    = 6.62111 * N;
     c    = 6.2607  * N;

    xlo = -a/2;
    xhi =  a/2;
    ylo =  -b/2;
    yhi =  b/2;
    zlo = -c/2;
    zhi =  c/2;

    [bond, nbonds] = find_bonds(p,a,b,c,1);

    [angle,nangle] = find_angles(bond)

     charge = [2.0  0.0; -2.0 0];

     fId = fopen('ZnS_3x3x3_CoreCore.data','w');
```

```
    fprintf(fId,'LAMMPS data file for ZnO ...\n\n')
    fprintf(fId,'%d atoms\n',n)
    fprintf(fId,'%d bonds\n',0)
    fprintf(fId,'%d angles\n',nangle)
    fprintf(fId,'%d dihedrals\n',0)
    fprintf(fId,'%d impropers\n\n\n',0)

    fprintf(fId,'%d atom types\n',2)
    fprintf(fId,'%d bond types\n',0)
    fprintf(fId,'%d angle types\n',1)
    fprintf(fId,'%d dihedral types\n',0)
    fprintf(fId,'%d improper types\n\n\n',0)
    fprintf(fId,'%f\t %f\t xlo xhi\n',xlo,xhi)
    fprintf(fId,'%f\t %f\t ylo yhi\n',ylo,yhi)
    fprintf(fId,'%f\t %f\t zlo zhi\n\n\n',zlo,zhi)

    fprintf(fId,'Masses\n\n');
    fprintf(fId,'1  65.38\n')
    fprintf(fId,'2  32.059\n')


%*********************************************************
% Setting number of atoms generation
%*********************************************************

    fprintf(fId,'Atoms\n\n');

    natoms = 1;
    for i = 1:n
        fprintf(fId,'%d %d %d %f %f %f
%f\n',natoms,i,p(i,1),charge(p(i,1),1),p(i,2),p(i,3),p(i,4)
);
        natoms = natoms + 1;
    end

%*********************************************************
% Setting number of angles generation
%*********************************************************

    fprintf(fId,'\n\nAngles\n\n');

    for i = 1:nangle

% Core-Core-Core
```

```
            fprintf(fId,'%d %d %d %d
%d\n',i,1,angle(i,1),angle(i,2),angle(i,3));


    end


    fclose(fId);

 end
```

MATLAB CODE FOR CORE-SHELL DATA FILE

```
%*************************************************************
% Reading atom coordinates and boundary
%*************************************************************

function Core_shell_data_file
       clc

         r  = dlmread('ZnS_3x3x3_sorted.xyz',' ',2,0);

         N = 6;
         a    = 3.8227  * N;
         b    = 6.62111 * N;
         c    = 6.2607  * N;
        Write_lAMMPS_CoreShell(r)

 end


%*************************************************************
% Setting conditions for bonds generation
%*************************************************************

 function [bond, nbonds] = find_bonds(atom,a,b,c,PERIODIC)

     n = length(atom(:,1));
     nbonds = 0;
     for i = 1:n

         atomA(i,:) = atom(i,:);

         for j = 1:n
```

```
            atomB(j,:) = atom(j,:);

            if atomA(i,1) ~= atomB(j,1) && atomA(i,1) == 2
&& atomB(j,1) == 1

                xl = abs(atomA(i,2) - atomB(j,2));
                yl = abs(atomA(i,3) - atomB(j,3));
                zl = abs(atomA(i,4) - atomB(j,4));

                if(PERIODIC)
                    if (xl > a / 2.0)
                        xl = a - xl;
                    end
                    if (yl > b / 2.0)
                        yl = b - yl;
                    end
                    if (zl > c / 2.0)
                        zl = c - zl;
                    end
                end

                d = sqrt(xl * xl + yl * yl + zl * zl);

                if (d <= 2.37 + 0.1 && d >= 1.25)
                     nbonds = nbonds+1;
                     bond(nbonds,:) = [nbonds  1  i  j];
                end

            end

        end
    end

 end

%*************************************************************
% Setting conditions for angles generation
%*************************************************************

 function [angle,nangle] = find_angles(bond)
    n = length(bond(:,1));
    nangle = 1;
    angle(nangle,:) = [0 0 0];
```

```
     for i = 1:n
         for j = 1:n
             if  bond(i,1) ~= bond(j,1) && bond(i,4) ==
bond(j,4)
                 temp(1,:)  = [bond(i,3) bond(j,4)
bond(j,3)];
                 temp2(1,:) = [bond(j,3) bond(j,4)
bond(i,3)];
                 if ismember(temp2,angle,'rows') == false
                     angle(nangle,:) = [temp];
                     nangle = nangle+1;
                 end
             end
         end
     end
     nangle = nangle - 1;
 end

%************************************************************
% Output to LAMMPS data file
%************************************************************

function Write_lAMMPS_CoreShell(p)

    n = length(p(:,1));
    N = 6;

     a    = 3.8227  * N;
     b    = 6.62111 * N;
     c    = 6.2607  * N;

    xlo = -a/2;
    xhi =  a/2;
    ylo =  -b/2;
    yhi =  b/2;
    zlo = -c/2;
    zhi =  c/2;

    [bond, nbonds] = find_bonds(p,a,b,c,1);

    [angle,nangle] = find_angles(bond)

     charge = [2.0, 0.0; 1.087526, -3.087526];  % Namsani
```

```
      fId = fopen('ZnS_3x3x3_CoreShell.data','w');

    fprintf(fId,'LAMMPS data file for ZnS ...\n\n')
    fprintf(fId,'%d atoms\n',n*2)
    fprintf(fId,'%d bonds\n',n)
    fprintf(fId,'%d angles\n',nangle)
    fprintf(fId,'%d dihedrals\n',0)
    fprintf(fId,'%d impropers\n\n\n',0)

    fprintf(fId,'%d atom types\n',4)
    fprintf(fId,'%d bond types\n',2)
    fprintf(fId,'%d angle types\n',1)
    fprintf(fId,'%d dihedral types\n',0)
    fprintf(fId,'%d improper types\n\n\n',0)

    fprintf(fId,'%f\t %f\t xlo xhi\n',xlo,xhi)
    fprintf(fId,'%f\t %f\t ylo yhi\n',ylo,yhi)
    fprintf(fId,'%f\t %f\t zlo zhi\n\n\n',zlo,zhi)

    fprintf(fId,'Masses\n\n');
    fprintf(fId,'1  58.842\n')
    fprintf(fId,'2  28.8585\n')
    fprintf(fId,'3  6.538\n')
    fprintf(fId,'4  3.2065\n\n\n')

    fprintf(fId,'Atoms\n\n');

%*************************************************************
% Setting number of atoms generation
%*************************************************************
     natoms = 1;
    for i = 1:n
        fprintf(fId,'%d %d %d %f %f %f
%f\n',natoms,i,p(i,1),charge(p(i,1),1),p(i,2),p(i,3),p(i,4)
);
        natoms = natoms + 1;
         fprintf(fId,'%d %d %d %f %f %f
%f\n',natoms,i,p(i,1)+2,charge(p(i,1),2),p(i,2),p(i,3),p(i,
4));
         natoms = natoms + 1;
    end
     fprintf(fId,'\n\nBonds\n\n');

    natoms = 1;
```

```
    for i = 1:n
        fprintf(fId,'%d %d %d
%d\n',i,p(i,1),natoms,natoms+1);
        natoms = natoms + 2;
    end

%************************************************************
% Setting number of angles generation
%************************************************************

    fprintf(fId,'\n\nAngles\n\n');

    for i = 1:nangle
        % Core-Core-Core
%             fprintf(fId,'%d %d %d %d
%d\n',i,1,angle(i,1)*2-1,angle(i,2),angle(i,3)*2-1);

        % Shell-Core-Shell
            fprintf(fId,'%d %d %d %d
%d\n',i,1,angle(i,1)*2,angle(i,2)*2-1,angle(i,3)*2);

        % Shell-Shell-Shell
%             fprintf(fId,'%d %d %d %d
%d\n',i,1,angle(i,1)*2,angle(i,2)*2,angle(i,3)*2);

    end

%************************************************************
% The additional section of atom ID and core-shell ID
%************************************************************

    fprintf(fId,'\n\nCS-Info\n\n');

    natoms = 1;
    for i = 1:n
        fprintf(fId,'%d %d\n',natoms,i)
        natoms = natoms + 1;
        fprintf(fId,'%d %d\n',natoms,i)
        natoms = natoms + 1;
    end

    fclose(fId);

 end
```

APPENDIX C:

CODE USED TO SIMULATE PIEZOELECTIC CONSTANTS IN THIS THESIS

LAMMPS CODE FOR CORE-CORE SIMULATION

```
# Units metal = Energy in eV, time in picoseconds, distance in
Angstroms, and charge in multiple of electron charge

units         metal

#Bounday condition
boundary      p p p

# Molecular + Charge
atom_style    full

# T = Temperature in K
variable      T1 equal 500
variable      T2 equal 0.1

variable      N equal 3
variable      N2 equal 10000

# Generate a Wurtzite crystal with data file
read_data               ZnS_3x3x3x2_dihedral_CoreCore.data

pair_style               buck/coul/long/cs 12

# Atomic mass of Zn
mass                     1 65.38

# Atomic mass of S
mass                     2 32.059

set                      type 1 charge  2.0
set                      type 2 charge -2.0

dielectric               1.0

# Zn_core – Zn_core
pair_coeff              * * 0.0 1.0 0.0 0.0

# Zn_core - S_core
pair_coeff              1 2 672.288 0.39089 0.0

# S_core - S_core
pair_coeff              2 2 1200.0 0.149 0.0

angle_style             harmonic_decay
```

```
angle_coeff             1 4714170 109.47 0.3 0.3

dihedral_style          charmm

dihedral_coeff          1 0.005 3 0 0.0

special_bonds           coul 0.0 0.0 1.0 angle yes dihedral yes

pair_modify             shift yes
kspace_style            ewald/disp 1.0E-06
neighbor                3.0 bin


variable                xl equal (lx/(2*$N))
variable                yl equal (ly/(2*$N))
variable                zl equal (lz/(2*$N))

neigh_modify            every 1 delay 0 check yes

dump                     1 all xyz ${N2}
ZnS_Binks_$Nx$Nx$N_${T2}K_Relaxed.xyz
#dump_modify              1 element Zn O


dump       12 all custom ${N2} ZnS_$Nx$Nx$N_${T2}K_Z_U.atom id type xu
yu zu

dump_modify 12 sort id


compute peratom all pe/atom

####################################### MD EQUILIBRATION

reset_timestep  0

# Time step in ps
timestep 0.001

velocity all create ${T1} 12345 mom yes rot no
fix 1 all npt temp ${T1} ${T1} 1 iso 0 0 1 drag 1
fix 2 all temp/rescale 1 ${T1} ${T1} 0.05 1.0

variable   al equal v_xl
variable   bl equal v_yl
variable   cl equal v_zl
variable   s equal step
variable   Vol  equal vol
variable   srate equal -1.0e08

# Set thermo output
thermo 1000
```

```
thermo_style custom step lx ly lz press pxx pyy pzz pe temp v_al v_bl
v_cl vol v_xl v_yl v_zl

# Run for at least 10 picosecond (assuming 1 fs timestep)
# Number of steps (Time = timestep * number of steps)

run                     10000

unfix                   1
unfix                   2

fix 1 all npt temp ${T1} ${T2} 1 iso 0 0 1 drag 1
fix 2 all temp/rescale 1 ${T1} ${T2} 0.05 1.0

run                     100000

unfix                   1
unfix                   2

fix 1 all npt temp ${T2} ${T2} 1 iso 0 0 1 drag 1
fix 2 all temp/rescale 1 ${T2} ${T2} 0.05 1.0

fix def2 all print 200000 "0 0 ${al} ${bl} ${cl} ${Vol}" file
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_lattice.txt screen no title ""

run                     200000

unfix                   1
unfix                   2

# MS

#min_style               cg
#min_modify              line quadratic dmax 0.001
#fix                     1 all box/relax iso 0.0 vmax 0.001
#minimize                1.0E-09 1.0E-09 100000 1000000

#unfix                   1
undump                  1


# Store final cell length for strain calculations
variable                tmp equal "lz"
variable                L0 equal ${tmp}
print                   "Initial Length, L0: ${L0}"


######################################## DEFORMATION

reset_timestep          0

fix        1 all nvt/sllod temp ${T2} ${T2} 1
```

```
# Strain rate in 1/ps

variable srate1 equal "v_srate/1.0e12"

fix        2 all deform 1 z erate ${srate1} units box remap v

# Output strain and stress info to file
# For units metal, pressure is in [bars] = 100 [kPa] = 1/10000 [GPa]

compute        iyad all stress/atom NULL
compute        p all reduce sum c_iyad[1] c_iyad[2] c_iyad[3]
variable       press equal -(c_p[1]+c_p[2]+c_p[3])/(3*vol)

# p2, p3, p4 are in GPa
# strain = dl/lo

variable strain equal "(lz - v_L0)/v_L0"
variable p1 equal "v_strain"
variable p2 equal "-pxx/10000"
variable p3 equal "-pyy/10000"
variable p4 equal "-pzz/10000"


fix def1 all print ${N2} "${p1} ${p2} ${p3} ${p4}" file
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_defl.txt screen no

fix def2 all print ${N2} "${s} ${p1} ${al} ${bl} ${cl} ${Vol}" append
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_lattice.txt screen no title ""

# Display thermo
thermo     ${N2}
thermo_style    custom step v_strain temp v_p2 v_p3 v_p4 ke pe press
v_al v_bl v_cl vol v_s v_press

dump         1 all xyz ${N2} ZnS_${T2}K_Z_SS_${srate}.*.xyz

dump         2 all custom ${N2} ZnS_${T2}K_Z_SS_U_${srate}.*.atom id
type xu yu zu

dump_modify  2 sort id

run        100000

#######################################
# SIMULATION DONE
print "All done"
```

LAMMPS CODE FOR CORE-SHELL SIMULATION

```
# -------------------- INITIALIZATION ----------------------------
# Units metal = Energy in eV, time in picoseconds, distance in
Angstroms, and charge in multiple of electron charge

units           metal
dimension       3

# Boundary condition
boundary        p p p

atom_style      full

# -------------------- ATOM DEFINITION ----------------------------

fix             csinfo all property/atom i_CSID

read_data       ZnS_3x3x3x2_dihedral_Coreshell.data fix csinfo NULL
CS-Info

group cores     type 1 2
group shells    type 3 4

neighbor        2.0 bin
comm_modify     vel yes

variable        N equal 3
variable        T1 equal 500
variable        T2 equal 0.1
variable        N2 equal 10000

# --------------------- FORCE FIELDS -----------------------------

kspace_style            ewald 1.0e-6

pair_style              buck/coul/long/cs 12

# Zn_core – Zn_core
pair_coeff              * * 0.0 1.0 0.0 0.0

# Zn_core - S_shell
pair_coeff              1 4 672.288 0.39089 0.0

# S_shell - S_shell
pair_coeff              4 4 1200.0 0.149 0.0

pair_modify             shift yes

angle_style             harmonic_decay
```

```
# Angle style S_Shell - Zn_Core - S_Shell
angle_coeff             1 4714170 109.47 0.3 0.3

# Four-body term
dihedral_style          charmm

dihedral_coeff          1 0.005 3 0 0.0

special_bonds           coul 0.0 0.0 1.0 angle yes dihedral yes

bond_style              harmonic
bond_coeff              1 0 0
bond_coeff              2 6.6513715 0


# ----------------- Equilibration Run --------------------------------

reset_timestep      0

variable            xl equal (lx/(2*$N))
variable            yl equal (ly/(2*$N))
variable            zl equal (lz/(2*$N))

thermo 1000
thermo_style custom step lx ly lz press pxx pyy pzz pe temp v_xl v_yl
v_zl

compute CStemp all temp/cs cores shells
compute thermo_press_lmp all pressure thermo_temp # press for correct
kinetic scalar

thermo_modify temp CStemp press thermo_press_lmp

# Time step in ps

timestep 0.001

# Velocity bias option

velocity all create ${T1} 134 dist gaussian mom yes rot no bias yes
temp CStemp
velocity all scale ${T1} temp CStemp

# Thermosetting using the core/shell decoupling

fix               1 all temp/berendsen ${T1} ${T1} 0.4
fix               2 all nve
fix_modify        1 temp CStemp

variable   al equal v_xl
variable   bl equal v_yl
variable   cl equal v_zl
```

```
variable   s equal step
variable   Vol  equal vol
variable   srate equal -1.0e08

dump              1 all xyz ${N2}
ZnS_Binks_$Nx$Nx$N_${T2}K_Relaxed.xyz
dump              12 all custom ${N2} ZnS_$Nx$Nx$N_${T2}K_Z_U.atom id
type xu yu zu
dump_modify       12 sort id

run 10000

unfix 1
unfix 2

fix               3 all npt temp ${T1} ${T2} 0.04 iso 0 0 0.4
fix              11 all temp/rescale 1 ${T1} ${T2} 0.05 1.0
fix_modify        3 temp CStemp press thermo_press_lmp # pressure for
correct kinetic scalar

fix def2 all print 100000 "0 0 ${al} ${bl} ${cl} ${Vol}" file
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_lattice.txt screen no title ""

run 100000

unfix 3
unfix 11


fix              1 all npt temp ${T2} ${T2} 1 iso 0 0 1 drag 1
fix              2 all temp/rescale 1 ${T2} ${T2} 0.05 1.0

fix def2 all print 100000 "0 0 ${al} ${bl} ${cl} ${Vol}" file
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_lattice.txt screen no title ""

run              100000

unfix            1
unfix            2


# Store final cell length for strain calculations
variable     tmp equal "lz"
variable     L0 equal ${tmp}
print        "Initial Length, L0: ${L0}"

#min_style       cg
#min_modify      line quadratic dmax 0.001
#fix             1 all box/relax iso 0.0 vmax 0.001
neigh_modify    every 1 delay 0 check yes
thermo_style    custom step etotal evdwl ecoul v_xl v_yl v_zl
thermo          10
```

```
#minimize                1.0E-09 1.0E-09 100000 1000000

write_restart           optimized.data

# Store final cell length for strain calculations
variable     tmp equal "lz"
variable     L0 equal ${tmp}
print "Initial Length, L0: ${L0}"

undump                  1

# ----------------------- Dynamic Run ------------------------------
reset_timestep  0

fix            3 all nvt/sllod temp ${T2} ${T2} 1 drag 1

variable     srate1 equal "v_srate / 1.0e12"

fix            2 all deform 1 z erate ${srate1} units box remap v

fix_modify    3 temp CStemp # pressure for correct kinetic scalar

# Output strain and stress info to file
# For units metal, pressure is in [bars] = 100 [kPa] = 1/10000 [GPa]
# p2, p3, p4 are in GPa

variable dl     equal "((lz-v_L0)/2) + v_L0"
variable strain equal "(lz-v_L0)/v_L0"
variable p1     equal "v_strain"
variable p2     equal "-pxx/10000"
variable p3     equal "-pyy/10000"
variable p4     equal "-pzz/10000"
variable vol    equal "lx*ly*lz"

fix def1 all print ${N2} "${p1} ${p2} ${p3} ${p4}" file
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_defl.txt screen no

fix def2 all print ${N2} "${s} ${p1} ${al} ${bl} ${cl} ${Vol}" append
ZnS_$Nx$Nx$N_${T2}K_Z_${srate}_lattice.txt screen no title ""

# Display thermo
thermo          ${N2}
thermo_style    custom step v_strain temp press v_xl v_yl v_zl vol

dump         1 all xyz ${N2} ZnS_${T2}K_Z_SS_${srate}.*.xyz
dump         2 all custom ${N2} ZnS_${T2}K_Z_SS_U_${srate}.*.atom id
type xu yu zu

dump_modify  2 sort id
```

```
run 100000
```

MATLAB CODE FOR CORE-CORE SIMULATION

```
function Core_core_simulation
    clc
    format short

    rn = load('ZnS_3x3x3_0.1K_Z_-100000000_lattice.txt');
    rp = load('ZnS_3x3x3_0.1K_Z_100000000_lattice.txt');

    V = 20569.64;

 % Charge of electron
    e = 1.6021766208 * 10^-19;
    q = 2 * e;

    dP3du = (4*(-2.06))/(sqrt(3)*(3.281^2))
    dudE3 = -2.3;
    e33 = -0.44 + dP3du*dudE3

    z_dir = 5;
    r  = dlmread('ZnS_0.1K_Z_SS_U_-100000000.0.atom','
',9,0);
    a = rn(1,3)
    b = rn(1,4)
    c = rn(1,5)
    V = rn(1,6)

    Pv_0 = Polarization(r,z_dir,V)
    Zn_O = Uk(r)
    c = c
    u_0  = Zn_O/c

    Pa_0 = (-4 * q * u_0)/(sqrt(3) * a^2 * 10^-20)


    [E_n,P_a_n,U_n] = Polarization_a(rn,'ZnS_0.1K_Z_SS_U_-
100000000.');
    [E_p,P_a_p,U_p] =
Polarization_a(rp,'ZnS_0.1K_Z_SS_U_100000000.');

% Strain
```

```
    E_n  = E_n(length(E_n):-1:1);
    E = [E_n,E_p];

% Polarizations
    P_a_n = P_a_n(length(P_a_n):-1:1);
    P_a = [P_a_n,P_a_p]

% Coordinate faction
    U_n  = U_n(length(U_n):-1:1);
    U = [U_n,U_p];

% Volume parameters
    V_n = rn(:,6);
    V_n = V_n(length(V_n):-1:1);
    Vc = [V_n;rp(:,6)]';

    [E_n,P_v_n] = Polarization_V(rn,'ZnS_0.1K_Z_SS_U_-
100000000.',z_dir);
    [E_p,P_v_p] =
Polarization_V(rp,'ZnS_0.1K_Z_SS_U_100000000.',z_dir);

    P_v_n = P_v_n(length(P_v_n):-1:1);
    P_v = [P_v_n,P_v_p]


% Experimental Value
    e_a_int_Die = -0.182 * -6.94


% Plot for dU/dE
figure(1)
    plot(E,U,'sk','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,U,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'--k','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('u (eV)','fontweight','bold','fontsize',15)
    legend('Wright (2004) potential','linear Fit of
u','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dUdE = p(2)
```

```
% Plot for dP/dU
figure(2)
    plot(U,P_v,'--k','LineWidth',3,'MarkerSize',10)
    p = polyfit(U,P_v,1);
    f = polyval(p,U);
    hold on
    plot(U,f,'sk','LineWidth',3,'MarkerSize',10)
    xlabel('U(eV)','fontweight','bold','fontsize',15')
    ylabel('P_v (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_vdU = p(1)

% Using left side of proper piezoelectric constants
    e_v_int_l = dP_vdU * dUdE

% Plot for dP/dE
figure(3)
    plot(E,P_v,'--b','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,P_v,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel([char(949)],'fontweight','bold','fontsize',15)
    ylabel('P_v (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_vdE = p(2)


% Plot for dP_a/dU
figure(4)
    plot(U,P_a,'--b','LineWidth',3,'MarkerSize',10)
    p1 = polyfit(U,P_a,1);
    f = polyval(p1,U);
    hold on
    plot(U,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel('U (eV)','fontweight','bold','fontsize',15)
    ylabel('P_a (C/m^2)','fontweight','bold','fontsize',15)
```

```
set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_adU = p1(1)


% Using right side of proper piezoelectric constants
    e_a_int_r = dP_adU * dUdE

% Plot for dP_a/dE
figure(5)
    plot(E,P_a,'--b','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,P_a,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel([char(949)],'fontweight','bold','fontsize',15)
    ylabel('P_a (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_adE = p(2)

% Calculate e_33^0
    e_a_0 = dP_adE - e_a_int_r

% Plot for dE/dV
figure(6)
    plot(E,Vc,'--k','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,Vc,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sk','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('V (A^3)','fontweight','bold','fontsize',15)
    legend('linear Fit of V - Wright
2004','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

% Plot for compare dP_a/dE with dP_v/dE
figure(7)
plot(E,P_a,'sk',E,P_v,'ok','LineWidth',3,'MarkerSize',10)
```

```
    p = polyfit(E,P_a,2)
    f_a = polyval(p,E);
    p = polyfit(E,P_v,2)
    f_v = polyval(p,E);
    hold on
    plot(E,f_a,'-.k',E,f_v,'--
k','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('P_3-core-core
(C/m^2)','fontweight','bold','fontsize',15)
    legend('P_3(u) - Wright 2004','P_3(v) - Wright
2004','linear Fit of
P_3','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

% Plot for compare dP_a/dU with dP_v/dU
figure(8)
plot(U,P_a,'sk',U,P_v,'ok','LineWidth',3,'MarkerSize',10)
    p1 = polyfit(U,P_a,1)
    f_a = polyval(p1,U);
    p2 = polyfit(U,P_v,1)
    f_v = polyval(p2,U);
    hold on
    plot(U,f_a,'sk',U,f_v,'--
k','LineWidth',3,'MarkerSize',10)
    xlabel('u (eV)','fontweight','bold','fontsize',15)
    ylabel('P_3-core-core
(C/m^2)','fontweight','bold','fontsize',15)
    legend('P_3(u) - Wright 2004','P_3(v) - Wright
2004','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

    hold off

% Calculate P3 using coordinate fraction u
    function  [E,P,U] = Polarization_a(r1,s)
       q = 2.0 * e;
       n = length(r1(:,1));
       for i = 1:n
            st = append(s,int2str(r1(i,1)),'.atom');
```

```
            r2  = dlmread(st,' ',9,0);
            Zn_O = Uk(r2);
            a = r1(i,3);
            c = r1(i,5);
            c_a = c/a;
            u   = Zn_O/c;
            U(i) = u;
            P(i) = (-4 * q * u)/(sqrt(3) * a^2 * 10^-20);
            dP_adu = (-4 * q)/(sqrt(3) * a^2 * 10^-20)
            E(i) = r1(i,2);
        end
    end

 % Calculate P3 using volume
    function  [E,P] = Polarization_V(r1,s,dir)
       n = length(r1(:,1));
       for i = 1:n
            V   = r1(i,6);
            st = append(s,int2str(r1(i,1)),'.atom');
            r2  = dlmread(st,' ',9,0);
            P(i) = Polarization(r2,dir,V);
            E(i) = r1(i,2);
       end
    end

end

% Calculate P3
function [P3] = Polarization(r,dir,V)
    e = 1.6021766208 * 10^-19;  % C
    z1 =  2.0;  % Zn electrons charge
    z2 = -2.0;  % S electrons charge
    n = length(r(:,1));
    sum = 0;
    for i = 1:n;
        type = r(i,2);
        ri = r(i,dir) * 10^-10;  % from A to m
        if(type == 1)
% fprintf('element = Zn  charge = 2\n');
            sum = sum + (z1 * e) * ri;   % C.m
        elseif (type ==2)
%           fprintf('element = O   charge = -2\n');
            sum = sum + (z2 * e) * ri;
        end
```

```
    end
    P3 = sum/V;
    P3 = P3 / (10^-10)^3;
end

% Calculate coordinate fraction u
function u = Uk(r)
    u = 1;
    n = length(r(:,1));
    sum = 0;
    counter = 0;
    for i = 1:n
        id1   = r(i,1);
        type1 = r(i,2);
        for j = 1:n
            id2 = r(j,1);
            type2 = r(j,2);
            if (type1 == 1 && type2 == 2)
                x1 = r(i,3);
                y1 = r(i,4);
                z1 = r(i,5);
                %
                x2 = r(j,3);
                y2 = r(j,4);
                z2 = r(j,5);
                if abs(x2-x1) < 0.3 && abs(y2-y1) < 0.3 &&
abs(z2-z1) <= 2.6
                    d = sqrt( (z2-z1)^2 );
                    sum = sum + d;
                    counter = counter + 1;
                end
            end
        end
        u = sum/counter;
    end
end
```

MATLAB CODE FOR CORE-SHELL SIMULATION

```
function Core_shell_simulation
    clc
    format short

    rn = load('ZnS_3x3x3_0.1K_Z_-100000000_lattice.txt');
    rp = load('ZnS_3x3x3_0.1K_Z_100000000_lattice.txt');

    V = 20569.64;

 % Charge of electron
    e = 1.6021766208 * 10^-19;
    q = 2 * e;

    dP3du = (4*(-2.06))/(sqrt(3)*(3.281^2))
    dudE3 = -2.3;
    e33 = -0.44 + dP3du*dudE3

    z_dir = 5;
    r  = dlmread('ZnS_0.1K_Z_SS_U_-100000000.0.atom','
',9,0);
    a = rn(1,3)
    b = rn(1,4)
    c = rn(1,5)
    V = rn(1,6)

    Pv_0 = Polarization(r,z_dir,V)
    Zn_O = Uk(r)
    c = c
    u_0  = Zn_O/c

    Pa_0 = (-4 * q * u_0)/(sqrt(3) * a^2 * 10^-20)


    [E_n,P_a_n,U_n] = Polarization_a(rn,'ZnS_0.1K_Z_SS_U_-
100000000.');
    [E_p,P_a_p,U_p] =
Polarization_a(rp,'ZnS_0.1K_Z_SS_U_100000000.');

% Strain
    E_n  = E_n(length(E_n):-1:1);
    E = [E_n,E_p];
```

```
% Polarizations
    P_a_n = P_a_n(length(P_a_n):-1:1);
    P_a = [P_a_n,P_a_p]

% Coordinate faction
    U_n  = U_n(length(U_n):-1:1);
    U = [U_n,U_p];

% Volume parameters
    V_n = rn(:,6);
    V_n = V_n(length(V_n):-1:1);
    Vc = [V_n;rp(:,6)]';

    [E_n,P_v_n] = Polarization_V(rn,'ZnS_0.1K_Z_SS_U_-
100000000.',z_dir);
    [E_p,P_v_p] =
Polarization_V(rp,'ZnS_0.1K_Z_SS_U_100000000.',z_dir);

    P_v_n = P_v_n(length(P_v_n):-1:1);
    P_v = [P_v_n,P_v_p]


% Experimental Value
    e_a_int_Die = -0.182 * -6.94


% Plot for dU/dE
figure(1)
    plot(E,U,'sk','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,U,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'--k','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('u (eV)','fontweight','bold','fontsize',15)
    legend('Wright (2004) potential','linear Fit of
u','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dUdE = p(2)


% Plot for dP/dU
```

```
figure(2)
    plot(U,P_v,'--k','LineWidth',3,'MarkerSize',10)
    p = polyfit(U,P_v,1);
    f = polyval(p,U);
    hold on
    plot(U,f,'sk','LineWidth',3,'MarkerSize',10)
    xlabel('U(eV)','fontweight','bold','fontsize',15')
    ylabel('P_v (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_vdU = p(1)

% Using left side of proper piezoelectric constants
    e_v_int_l = dP_vdU * dUdE

% Plot for dP/dE
figure(3)
    plot(E,P_v,'--b','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,P_v,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel([char(949)],'fontweight','bold','fontsize',15)
    ylabel('P_v (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_vdE = p(2)


% Plot for dP_a/dU
figure(4)
    plot(U,P_a,'--b','LineWidth',3,'MarkerSize',10)
    p1 = polyfit(U,P_a,1);
    f = polyval(p1,U);
    hold on
    plot(U,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel('U (eV)','fontweight','bold','fontsize',15)
    ylabel('P_a (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_adU = p1(1)
```

```
% Using right side of proper piezoelectric constants
    e_a_int_r = dP_adU * dUdE

% Plot for dP_a/dE
figure(5)
    plot(E,P_a,'--b','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,P_a,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sr','LineWidth',3,'MarkerSize',10)
    xlabel([char(949)],'fontweight','bold','fontsize',15)
    ylabel('P_a (C/m^2)','fontweight','bold','fontsize',15)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    dP_adE = p(2)

% Calculate e_33^0
    e_a_0 = dP_adE - e_a_int_r

% Plot for dE/dV
figure(6)
    plot(E,Vc,'--k','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,Vc,2);
    f = polyval(p,E);
    hold on
    plot(E,f,'sk','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('V (A^3)','fontweight','bold','fontsize',15)
    legend('linear Fit of V - Wright
2004','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

% Plot for compare dP_a/dE with dP_v/dE
figure(7)
plot(E,P_a,'sk',E,P_v,'ok','LineWidth',3,'MarkerSize',10)
    p = polyfit(E,P_a,2)
    f_a = polyval(p,E);
    p = polyfit(E,P_v,2)
    f_v = polyval(p,E);
```

```
    hold on
    plot(E,f_a,'-.k',E,f_v,'--
k','LineWidth',3,'MarkerSize',10)
    xlabel('\epsilon_3','fontweight','bold','fontsize',15)
    ylabel('P_3-core-core
(C/m^2)','fontweight','bold','fontsize',15)
    legend('P_3(u) - Wright 2004','P_3(v) - Wright
2004','linear Fit of
P_3','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

% Plot for compare dP_a/dU with dP_v/dU
figure(8)
plot(U,P_a,'sk',U,P_v,'ok','LineWidth',3,'MarkerSize',10)
    p1 = polyfit(U,P_a,1)
    f_a = polyval(p1,U);
    p2 = polyfit(U,P_v,1)
    f_v = polyval(p2,U);
    hold on
    plot(U,f_a,'sk',U,f_v,'--
k','LineWidth',3,'MarkerSize',10)
    xlabel('u (eV)','fontweight','bold','fontsize',15)
    ylabel('P_3-core-core
(C/m^2)','fontweight','bold','fontsize',15)
    legend('P_3(u) - Wright 2004','P_3(v) - Wright
2004','fontweight','bold','fontsize',20)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on

    hold off

% Calculate P3 using coordinate fraction u
function  [E,P,U] = Polarization_a(r1,s)
       q = 2.0 * e;
       n = length(r1(:,1));
       for i = 1:n
            st = append(s,int2str(r1(i,1)),'.atom');
            r2  = dlmread(st,' ',9,0);
            Zn_O = Uk(r2);
            a = r1(i,3);
            c = r1(i,5);
```

```
            c_a = c/a;
            u   = Zn_O/c;
            U(i) = u;
            P(i) = (-4 * q * u)/(sqrt(3) * a^2 * 10^-20);
            dP_adu = (-4 * q)/(sqrt(3) * a^2 * 10^-20)
            E(i) = r1(i,2);
        end
    end


% Calculate P3 using volume
    function  [E,P] = Polarization_V(r1,s,dir)
       n = length(r1(:,1));
       for i = 1:n
            V   = r1(i,6);
            st = append(s,int2str(r1(i,1)),'.atom');
            r2  = dlmread(st,' ',9,0);
            P(i) = Polarization_CS(r2,dir,V);
            E(i) = r1(i,2);
       end
    end

end

% Calculate Polarization P3
function [P3] = Polarization(r,dir,V)
    e = 1.6021766208 * 10^-19;  % C
    z1 =  2.0;  % Valence electrons
    z2 = -2.0;  %
    n = length(r(:,1));
    sum = 0;
    for i = 1:n;
        type = r(i,2);
        ri = r(i,dir) * 10^-10;  % from A to m
        if(type == 1)
% fprintf('element = Zn  charge = 2\n');
            sum = sum + (z1 * e) * ri;
        elseif (type ==2)
% fprintf('element = S   charge = -2\n');
            sum = sum + (z2 * e) * ri;
        end
    end
    P3 = sum/V;
    P3 = P3 / (10^-10)^3;
```

```
end

% Calculate coordinate fraction u
function u = Uk(r)
    u = 1;
    n = length(r(:,1));
    sum = 0;
    counter = 0;
    for i = 1:n
        id1   = r(i,1);
        type1 = r(i,2);
        for j = 1:n
            id2 = r(j,1);
            type2 = r(j,2);
            if (type1 == 1 && type2 == 2)
                x1 = r(i,3);
                y1 = r(i,4);
                z1 = r(i,5);

                x2 = r(j,3);
                y2 = r(j,4);
                z2 = r(j,5);
                if abs(x2-x1) < 0.3 && abs(y2-y1) < 0.3 &&
abs(z2-z1) <= 2.6
                    d = sqrt( (z2-z1)^2 );
                    sum = sum + d;
                    counter = counter + 1;
                end
            end
        end
        u = sum/counter;
    end
end

function [P3] = Polarization_CS(r,dir,V)
    e = 1.6021766208 * 10^-19;
% Charge of Core, Core, Shell, Shell
z = [2.00, 1.030610, 0.0, -3.030610];
%       z = [2.00, -2.0, 0.0, 0.0];
    n = length(r(:,1));
    sum = 0;
    for i = 1:n;
        type = r(i,2);
        ri = r(i,dir) * 10^-10;  % from A to m
```

```
            sum = sum + (z(type) * e) * ri;    % C.m
        end
        P3 = sum/V;
        P3 = P3 / (10^-10)^3;
end
```

APPENDIX D:

CODE USED TO CALCULATE ELASTIC CONSTANTS IN THIS THESIS

ELASTIC CONSTANT LAMMPS CODE

Displace.mod

```
# NOTE: This script should not need to be
# modified. See in.elastic for more info.
#
# Find which reference length to use

if "${dir} == 1" then &
   "variable len0 equal ${lx0}"
if "${dir} == 2" then &
   "variable len0 equal ${ly0}"
if "${dir} == 3" then &
   "variable len0 equal ${lz0}"
if "${dir} == 4" then &
   "variable len0 equal ${lz0}"
if "${dir} == 5" then &
   "variable len0 equal ${lz0}"
if "${dir} == 6" then &
   "variable len0 equal ${ly0}"

# Reset box and simulation parameters

clear
read_restart restart.equil
include potential.mod

# Negative deformation

variable delta equal -${up}*${len0}
if "${dir} == 1" then &
   "change_box all x delta 0 ${delta} remap units box"
if "${dir} == 2" then &
   "change_box all y delta 0 ${delta} remap units box"
if "${dir} == 3" then &
   "change_box all z delta 0 ${delta} remap units box"
if "${dir} == 4" then &
   "change_box all yz delta ${delta} remap units box"
if "${dir} == 5" then &
   "change_box all xz delta ${delta} remap units box"
if "${dir} == 6" then &
   "change_box all xy delta ${delta} remap units box"

# Relax atoms positions

minimize ${etol} ${ftol} ${maxiter} ${maxeval}
```

```
# Obtain new stress tensor

variable tmp equal pxx
variable pxx1 equal ${tmp}
variable tmp equal pyy
variable pyy1 equal ${tmp}
variable tmp equal pzz
variable pzz1 equal ${tmp}
variable tmp equal pxy
variable pxy1 equal ${tmp}
variable tmp equal pxz
variable pxz1 equal ${tmp}
variable tmp equal pyz
variable pyz1 equal ${tmp}

# Compute elastic constant from pressure tensor

variable C1neg equal ${d1}
variable C2neg equal ${d2}
variable C3neg equal ${d3}
variable C4neg equal ${d4}
variable C5neg equal ${d5}
variable C6neg equal ${d6}

# Reset box and simulation parameters

clear
read_restart restart.equil
include potential.mod

# Positive deformation

variable delta equal ${up}*${len0}
if "${dir} == 1" then &
   "change_box all x delta 0 ${delta} remap units box"
if "${dir} == 2" then &
   "change_box all y delta 0 ${delta} remap units box"
if "${dir} == 3" then &
   "change_box all z delta 0 ${delta} remap units box"
if "${dir} == 4" then &
   "change_box all yz delta ${delta} remap units box"
if "${dir} == 5" then &
   "change_box all xz delta ${delta} remap units box"
if "${dir} == 6" then &
   "change_box all xy delta ${delta} remap units box"

# Relax atoms positions

minimize ${etol} ${ftol} ${maxiter} ${maxeval}

# Obtain new stress tensor
```

```
variable tmp equal pe
variable e1 equal ${tmp}
variable tmp equal press
variable p1 equal ${tmp}
variable tmp equal pxx
variable pxx1 equal ${tmp}
variable tmp equal pyy
variable pyy1 equal ${tmp}
variable tmp equal pzz
variable pzz1 equal ${tmp}
variable tmp equal pxy
variable pxy1 equal ${tmp}
variable tmp equal pxz
variable pxz1 equal ${tmp}
variable tmp equal pyz
variable pyz1 equal ${tmp}

# Compute elastic constant from pressure tensor

variable C1pos equal ${d1}
variable C2pos equal ${d2}
variable C3pos equal ${d3}
variable C4pos equal ${d4}
variable C5pos equal ${d5}
variable C6pos equal ${d6}

# Combine positive and negative

variable C1${dir} equal 0.5*(${C1neg}+${C1pos})
variable C2${dir} equal 0.5*(${C2neg}+${C2pos})
variable C3${dir} equal 0.5*(${C3neg}+${C3pos})
variable C4${dir} equal 0.5*(${C4neg}+${C4pos})
variable C5${dir} equal 0.5*(${C5neg}+${C5pos})
variable C6${dir} equal 0.5*(${C6neg}+${C6pos})

# Delete dir to make sure it is not reused

variable dir delete
```

In.elastic

```
# Compute elastic constant tensor for a crystal
#
# Written by Aidan Thompson (Sandia, athomps@sandia.gov)
#
#  This script uses the following three include files.
#
#   init.mod      (must be modified for different crystal structures)
```

```
#                 Define units, deformation parameters and initial
#           configuration of the atoms and simulation cell.
#
#
#   potential.mod    (must be modified for different pair styles)
#               Define pair style and other attributes
#               not stored in restart file
#
#
#   displace.mod    (displace.mod should not need to be modified)
#              Perform positive and negative box displacements
#              in direction ${dir} and size ${up}.
#              It uses the resultant changes
#              in stress to compute one
#              row of the elastic stiffness tensor
#
#              Inputs variables:
#                   dir = the Voigt deformation component
#                         (1,2,3,4,5,6)
#              Global constants:
#                   up = the deformation magnitude (strain units)
#                   cfac = conversion from LAMMPS pressure units to
#                   output units for elastic constants
#
#
#  To run this on a different system, it should only be necessary to
#  modify the files init.mod and potential.mod. In order to calculate
#  the elastic constants correctly, care must be taken to specify
#  the correct units in init.mod (units, cfac and cunits). It is also
#  important to verify that the minimization of energy w.r.t atom
#  positions in the deformed cell is fully converged.
#  One indication of this is that the elastic constants are
insensitive
#  to the choice of the variable ${up} in init.mod. Another is to
check
#  the final max and two-norm forces reported in the log file. If you
know
#  that minimization is not required, you can set maxiter = 0.0 in
#  init.mod.
#
#  There are two alternate versions of displace.mod provided.
#  They are displace_restart.mod and displace_reverse.mod.
#  The former resets the box using a restart file while
#  the latter reverses the deformation. Copy whichever
#  one you like best to displace.mod.
#

include init.mod
include potential.mod
include NPT.mod
# Compute initial state
fix 3 all box/relax  iso 0.0
```

```
minimize ${etol} ${ftol} ${maxiter} ${maxeval}

variable tmp equal pxx
variable pxx0 equal ${tmp}
variable tmp equal pyy
variable pyy0 equal ${tmp}
variable tmp equal pzz
variable pzz0 equal ${tmp}
variable tmp equal pyz
variable pyz0 equal ${tmp}
variable tmp equal pxz
variable pxz0 equal ${tmp}
variable tmp equal pxy
variable pxy0 equal ${tmp}

variable tmp equal lx
variable lx0 equal ${tmp}
variable tmp equal ly
variable ly0 equal ${tmp}
variable tmp equal lz
variable lz0 equal ${tmp}

# These formulas define the derivatives w.r.t. strain components
# Constants uses $, variables use v_
variable d1 equal -(v_pxx1-${pxx0})/(v_delta/v_len0)*${cfac}
variable d2 equal -(v_pyy1-${pyy0})/(v_delta/v_len0)*${cfac}
variable d3 equal -(v_pzz1-${pzz0})/(v_delta/v_len0)*${cfac}
variable d4 equal -(v_pyz1-${pyz0})/(v_delta/v_len0)*${cfac}
variable d5 equal -(v_pxz1-${pxz0})/(v_delta/v_len0)*${cfac}
variable d6 equal -(v_pxy1-${pxy0})/(v_delta/v_len0)*${cfac}

# Write restart
unfix 3
write_restart restart.equil

# uxx Perturbation

variable dir equal 1
include displace.mod

# uyy Perturbation

variable dir equal 2
include displace.mod

# uzz Perturbation

variable dir equal 3
include displace.mod

# uyz Perturbation
```

```
variable dir equal 4
include displace.mod

# uxz Perturbation

variable dir equal 5
include displace.mod

# uxy Perturbation

variable dir equal 6
include displace.mod

# Output final values

variable C11all equal ${C11}
variable C22all equal ${C22}
variable C33all equal ${C33}

variable C12all equal 0.5*(${C12}+${C21})
variable C13all equal 0.5*(${C13}+${C31})
variable C23all equal 0.5*(${C23}+${C32})

variable C44all equal ${C44}
variable C55all equal ${C55}
variable C66all equal ${C66}

variable C14all equal 0.5*(${C14}+${C41})
variable C15all equal 0.5*(${C15}+${C51})
variable C16all equal 0.5*(${C16}+${C61})

variable C24all equal 0.5*(${C24}+${C42})
variable C25all equal 0.5*(${C25}+${C52})
variable C26all equal 0.5*(${C26}+${C62})

variable C34all equal 0.5*(${C34}+${C43})
variable C35all equal 0.5*(${C35}+${C53})
variable C36all equal 0.5*(${C36}+${C63})

variable C45all equal 0.5*(${C45}+${C54})
variable C46all equal 0.5*(${C46}+${C64})
variable C56all equal 0.5*(${C56}+${C65})
variable Bm     equal (1/3)*(${C11all}+2*(${C12all}))
variable Cpr    equal 0.5*(${C11all}-(${C12all}))

# For Stillinger-Weber silicon, the analytical results
# are known to be (E. R. Cowley, 1988):
#               C11 = 151.4 GPa
#               C12 = 76.4 GPa
#               C44 = 56.4 GPa

print "Elastic Constant C11all = ${C11all} ${cunits}"
```

```
print "Elastic Constant C22all = ${C22all} ${cunits}"
print "Elastic Constant C33all = ${C33all} ${cunits}"

print "Elastic Constant C12all = ${C12all} ${cunits}"
print "Elastic Constant C13all = ${C13all} ${cunits}"
print "Elastic Constant C23all = ${C23all} ${cunits}"

print "Elastic Constant C44all = ${C44all} ${cunits}"
print "Elastic Constant C55all = ${C55all} ${cunits}"
print "Elastic Constant C66all = ${C66all} ${cunits}"

print "Elastic Constant C14all = ${C14all} ${cunits}"
print "Elastic Constant C15all = ${C15all} ${cunits}"
print "Elastic Constant C16all = ${C16all} ${cunits}"

print "Elastic Constant C24all = ${C24all} ${cunits}"
print "Elastic Constant C25all = ${C25all} ${cunits}"
print "Elastic Constant C26all = ${C26all} ${cunits}"

print "Elastic Constant C34all = ${C34all} ${cunits}"
print "Elastic Constant C35all = ${C35all} ${cunits}"
print "Elastic Constant C36all = ${C36all} ${cunits}"

print "Elastic Constant C45all = ${C45all} ${cunits}"
print "Elastic Constant C46all = ${C46all} ${cunits}"
print "Elastic Constant C56all = ${C56all} ${cunits}"

print "Bulk Modulus BMall      = ${Bm}     ${cunits}"
print "Elastic Constant Cprall = ${Cpr}    ${cunits}"
```

Init.mod

```
# NOTE: This script can be modified for different atomic structures,
# units, etc. See in.elastic for more info.
#

# Define the finite deformation size. Try several values of this
# variable to verify that results do not depend on it.
variable up equal 1.0e-6

# Uncomment one of these blocks, depending on what units
# you are using in LAMMPS and for output

# metal units, elastic constants in eV/A^3
#units          metal
#variable cfac equal 6.2414e-7
#variable cunits string eV/A^3

# metal units, elastic constants in GPa
units      metal
```

```
variable cfac equal 1.0e-4
variable cunits string GPa

# real units, elastic constants in GPa
#units          real
#variable cfac equal 1.01325e-4
#variable cunits string GPa

# Define minimization parameters
variable etol equal 0.0
variable ftol equal 1.0e-10
variable maxiter equal 10000
variable maxeval equal 1000000
variable dmax equal 1.0e-6

# generate the box and atom positions using a diamond lattice


boundary   p p p

atom_style    full

variable      N equal 6

# Generate a Wurtzite crystal with 4 basis


variable      a    equal 3.8227
variable      c    equal 6.2607
variable      u    equal 0.3748

variable      c_a  equal $c/$a
variable      s_2  equal sqrt(3.0)/2.0
variable      u_2  equal ${u}+0.5
variable      1_3  equal 1/3
variable      2_3  equal 2/3


lattice  custom $a &
                a1 0.5 -${s_2} 0.0 &
                a2 0.5  ${s_2} 0.0  &
                a3 0.0  0.0  ${c_a} &
                basis ${1_3} ${2_3} 0.0 &
                basis ${2_3} ${1_3} 0.5 &
                basis ${1_3} ${2_3} $u  &
                basis ${2_3} ${1_3} ${u_2}


region          sregion prism 0 $N  0 $N  0 $N 0 0 0

create_box      2 sregion
```

```
# Basis 1 & 2 is Zn, Basis 3 Basis 4 is S

create_atoms     1 region sregion basis 3 2 basis 4 2
```

NPT.mod

```
variable                t equal 300
variable                p equal 0
velocity                all create $t 4928459

fix                     1 all npt temp $t 1.0 10 iso 0.0 0.0 100
fix                     2 all temp/rescale 1 $t 1.0 0.01 1.0

run                     10000
unfix                   1
unfix                   2
```

Potential.mod

```
# NOTE: This script can be modified for different pair styles
# See in.elastic for more info.

#include  init.mod

# Choose potential
#
pair_style               buck/coul/long/cs 12


# Zn-Zn
pair_coeff              * * 0.0 1.0 0.0 0.0

# Zn_shell - S_shell
pair_coeff              1 2 580.84615 0.400505 0.0

# S_shell - S_shell

pair_coeff              2 2 1199.78975 0.148604 0.0

pair_modify             shift yes


# atomic mass of Zn
mass                     1 65.38
# atomic mass of O
mass                     2 32.059

# charges of Zn
set                      type 1 charge  2.0
```

```
# charges of O
set                      type 2 charge -2.0

dielectric               1.0

pair_modify              shift yes
kspace_style             ewald/disp 1.0E-06
neighbor                 3.0 bin

# Setup neighbor style
neighbor 1.0 nsq
neigh_modify once no every 1 delay 0 check yes

# Setup minimization style
min_style       cg
min_modify      dmax ${dmax} line quadratic

dump        1  all xyz 10 ZnS.xyz
dump        2  all custom 10 ZnS_Custom.xyz type id x y z
dump_modify 2  sort  1

# Setup output
thermo          1

#  pxx,pyy,pzz,pxy,pxz,pyz = 6 components of pressure tensor

thermo_style custom step temp pe press pxx pyy pzz pxy pxz pyz lx ly
lz vol
thermo_modify norm no
```

APPENDIX E:

CODE USED TO CALCULATE BULK MODULUS IN THIS THESIS

BULK MODULUS LAMMPS CODE

```
units                   metal
boundary                p p p
atom_style              full

fix             csinfo all property/atom i_CSID

read_data       ZnS_3x3x3x2_dihedral_Coreshell.data fix csinfo NULL
CS-Info

group cores     type 1 2
group shells    type 3 4

neighbor        2.0 bin
comm_modify     vel yes

kspace_style            ewald 1.0e-6

pair_style              buck/coul/long/cs 12

# *-*
pair_coeff              * * 0.0 1.0 0.0 0.0

# Zn_core - S_shell
pair_coeff              1 4 672.288 0.39089 0.0

# S_shell - S_shell
pair_coeff              4 4 1200.0 0.149 0.0

pair_modify             shift yes

angle_style             harmonic_decay

#S_Shell - Zn_Core - S_Shell
angle_coeff             1 4714170 109.47 0.3 0.3

dihedral_style          charmm

dihedral_coeff          1 0.005 3 0 0.0

special_bonds           coul 0.0 0.0 1.0 angle yes dihedral yes

bond_style              harmonic
bond_coeff              1 0 0
bond_coeff              2 6.6513715 0
```

```
variable                l loop 21
variable                p index -10000 -9000 -8000 -7000 -6000 -5000 -
4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000 6000 7000 8000 9000
10000
variable                t index 1
#velocity               all create $t 4928459

compute CStemp all temp/cs cores shells
compute thermo_press_lmp all pressure thermo_temp

thermo_modify temp CStemp press thermo_press_lmp

velocity                all create $t 4928459 dist gaussian mom yes
rot no bias yes temp CStemp
velocity                all scale $t temp CStemp

fix                     5 all temp/berendsen $t $t 0.4
fix                     1 all nve
fix                     2 all temp/rescale 1 $t $t 0.01 1.0
fix                     3 all nph iso $p $p 1 drag 100
fix_modify              5 temp CStemp

fix_modify              3 press thermo_press_lmp # pressure for
correct kinetic scalar

variable                Vol  equal vol
variable                xl   equal lx/6
variable                yl   equal ly/6
variable                zl   equal lz/6
variable                Ec   equal pe/atoms

thermo_style            custom step press etotal lx ly lz vol
thermo                  100

run                     10000

print                   "$p ${Vol}" append ZnS_BM.dat screen yes

next                    p
clear
next                    l
jump                    ZnS_BM_input.text
```

BULK MODULUS MATLAB CODE

```
%*************************************************************
%Nonlinear Least Squares
%*************************************************************

function NLLS
warning off
    clc
    load ZnS_BM.dat
    r  =  ZnS_BM(:,:);

    P = r(:,1)
    V = r(:,2)

    BM = LS(P,V);
%*************************************************************
% Convert from bars to Pa
%*************************************************************

    BM_Pa = BM * 10^5
%*************************************************************
% Convert from Pa to GPa
%*************************************************************

    BM_GPa = BM_Pa * 10^-9
     s = sprintf('ZnS Bulk Modulaus = %s GPa',BM_GPa)

    plot(P,V,'k','LineWidth',5,'MarkerSize',10)

    xlabel('Pressure /
(GPa)','fontweight','bold','fontsize',15)
    ylabel('Volume / (angst
cubed)','fontweight','bold','fontsize',15)
    legend(s)

set(gca,'fontweight','bold',"linewidth",3,'fontsize',25)
    box on
    set(gca,'fontweight','bold','linewidth',3)
    set(gca,'fontsize',18)

end
```

```
%************************************************************
% Calculate Q Matrix
%************************************************************

function Q = Qm(c,v)
    n = length(v);
    for i=1:n
        Q(i,1) = v(i)*v(i);
        Q(i,2) = v(i);
        Q(i,3) = 1.0;
    end
end

%************************************************************
% Calculate r Vector
%************************************************************

function r = rv(v,p,c)
    n = length(v);
    r = zeros(n,1);
    for i = 1:n
        r(i) = p(i)-(c(1)*v(i)*v(i)+c(2)*v(i)+c(3));
    end
end


%************************************************************
% Calculate bulk modulus value
%************************************************************

function B_Ave = LS(p,v)
    c = [1;1;1];
    dc = 2*c;
    while (dc' * dc) > 10^-7
        Q = Qm(c,v);
        r = rv(v,p,c);
        dc = (Q'*Q)\(Q'*r);
        c = c + dc;
        fprintf('%f\t\t%f\t%f\n',c(1),c(2),c(3))
    end
    fprintf('\n     P                  V                B\n')
    n = length(v);
    for i = 1:n
```

```
        B(i) = -v(i)*(2*c(1)*v(i)+c(2));
        fprintf('%f\t\t%f\t\t%f\n',p(i),v(i),B(i))
    end
    B_Ave = sum(B)/n
end
```