THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF PHYSICS

FORCE FIELDS FOR METALLIC CLUSTERS AND NANOPARTICLES

NICOLE LEGENSKI
Spring 2012

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Electrical Engineering
with honors in Physics

Reviewed and approved* by the following:

Dr. Robert Forrey
Professor of Physics
Thesis Supervisor

Dr. Richard Robinett
Professor of Physics
Honors Advisor

* Signatures are on file in the Schreyer Honors College.

Abstract

Atomic force fields for small clusters of gold, silver, and copper atoms are necessary for performing molecular dynamic simulations. This work presents a method for developing these force fields and an analysis of the force fields we obtained. Specifically, a potential energy function form, a parameterization method, and training algorithms are presented. This thesis also provides a theoretical background of the project including discussions on Density Functional Theory, the concept of a force field, potential energy functions, and computational methods used.

Table of Contents

List of Figures

Acknowledgements

My contribution to this work would not have been possible without the long-term, patient support and guidance of Dr. Robert Forrey. I am and always will be grateful for the guidance that Dr. Forrey has given me throughout my undergraduate career and was most proud when he was elected a fellow of the American Physical Society in January of 2012—an honor reserved for the most brilliant contributors to the field. In addition to Dr. Forrey's contributions, much of the data and inspiration for this project was supplied by Dr. Hansong Cheng and his research team, most notably Dr. Chengang Zhou, at the National University of Singapore. Many thanks also go to Dr. Richard Robinett and Dr. Jeffery Schiano for reviewing this thesis, and special appreciation is given to Dr. Sandy Feinstein, without whose encouragement I would not have pursued honors research. This work was supported by National Science Foundation grant No. PHY-0854838.

[309 East Beaver Ave, State College, PA, 16801] [484-663-4956] [nvl5026@psu.edu]

Introduction

The evolution of structures and properties of metal clusters has been a subject of considerable interest for many years [1-22]. Clusters often serve as a convenient means for gaining a systematic understanding of bulk structure and properties. The importance of clusters, which can range from subnano scale to meso scale in size, is further underscored in view of recent developments in nanoscience and nanotechnology. As the cluster size increases, the structure and properties can undergo systematic and often dramatic changes. Modeling novel materials requires development of computational tools which can account for such behaviors. One such tool is a reactive force field (FF) obtained as gradients of a potential energy function (PEF).

A fast and reliable FF allows molecular dynamics simulations to be carried out for systems where first principle electronic structure-based simulations would be too computationally intensive to be of practical use. The FF may also be used to study the evolution of structure and properties of clusters and nanoparticles. Detailed understanding of the growth of metal nanostructures would allow particle sizes and shapes to be controlled during synthesis for a variety of applications including heterogeneous catalysis [23, 24] and the formation of metallic glasses which lack the long-range order of normal crystalline metals [25, 26]. Parameterization of the FF using bulk data may lead to significant errors for the energies of small clusters and nanoparticles. Since the formation process may involve the coalescence of nanostructures and clusters, it is desirable to develop analytic and transferable FFs that are capable of describing clusters and nanoparticles of all sizes and shapes.

Many different PEF forms have been introduced in the literature with varying success [27-50]. Whatever the chosen form of PEF, the ultimate success depends on a combination of flexibility and constraint from a training set of reliable benchmarks. A FF derived from a PEF containing only a few parameters with limited flexibility would likely be incapable of modeling important features of novel materials. On the other hand, the enhanced flexibility provided by the use of a large number of parameters may introduce unphysical behavior for cluster configurations that are not included in the training set. A balance is sought where a physically motivated form of the PEF is used together with a compact set of parameters which may be safely constrained with benchmark experimental or theoretical data. This is the approach taken in this thesis. We first describe the theory used to obtain the benchmark electronic structure data used to constrain the PEF. We then describe the general properties of an atomic FF followed by

the details of our specific model. Algorithms of our approach and a sample of our extensive testing are provided. The efficiency and reliability of our model are assessed and we conclude with a discussion of applications which could benefit from the use of our novel FF.

Theory

Density Functional Theory (DFT)

All information about an atomic system may be described by the Schrodinger equation. The Schrodinger equation facilitates calculating the quantum state or wavefunction (ψ) of the system, which is the most accurate description of that system. Results from the Schrodinger equation are called *ab initio* (Latin for *from first principles*). The time-independent Schrodinger equation for a single particle with mass *m* in a central potential *V(r)* is

$$E\psi(r) = -\frac{\hbar^2}{2m}\nabla^2\psi(r) + V(r)\psi(r) \qquad (1)$$

where $\hbar$ is Planck's constant. For atomic systems, the Born-Oppenheimer approximation allows the Schrodinger equation for the combined nuclear and electronic motion to be approximately separated into a nuclear Schrodinger equation which describes the molecular dynamics (at ordinary temperatures, this part may be further approximated using Newtonian physics) and an electronic Schrodinger equation, which is given by

$$\left\{-\frac{\hbar^2}{2m}\sum_j {\nabla_j}^2 - \sum_{j,l}\frac{Z_l e^2}{|r_j - R_l|} + \frac{1}{2}\sum_{j\neq j'}\frac{e^2}{|r_j - r_{j'}|} - E\right\}\Psi = 0 \qquad (2)$$

where r_j are the positions of the electrons, R_l and Z_l are the positions and atomic numbers of the nuclei, and *e* is the charge of an electron. The electronic Schrodinger equation is solved under the assumption that the light electrons are moving so much faster than the heavy nuclei that the nuclei may be treated as though they are stationary with respect to the electrons. The energy eigenvalue of the electronic part is then computed as a function of nuclear coordinates and used in the molecular dynamics part as an effective potential energy surface.

The Schrodinger equation, when solved allows observable quantities to be computed. However, often the only practical means of solving the Schrodinger equation is by using computational methods that

approximate the solution [52]. A few of these computational methods are: the Quantum Monte Carlo method, Peturbation theory, and the variational method. Determining lowest energy states based on these wavefunction methods gives very accurate results.

However, wave-function based results may only be found for small atomic systems where the number of electrons, $n < 20$. The Schrodinger equation for larger systems may be impractical to solve. Nevertheless, there are methods that facilitate accurate calculations of the properties for larger atomic systems. One such method is Density Functional Theory (DFT).

DFT is an *ab initio* method that seeks to describe a system in relation to its electron density *n(r)*. The electron density depends only on 3-spatial variables and is formally related to the $3N$-dependent wavefunction via the identity

$$n(r) = N \int \Psi^*(\mathrm{r}, \mathrm{r}_2, \dots, \mathrm{r}_\mathrm{N}) \Psi(r, r_2, \dots, r_N) dr_2 \dots dr_N \,. \qquad (3)$$

This identity is easily verified for small numbers of particles. For large-N, the wavefunction contains so many independent variables that it is impractical to compute it in order to obtain the density. Instead, Hohenberg and Kohn found that the Rayleigh-Ritz variational principle, which was conventionally formulated in terms of wavefunctions, could be formulated for the density function itself [52].

An early approximation of the electronic density was published by Hartree [53]. He proposed that *n(r)* could be well-approximated by treating each electron as if it were in independent motion. That is,

$$n(r) = \sum_{j=1}^{N} \left|\varphi_j(r)\right|^2 \qquad (4)$$

where $\varphi_j(r)$ is the eigenfunction of a single particle Schrodinger equation of the form given in equation (1). Then, by considering the electrons to be non-interacting, but under an external potential v(r), the variational energy of the system becomes

$$E_{v(r)}[\tilde{\mathrm{n}}] \equiv \int v(r)\tilde{\mathrm{n}}(r)dr + T_s[\tilde{\mathrm{n}}(r)] \qquad (5)$$

where $T_s[\tilde{\mathrm{n}}(r)]$ ≡ kinetic energy of the groundstate of non-interacting electrons with density distribution ñ(r). The groundstate energy is found by minimizing $E_{V(r)}[\tilde{\mathrm{n}}]$ with respect to $\tilde{\mathrm{n}}(r)$. To apply this method to a system of interacting electrons, the energy of the system becomes

$$E_{v(r)}[\tilde{\mathrm{n}}] \equiv \int v(r)\tilde{\mathrm{n}}(r)dr + T_s[\tilde{\mathrm{n}}(r)] + \frac{1}{2}\int \frac{\tilde{\mathrm{n}}(r)\tilde{\mathrm{n}}(r')}{|r-r'|} dr dr' + E_{xc}[\tilde{\mathrm{n}}(r)] \qquad (6)$$

where E_{xc} is the exchange-correlation energy functional that accounts for many-body effects. The exchange-correlation energy is often approximated by the Local Density Approximation (LDA). LDA is derived from the exchange-correlation energy of a uniform electron gas, and it depends only on the electron density at a particular coordinate.

A better approximation is the Generalized Gradient Approximation (GGA) which depends on the electron density as well as the gradients of the electron density. The particular GGA functional that we use is the Perdew-Wang exchange-correlation functional (PW91) as implemented in the DMol package [50]. DFT provides very accurate results from first principles. It is for this reason that we use DFT to obtain empirical data for our study of metallic systems.

Concept of a Force Field (FF)

A force field is a vector field, i.e. a function that produces a vector given a set of coordinates. For example, in a system of N particles, the force on the kth particle may be found by

$$\overrightarrow{F_k} = \overrightarrow{F_k}(\overrightarrow{r_1}, \overrightarrow{r_2}, \dots, \overrightarrow{r_N}). \qquad k = 1, 2, \dots, N \qquad (7)$$

Example 1. Distribution of Point Charges

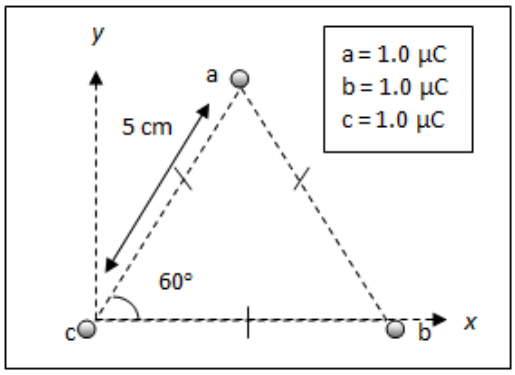
A fundamental example of a force field for a system of *n* point charges is described by Coulomb's law, where the force between each pair of point charges is given by

$$F(r_{12}) = k\,\frac{|q_1||q_2|}{{r_{12}}^2}. \qquad (8)$$

To find the total force on point charge 1 of the system, the vector sum of all forces between point charge 1 and every other point charge in the system is needed. That is,

$$\overrightarrow{F_{1,net}} = \overrightarrow{F_{1,2}} + \overrightarrow{F_{1,3}} + \overrightarrow{F_{1,4}} + \cdots + \overrightarrow{F_{1,n}} \qquad (9)$$

Figure 1: System of Point Charges



This is called the "superposition principle." By separating the forces into their respective x-, y-, and z-components, it is simple to find the sum of the forces in each direction.

Consider, for instance, a two-dimensional system of three identical point charges as shown in figure 1. To find the force on electron *c*, we use Coulumb's law to find the forces from *b* onto *c* and from *a* onto *c.* Then, using the superposition principle, the force on *c* is the sum of the forces from *a* and *b*.

$$\overrightarrow{F_{c,net}} = \overrightarrow{F_{c,a}} + \overrightarrow{F_{c,b}}$$

$$= k\,\frac{|q_c||q_a|}{{r_{ac}}^2}\widehat{r_{c,a}} + k\,\frac{|q_c||q_b|}{{r_{bc}}^2}\widehat{r_{c,b}}$$

$$= k * 10^{-12}C^2\left(\frac{1}{25\ cm^2}\measuredangle 240^o + \frac{1}{25\ cm^2}\measuredangle 180^o\right)$$

$$= k * 10^{-12}C^2(400\ m^{-2}\measuredangle 240^{\ o} + 400\ m^{-2}\measuredangle 180^o)$$

$$= k * 10^{-12}C^2(-200\ m^{-2}\hat{x} - 346\ m^{-2}\hat{y} - 400\ m^{-2}\hat{x})$$

$$= k * 10^{-12}C^2(-600\ m^{-2}\hat{x} - 346\ m^{-2}\hat{y})$$

$$= k * 10^{-12}C^2(692\ m^{-2}\measuredangle 210^o)$$

$$= 6.2N\measuredangle 210^o$$

The force field may also be obtained by taking the gradient of the electric potential energy of the system.

$$V = V(\overrightarrow{r_1}, \overrightarrow{r_2}, \dots, \overrightarrow{r_N}) \;\rightarrow\; \overrightarrow{F_k} = -\nabla_k V \qquad (10)$$

In molecular dynamics, the forces in question are used in Newton's second law of classical mechanics.

$$F_k = m_k\vec{a}_k \;\rightarrow\; -\nabla_k V = m_k\frac{d^2\vec{r}_k}{dt^2} \qquad (11)$$

Coulombic particles exhibit a pair potential

$$V(r_{12}) = k\,\frac{|q_1||q_2|}{r_{12}}. \qquad (12)$$

For a system of charged particles, the potential also obeys the superstition principle

$$V = V(\overrightarrow{r_1}, \overrightarrow{r_2}, \dots, \overrightarrow{r_n}) = \sum_i^n \sum_{j>i}^n V_c(r_{ij}) \qquad (13)$$

where V_c is the pair potential for point charges, and $r_{ij} = \left|\vec{r_i} - \vec{r_j}\right|$.

Therefore, for the example in Figure 1,

$$V = V(\vec{r_1}, \vec{r_2}, \dots, \vec{r_n}) = \sum_i^n \sum_{j>i}^n V_c(r_{ij}) = V_c(r_{ab}) + V_c(r_{ac}) + V_c(r_{bc}).$$

We will first solve for the force on particle *c* in the *x*-direction.

$$\overrightarrow{F_{x_c}} = -\frac{\partial}{\partial x_c} V_c(r_{ab}) - \frac{\partial}{\partial x_c} V_c(r_{ac}) - \frac{\partial}{\partial x_c} V_c(r_{bc})$$

$$= 0 - \frac{\partial V_c(r_{ac})}{\partial r_{ac}} \frac{\partial r_{ac}}{\partial x_c} - \frac{\partial V_c(r_{bc})}{\partial r_{bc}} \frac{\partial r_{bc}}{\partial x_c}$$

$$= \left(k \frac{|q_a||q_c|}{{r_{ac}}^2}\right) \frac{\partial \sqrt{(x_a - x_c)^2 + (y_a - y_c)^2 + (z_a - z_c)^2}}{\partial x_c} + \left(k \frac{|q_b||q_c|}{{r_{bc}}^2}\right) \frac{\partial \sqrt{(x_b - x_c)^2 + (y_b - y_c)^2 + (z_b - z_c)^2}}{\partial x_c}$$

$$= -\left(k \frac{|q_a||q_c|}{{r_{ac}}^2}\right)\left(\frac{x_a - x_c}{r_{ac}}\right) - \left(k \frac{|q_b||q_c|}{{r_{bc}}^2}\right)\left(\frac{x_b - x_c}{r_{bc}}\right)$$

$$= -\left(k \frac{1 * 10^{-12}\ C^2}{2.5 * 10^{-3}\ m^2}\right)\left(\frac{.025\ m}{.05\ m} + \frac{.05\ m}{.05\ m}\right)$$

$$= -(3.5952N)\frac{3}{2} = -5.4\ N(\hat{x})$$

which is the same result that we obtained previously using superposition.

In the same way, the force on particle *c* in the *y*-direction is

$$\overrightarrow{F_{y_c}} = -\left(k \frac{|q_a||q_c|}{{r_{ac}}^2}\right)\left(\frac{y_a - y_c}{r_{ac}}\right) - \left(k \frac{|q_b||q_c|}{{r_{bc}}^2}\right)\left(\frac{y_b - y_c}{r_{bc}}\right)$$

$$= -(3.5952N)\left(\frac{.025\ m * \tan 60^o}{.05\ m} + 0\right)$$

$$= -3.1N(\hat{y}).$$

In polar coordinates,

$$\overrightarrow{F_c} = -5.4\,N(\hat{x}) - 3.1N(\hat{y}) = 6.2N\measuredangle 210^o$$

Example 2. Distribution of Atoms

Metallic particles, which are the objects of this study, are not point charges. The clusters comprise atoms, which are more complex than point charges and do not satisfy the superposition principle for their pair potentials. That is,

$$V \neq \sum_i^n \sum_{j>i}^n V_A(r_{ij}) \qquad (14)$$

where V_A = atomic pair potential.

The reason that this equality does not hold is that every metallic atom possesses a nucleus and an electron cloud. While the nucleus is by far the more massive of the two, most of an atom's space is due to the electron cloud. Because the electrons do not often have a strong allegiance to their own metallic nucleus, there are many interactions between the electrons of the different atoms of the system. The effects of an atom's electron cloud, as well as other nuclei and electrons in the system, are called many-body effects. Thus, we add a correcting term

$$V = \sum_i^n \sum_{j>i}^n V_A(r_{ij}) + \sum_i^N V_i(\rho_i). \qquad (15)$$

The correcting term is an energy functional that, in general, depends on the local electron density. For bulk systems, the correction term is equal to the energy needed to embed an atom into a lattice. Potential energy functions of this form are therefore referred to as embedded-atom (EA) functions. Many-body effects are accounted for via the electron density, which depends on the geometry of the system. Rather than using DFT electron density, we use

$$\rho_i = \sum_{j\neq i} \phi(r_{ij}) \qquad (16)$$

and compute the pair potential $\phi(r_{ij})$ by fitting to the DFT data.

Finding an accurate potential energy function (PEF) is vital to obtaining a reliable force field. Therefore, this PEF must accurately predict the electronic potential energies of any arrangement of any number of metallic particles.

Form of the Force Field

Potential Energy Function (PEF)

The EA method that we use is the quantum Sutton-Chen (QSC) method, which is capable of providing accurate values for surface energies, vacancy energies, stacking fault, and cohesive energies in bulk limit [50]. The form is described by:

$$E = \sum_i \left[\frac{1}{2} \sum_{j \neq i} D_{ij} V(r_{ij}) - c_{ii} D_{ii} \rho_i^{\frac{1}{2}} \right] \quad (17)$$

$$V(r_{ij}) = \left(\frac{\alpha_{ij}}{r_{ij}} \right)^{p_{ij}} \quad (18)$$

$$\rho_i = \sum_{j \neq i} \left(\frac{\alpha_{ij}}{r_{ij}} \right)^{q_{ij}}. \quad (19)$$

Note that the term $c_{ii} D_{ii} \rho_i^{\frac{1}{2}}$ in equation 18 corresponds to the correcting term in the more general form in equation 16.

When computing the potential energy function, it is convenient to eliminate the van der Waals region (see figure 2). This is due to the need to increase computational efficiency and the fact that our DFT calculations are expected to be inaccurate in the van der Waals region. DFT, in principle, is capable of providing all ground-state properties including the long range van der Waals energies. However, both the LDA and GGA exchange-correlation functions do not completely capture the correlated motion of electrons arising from Coulomb interactions between distant nonoverlapping electronic systems. Therefore, we simply eliminate the long-range part of the potential using the cutoff function

$$f_c(r) = \frac{1}{2} \left[1 + \cos \left(\frac{\pi (r - r_{min})}{(r_{max} - r_{min})} \right) \right] \quad (20)$$

for interatomic distances greater than r_{min} and less than r_{max}. It is important to note that this method will be most successful if values for r_{MIN} and r_{MAX} are chosen to be greater than typical bond lengths of the system. For instance, the bond length for a diatomic Cu molecule is about 3.00 angstroms. By chosing r_{MIN} and r_{MAX} to be 3 and 5, respectively, we are able to capture the true minimum and maintain the approximate shape of the true potential while cutting off the Van der Waals region. In this study, we allow r_{max} and r_{min} to be fixed values of 5 and 3 angstroms, respectively.

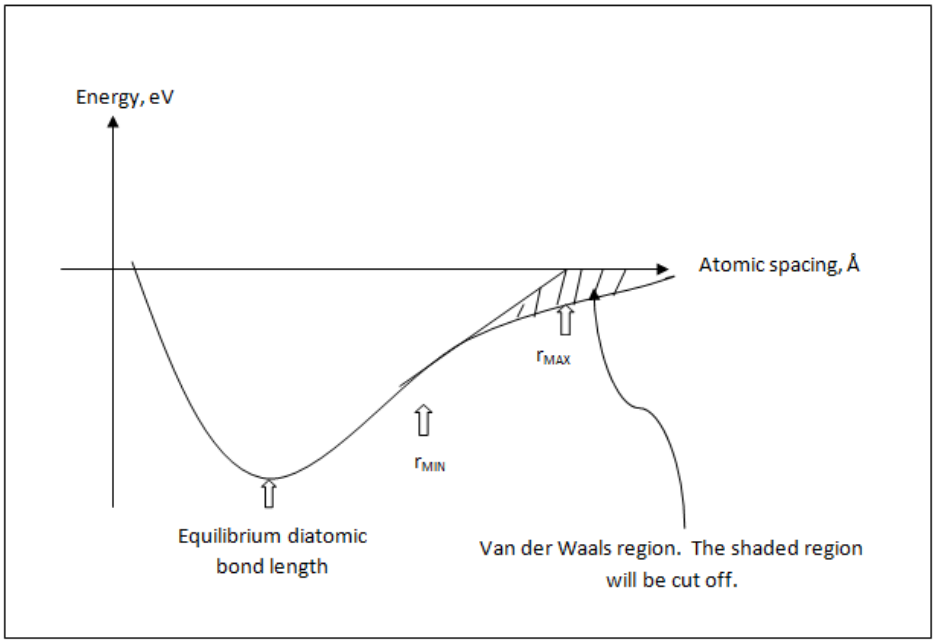


Figure 2: Van der Waals Cutoff

The QSC potential uses five parameters D, p, α, c, and q. We have found [Legenski] that the variation among the metallic clusters in size and shape cannot be accounted for using constant values of these five parameters. Therefore, we allow the five parameters to vary with the number of nearest neighbors by defining an effective coordination number

$$M_i = \sum_{j=1}^{N} l_c(r_{ij}) - 1. \qquad (21)$$

Here $l_c(r_{ij})$ is a local cutoff function which allows the model to disregard atoms that are far away from the atom in question. In our work, we have assumed the local cutoff function to be the same as the one used to cutoff the van der Waals region (see Figure 2). We then use the following scheme which allows the QSC parameters to linearly interpolate between small and large clusters:

$$D(M_i) = D_0 + (D_1 - D_0) \cdot Min(12, M_i)/12 \qquad (22)$$

$$c(M_i) = c_0 + (c_1 - c_0) \cdot Min(12, M_i)/12 \qquad (23)$$

$$\propto (M_i) = \propto_0 + (\propto_1 - \propto_0) \cdot Min(12, M_i)/12 \qquad (24)$$

$$p(M_i) = p_0 + (p_1 - p_0) \cdot Min(12, M_i)/12 \quad (25)$$

$$q(M_i) = q_0 + (q_1 - q_0) \cdot Min(12, M_i)/12. \quad (26)$$

The constant parameters D_0, c_0, D_1, c_1, etc, are determined through training the model with the monatomic DFT data. Equations 23 through 27 show how the parameters interpolate between the two extreme sets of parameters depending on how many atoms are in a particular locality.

The dependence of the parameters on the coordination number M_i allows for smooth interpolation between small clusters and the bulk limit. When M_i is small, D is close to D_0. But when M_i is large, as is the case with bulk structures, then D will be closer or equal to D_1. The same is true for the other parameters, such that they are always between their respective X_0 and X_1 parameter sets.

The parametrization scheme for binary systems is similar to that of the single metallic systems.

$$D(M_i, N_i) = D(M_i) + (D_2 - D_0) \cdot Min(12, N_i)/12 \quad (27)$$

$$c(M_i, N_i) = c(M_i) + (c_2 - c_0) \cdot Min(12, N_i)/12 \quad (28)$$

$$\alpha(M_i, N_i) = \alpha(M_i) + (\alpha_2 - \alpha_0) \cdot Min(12, N_i)/12 \quad (29)$$

$$p(M_i, N_i) = p(M_i) + (p_2 - p_0) \cdot Min(12, N_i)/12 \quad (30)$$

$$q(M_i, N_i) = q(M_i) + (q_2 - q_0) \cdot Min(12, N_i)/12 \quad (31)$$

In this case, M_i is the number of atoms of the same type as atom i within the cutoff function and N_i is the number of atoms of a different type as atom i. In this case, there are two sets of X_2 parameters: one for atom type i and one set for atom type j. These parameters are determined using the EA method trained with DFT data for binary alloy systems.

We then obtain the parameters necessary for equations 18 - 20 by using the following combination rules.

$$D_{ij} = \sqrt{D(M_i, N_i) \cdot D(M_j, N_j)} \quad (32)$$

$$c_{ij} = \frac{1}{2}[c(M_i, N_i) + c(M_i, N_i)] \quad (33)$$

$$\alpha_{ij} = \frac{1}{2}[\alpha(M_i, N_i) + \alpha(M_i, N_i)] \quad (34)$$

$$p_{ij} = \frac{1}{2}[p(M_i, N_i) + p(M_i, N_i)] \qquad (35)$$

$$q_{ij} = \frac{1}{2}[q(M_i, N_i) + q(M_i, N_i)] \qquad (36)$$

Equations 33 - 37 are satisfied by a binary system. For monatomic systems, N_i and N_j are zero, and equations 33-37 are reduced to equations 23-27.

Analytical Derivatives

Most recently, it has become an object of our study to find analytical derivatives to determine the force fields. In the past, we used a finite differences method to approximate the derivatives of the PEF [50]. The finite differences method of approximating the derivative uses the limit definition of a derivative.

$$\frac{df}{dx} = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h} \qquad (37)$$

which means for the ith particle,

$$\rightarrow F_{x_i} = \lim_{h\to 0} \frac{E(x_i+h,y_i,z_i)-E(x_i,y_i,z_i)}{h} \qquad (38)$$

and similarly for *y* and *z*- components. In our codes, *h* is given a value of $1*10^{-6}$. Because we have chosen a small value for *h*, the approximations are accurate. However, we would like to find the derivatives analytically to increase the computational efficiency in calculating the force field. While it is possible to find the necessary derivatives using mathematics software such as Maple or Mathematica, output from these codes could result in redundancies and inefficient algorithms for our application. We therefore find the derivatives by hand. While finding the derivatives of our PEF by hand requires little more than the knowledge from an introductory course in calculus, doing so is somewhat cumbersome.

To simplify matters, we explored the possibility of reducing the number of variable parameters in our PEF, thereby reducing the number of significant derivatives. In the past, we implemented an interpolation scheme to calculate five variable parameters. We now explore the possibility of reducing the number of variable parameters to two. That is, only the parameters *D* and *c* will be determined using the interpolation scheme, and the parameters α, *p*, and *q* will remain constant. While the form of the PEF is maintained, equations 13 through 15 will be reduced to

$$\alpha(M_i) = \alpha_0 = \alpha_1 \qquad (39)$$

$$p(M_i) = p_0 = p_1 \qquad (40)$$

$$q(M_i) = q_0 = q_1. \qquad (41)$$

After changing the number of parameters from 10 to 7, the PEF suffers slightly due to reduced flexibility (see chart). However, the change is not very significant, and the computational efficiency and simplicity gained is enough to justify using this simplified model. In addition to simplifying our model, it also simplifies the analytical derivatives. To find our force field, we solve the following equations:

$$-\frac{\partial E}{\partial x_1} = F_{x_1} \qquad (42)$$

$$-\frac{\partial E}{\partial y_1} = F_{y_1} \qquad (43)$$

$$-\frac{\partial E}{\partial z_1} = F_{z_1}. \qquad (44)$$

It is easy to see that solving these derivatives is greatly simplified because equations 7 and 8 are reduced to the terms

$$V(r_{ij}) = \left(\frac{\alpha}{r_{ij}}\right)^p \qquad (45)$$

$$\rho_i = \sum_{j\neq i}\left(\frac{\alpha}{r_{ij}}\right)^q. \qquad (46)$$

These terms are dependent only upon atomic spacing, eliminating the need for logarithms in the derivatives.

If we consider, for the ease of illustration, that all parameters are constant, then the PEF form for a system of three atoms is

$$E(\overrightarrow{r_1},\overrightarrow{r_2},\overrightarrow{r_3}) = \sum_{i=1}^{3} D\left[\frac{1}{2}\sum_{j\neq i} V(r_{ij}) - c_i\rho_i^{\frac{1}{2}}\right] \qquad (47)$$

$$E(\overrightarrow{r_1},\overrightarrow{r_2},\overrightarrow{r_3}) = D\left[\frac{1}{2}V(r_{12}) + \frac{1}{2}V(r_{13}) + \frac{1}{2}V(r_{21}) + \frac{1}{2}V(r_{23}) + \frac{1}{2}V(r_{31}) + \frac{1}{2}V(r_{32}) - c_1\rho_1^{\frac{1}{2}} - c_2\rho_2^{\frac{1}{2}} - c_3\rho_3^{\frac{1}{2}}\right]$$

$$\rho_1 = \varphi(r_{12}) + \varphi(r_{13})$$

$$\rho_2 = \varphi(r_{21}) + \varphi(r_{23})$$

$$\rho_3 = \varphi(r_{31}) + \varphi(r_{32})$$

$$\Rightarrow$$

$$\frac{\partial E}{\partial x_1} = D\left[\frac{\partial V(r_{12})}{\partial r_{12}}\frac{\partial r_{12}}{\partial x_1} + \frac{\partial V(r_{13})}{\partial r_{13}}\frac{\partial r_{13}}{\partial x_1} - \frac{1}{2}c_1{\rho_1}^{-\frac{1}{2}}\left(\frac{\partial \varphi(r_{12})}{\partial r_{12}}\frac{\partial r_{12}}{\partial x_1} + \frac{\partial \varphi(r_{13})}{\partial r_{13}}\frac{\partial r_{13}}{\partial x_1}\right) - \frac{1}{2}c_2{\rho_2}^{-\frac{1}{2}}\frac{\partial \varphi(r_{21})}{\partial r_{21}}\frac{\partial r_{21}}{\partial x_1} - \frac{1}{2}c_3{\rho_3}^{-\frac{1}{2}}\frac{\partial \varphi(r_{31})}{\partial r_{31}}\frac{\partial r_{31}}{\partial x_1}\right]$$

$$r_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2}$$

$$\frac{\partial r_{ij}}{\partial x_1} = \frac{\left(x_i - x_j\right)}{r_{ij}} = \frac{x_{ij}}{r_{ij}} = \frac{\partial r_{ji}}{\partial x_1}$$

$$V\left(r_{ij}\right) = \left(\frac{\propto}{r_{ij}}\right)^p \rightarrow \frac{\partial V(r_{ij})}{\partial x_1} = -p \propto^p {r_{ij}}^{-p-1}$$

$$\varphi\left(r_{ij}\right) = \left(\frac{\propto}{r_{ij}}\right)^q = \varphi\left(r_{ji}\right) \rightarrow \frac{\partial \varphi}{\partial x_1} = \frac{-q \propto^q}{{r_{ij}}^{q+1}}$$

$$\rightarrow \frac{\partial E_{const.}}{\partial x_1} = D\left[-p \propto^p {r_{12}}^{-p-2}x_{12} + -p \propto^p {r_{13}}^{-p-2}x_{13} - \frac{1}{2}c_1{\rho_1}^{-\frac{1}{2}}\left(\frac{-q \propto^q}{{r_{12}}^{q+2}}\ x_{12} + \frac{-q \propto^q}{{r_{13}}^{q+2}}x_{13}\right) - \frac{1}{2}c_2{\rho_2}^{-\frac{1}{2}}\frac{-q \propto^q}{{r_{12}}^{q+1}}\ x_{12} - \frac{1}{2}c_3{\rho_3}^{-\frac{1}{2}}\frac{-q \propto^q}{{r_{13}}^{q+1}}x_{13}\right].$$

Ultimately, when we apply the derivatives for the variable parameters *D* and *c*, the PEF derivative becomes

$$\rightarrow \frac{\partial E}{\partial x_1} = \frac{\partial E_{const.}}{\partial x_1} + \frac{1}{2}V(r_{12})\frac{\partial}{\partial x_1}(D_{12} + D_{21}) + \frac{1}{2}V(r_{13})\frac{\partial}{\partial x_1}(D_{13} + D_{31}) + \frac{1}{2}V(r_{23})\frac{\partial}{\partial x_1}(D_{23} + D_{32}) - {\rho_1}^{\frac{1}{2}}\frac{\partial}{\partial x_1}c_{11}D_{11} - {\rho_2}^{\frac{1}{2}}\frac{\partial}{\partial x_1}c_{22}D_{22} - {\rho_3}^{\frac{1}{2}}\frac{\partial}{\partial x_1}c_{33}D_{33} \qquad (48)$$

where

$$\frac{\partial}{\partial x_1}c_{11} = \left(\frac{c_1 - c_0}{12}\right)\frac{\partial M_i}{\partial x_1} + \left(\frac{c_2 - c_0}{12}\right)\frac{\partial N_i}{\partial x_1}$$

and similarly for derivatives with respect to y_1 and z_1. The above analytic formulas easily generalize to more than 3 atoms and may be conveniently programmed.

Training the Force Field

We obtain the PEF using an embedded atom (EA) method. An EA method produces good agreement with our benchmark data for large clusters of atoms. Specifically, we use the Quantum Sutton-Chen model [45] to obtain our PEF. Our benchmark data, which comprises thousands (see tables 1 and 2) of energy structures, is obtained using density function theory (DFT) calculations.

In order to see the improvement that may be achieved by our approach, we also consider a QSC PEF with constant parameters. We refer to the force fields which use constant parameters as FF0, and the force fields which use PEF's determined from equations 11-25 as FF1. Figures 1 and 2 compare FF0 (blue) and FF1 (red). The results show that FF1 produces consistently less error than the FF0.

We find that the clusters are too structurally different to adequately be represented using only one set of parameters. This may be because the geometries of very small clusters are often planar while larger structures are three-dimensional. The transitional trends from small to intermediate to large cluster sizes are best modeled by FF1.

We train the model by solving the equation

$$f(N) = \frac{\sum_m^{M_N} \omega_m \left| E_m^{DFT} - E_m^{FF} \right|}{N \sum_m^{M_N} \omega_m} \qquad (49)$$

and simultaneously minimizing the function

$$g = \frac{\sum_N^{Nmax} N f(N)}{\sum_N^{Nmax} N}. \qquad (50)$$

f(N) is the average error in electron volts per atom between the energy of a particular structure as determined from our benchmark data and the energy value given from our potential energy function. We use a quasi-Newtonian algorithm minimization routine to minimize g, thereby minimizing *f(N)* [51].

The weighting function ω_m causes structures with lower energies to be weighted more heavily than non-minima. The weighting function N helps to ensure appropriate limits as cluster sizes reach the bulk size. A unique feature of our potential energy functions is found when we minimize g. We include the energies from every cluster size, not each cluster size individually or exclusively bulk. Instead, this

minimization provides a smooth extrapolation from small cluster sizes, to intermediate cluster sizes, to bulk structures. This minimization weights the larger cluster sizes to provide the best results in the bulk cluster structures and reasonable, though less accurate, results for smaller cluster sizes.

Structures that are included in the training data are stable and metastable structures for the given metal as well as other random structures. The stable and metastable structures are equilibrium arrangements that have low energies. Stable structures have lower energies than metastable structures, though both are equilibrium arrangements of atoms. The training sets also have non-equilibrium structures.

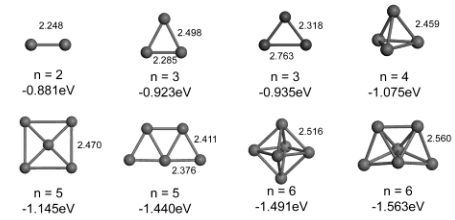


Figure 3: Stable and Metastable Structures

If the n-size corresponds to a "magic number" the structure may be icosahedral—that is, a spherical-like structure made of triangles. Some examples of magic numbers are 13, 19, 25, and 55. These numbers are considered "magic" numbers because they represent the numbers of particles that will perfectly enclosed a spherical structure with equilateral triangles. Clusters of magic-number sizes and similar sizes with near-icosahedral symmetry are included in the training set. These structures have low surface energy. However, the internal strain on the clusters precludes icosahedral structures from becoming very large.

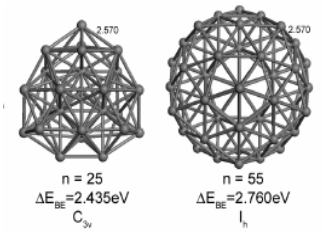


Figure 5: Magic Number Structures

Instead, larger structures will become face-centered-cubic (fcc) in shape. FCC is the crystalline structure that is found in which there are enough atoms to create long-range order (that is, an infinite number of unit cells in any direction). The fcc structure is considered the bulk limit because there must be many atoms to obtain the long-range order it requires. Although this structure has very low internal energies, the surface energy necessary to retain the cubic shape of the unit cell is too great to be maintained by only a few cells.

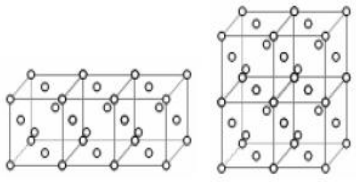


Figure 4: Face-Center-Cubic (FCC) Structures

Optimization

After we obtain what we believe to be an accurate PEF, it is necessary to test how this model will perform. To do this, we have developed an optimizer for the PEF. Specifically, we use the optimization to stretch and compress the bond lengths to search for new minimum energy structures. That is, the optimizer will fine-tune the coordinates of the atoms in a structure to find an arrangement that results in the lowest total energy. If the PEF and DFT are perfectly accurate, the minimum that the optimization indicates will be the same minimum that we received from DFT data.

A good example of an accurate PEF resulting in a well-behaved optimization is observed in Figure 6. The red line indicates energy values given by the PEF while the blue line shows the minimum energy values found by the optimizer. Specifically, the agreement between the PEF and the DFT is very good. Additionally, the optimization using the PEF agrees with the minimum energy DFT structures.

Inconsistencies between the PEF and the optimization can sometimes occur. It is possible that the PEF may be unconstrained in a region where the parametrization is too flexible. If the form of the PEF has many parameters, an optimization can reveal minima much lower than what is physically correct.

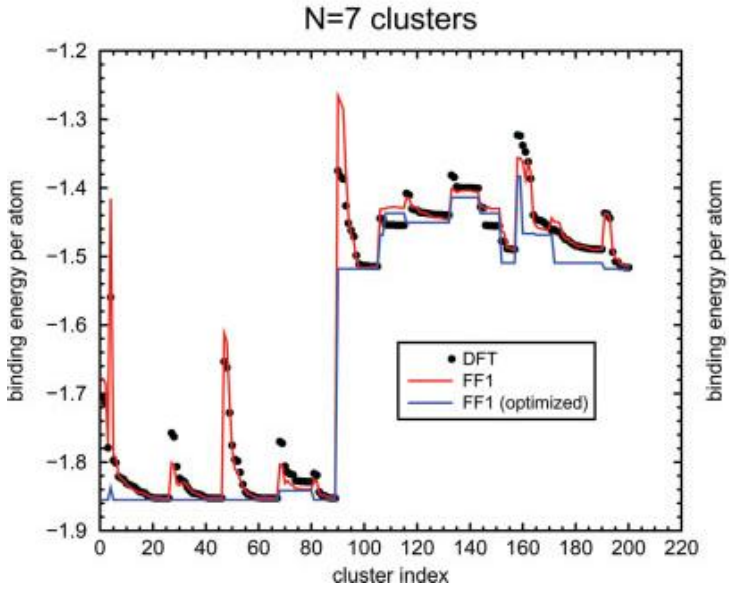


Figure 6: Optimization

When the optimization finds lower energy structures, it may mean that the lowest energy structure was not represented in the DFT training set or that not enough structures with compressed or stretched bonds were not included in the training set. In this case, we add the structure found by the optimization to our DFT data. We then retrain with the new structures in the training set.

Computer Algorithms

In this study, we write the necessary codes in Fortran, using a compiler for Linux-based systems. The basic algorithm for finding the parameters is solving for the error per atom, slightly changing the

parameters, and then adjusting the parameters via a minimization routine. Because we are minimizing equation 27, this process is repeated hundreds of times, obtaining a smaller error each time.

The parameters are changed in very small increments. This process is necessary to avoid violent oscillations of all of the parameters. Depending upon how much the error has changed in the immediately previous iterations, the changes in the parameters may be small. The following excerpt of code is responsible for changing the parameters before they are adjusted:

```
do i=1,2
    do j=3,3
    if(offset)then
      e(i,j)=e(i,j)-exp(-5.d0)
      c(i,j)=c(i,j)-exp(-5.d0)
      a(i,j)=a(i,j)-exp(-5.d0)
      p(i,j)=p(i,j)-exp(-5.d0)
      q(i,j)=q(i,j)-exp(-5.d0)
    else
      e(i,j)=e(i,j)-exp(-4.d0)
      c(i,j)=c(i,j)-exp(-4.d0)
      a(i,j)=a(i,j)-exp(-4.d0)
      p(i,j)=p(i,j)-exp(-4.d0)
      q(i,j)=q(i,j)-exp(-4.d0)
    endif
    end do
    end do
```

This set will reduce the current parameters by e^{-5}.

This set will reduce the current parameters by e^{-4}.

It is necessary to have two different step sizes for the parameters. The Boolean value *offset* in the above code is determined by the change in the error of the previous iterations. If the error does not improve after several successive iterations of the minimization, we assume that it has reached a minimum error. However, not all minimal errors found in this way are "absolute". That is, a minimum error found in this way may not be the smallest error obtainable. In order to determine if the error is, indeed, the most ideal, the code gives the parameters an extra "boost." That is, the parameters are reduced by e^{-4} rather than e^{-5}. This usually changes the parameters enough to find a lower, more ideal error. If, however, the error does not change, and the parameters return to the same arrangement, we then assume them to be the best parameters.

The binary force fields are determined in much the same way as the uniform atomic cases. We use the parameters for each of the monatomic substances in the system and extrapolate the results to create a new potential energy function. We then improve upon this model by constraining it with additional DFT data of binary clusters. We use DFT data of clusters of varying mixing ratios between substance i and

substance j (see tables in appendix). This method uses a linear dependence on the coordination numbers for each type of atom in the system performs well in the binary systems.

Discussion of Results

It is important to note that there is no perfect or unique set of parameters for the force field application. However, the parameters shown in Table 3 and Table 4 are the best that our training provides, and these parameters are used to derive the force fields. The quality of our results is best demonstrated by the error of the PEF's we develop. The graphs in the appendix show the average error per atom for each cluster size. The FF1 results are in red, and the FF0 results are in blue.

An important quality for a PEF to have is its ability to extrapolate reasonable results for cluster sizes that have not been included in its training set. For example, Figure 7 shows PEF error results for Au for cluster sizes between 2 and 147. However, only cluster sizes from 2 to 100 were included in the training set for Au. Even though data for cluster sizes 105 to 147 were not included, the error for these clusters is quite small and, in some respects, better than the results for the cluster sizes that are included in the training set. Well-behaved extrapolation results ensure that the PEF will be useful for larger clusters.

At first thought, an interpolation procedure that creates a binary PEF exclusively from the monatomic parameters of the respective elements should be accurate without needing extra training from binary data. Indeed, our first studies showed good results for Ag and Cu. However, Au binary substances were not met with the same success.

There are many unique synergy properties between Au and the other metals that cannot be properly modeled using the respective monatomic parameters. That is, the binary systems have their own properties that require their own sets of parameters to accurately depict. This observation attests to the unique characteristics specifically contributed to Au.

Conclusions

Force fields for simulating copper, silver, and gold clusters and nanoparticles were developed. Potential energy functions were obtained for both monatomic and binary metallic systems using an embedded atom method which uses an energy functional that depends on the local electron density at a given atomic site. Many cluster configurations of varying size and shape were used to constrain the parametrization for each system. Binding energies for these training clusters were computed using DFT.

Extensive testing showed that the many-body potentials are able to reproduce the DFT energies for most of the structures that were included in the training set. The force fields were used to calculate surface energies, bulk structures, and thermodynamic properties [50]. The results were found to be in good agreement with the DFT values and consistent with the available experimental data. Our initial efforts used a somewhat complicated procedure which required a large set of parameters with limited transferability. Our subsequent work uses a single form for the PEF and a greatly reduced set of parameters which provides greater transferability and efficiency. The general procedure appears to be robust and ready for targeted applications, such as the following:

- MD simulations in a canonical ensemble for clusters ranging from subnano to nano scales. The focus would be to obtain global minima. The force field would be used to obtain refined cluster structures at the global minima and to simulate the structural evolution of the clusters.

- Phase transitions. At an appropriate size under certain conditions (pressure/temperature), clusters may undergo a transition from one structure to another structure and eventually into bulk. This kind of structure transition is a common feature of structural evolution studies for clusters. However, to our knowledge, no direct investigations of the region near the transition have been performed. This is due to the large number of constituent atoms in nanoparticles that make *ab initio* quantum mechanical calculations prohibitively expensive to perform. Therefore, the only knowledge we have of these interesting transition regions comes from extrapolation or interpolation between limiting cases. Such indirect knowledge offers little insight into important material properties which ultimately limits efforts to understand the underlying mechanisms that control the formation of nanostructures. The current FF would allow a direct investigation of the behavior of large clusters near the transition region between cluster and bulk.

- Reactivity. The novel force field may be utilized to study the chemical reactivity of metal clusters, alloys, and surfaces toward molecules, which is directly associated with many technologically important applications such as heterogeneous catalysis, hydrogen storage, and chemical corrosion. Due to the technological importance of hydrogen storage, it would be desirable to perform simulations of light metal hydrides in order to understand the interactions between hydrogen atoms and the metal elements. This would allow practical schemes to be developed for controlling the release of molecular hydrogen at a relatively low temperature using appropriately designed metal alloys. Extensions to other

important technological applications could also be pursued such as the development of low cost and highly efficient metal catalysts for hydrogen dissociative chemisorption.

- Formation of metallic glasses. Metallic glasses are metal alloys with amorphous or glassy structures obtained from liquids of the constituent elements which do not crystallize during cooling. These materials exhibit novel physical properties that significantly differ from their corresponding crystalline forms. It is therefore important to understand their structures at atomistic detail and to quantify the structure-property relationship. The FF developed here would provide a unique opportunity to study such a relationship and would allow novel metallic glass materials to be designed with desired properties for specific applications.

- Thermodynamic and mechanical properties of metal alloys. The FF developed here would enable thermodynamic and mechanical properties of metal alloys to be evaluated by performing lattice dynamics simulations and calculations of bulk modulus. These properties are critically important for design of novel materials for applications such as corrosion resistant alloys, and would otherwise be difficult to obtain due to the complexity of alloy compositions and structures.

It is hoped that the work described in this thesis will be useful in such applications and provide a foundation for further investigations.

Appendix

References

1. **J. Raty, F. Gygi, G. Galli,** *Growth of carbon nanotubes on metal nanoparticles: a microscopic mechanism from ab inito molecular dynamics simulations*, Phys. Rev. Lett. 95, 096103 (2005).
2. **F. Ding, A. Rosen, and K. Bolton,** *Molecular dynamics study of the catalyst particle size dependence on carbon nanotube growth*, J. Chem. Phys. 121, 2775 (2004).
3. **O. A. Louchev, H. Kanda, A. Rosen, and K. Bolton,** *Thermal physics in carbon nanotube growth kinetics*, J. Chem. Phys. 121, 446 (2004).
4. **O. A. Louchev and J. R. Hester**, *Kinetic pathways of carbon nanotube nucleation from graphitic nanofragments*, J. Appl. Phys. 94, 2002 (2003).
5. **D. P. Pappas, A. P. Popov, A. N. Anisimov, B. V. Reddy, and S. N. Khanna,** *Spin configuration of Gd_{13} clusters*, Phys. Rev. Lett. 76, 4332 (1996).
6. **I. L. Billas, A. Chatelain, and W. A. de Heer,** *Magnetism from the Atom to the Bulk in Iron, Cobalt, and Nickel Clusters*, Science 265, 1982 (1994).
7. **S. E. Apsel, J. W. Emmert, J. Deng, and L. A. Bloomfield,** *Surface-enhanced magnetism in nickel clusters*, Phys. Rev. Lett. 76, 1441 (1996).
8. **Z. Fthenakis, A. N. Andriotis, and M. Menon,** *Temperature evolution of structural and magnetic properties of transition metal clusters*, J. Chem. Phys. 119, 10911 (2003).
9. **E. K. Parks, K. P. Kerns, S. J. Riley,** *The structure of Ni_{39}*, J. Chem. Phys. 109, 10207 (1998).
10. **E. K. Parks, G. C. Nieman, K. P. Kerns, and S. J. Riley**, *The thermodynamics of nitrogen adsorption on nickel clusters: Ni_{19}-Ni_{71}*, J. Chem. Phys. 108, 3731 (1998).
11. **D. Gerion, A. Hirt, M. L. Billas, A. Chatelain, and W. A. de Heer,** *Experimental specific heat of iron, cobalt, and nickel clusters studied in a molecular beam*, Phys. Rev. B 62, 7491 (2000).
12. **R. M. Olson, S. Varganov, M. S. Gordon, H. Metiu, S. Chretien, P. Piecuch, K. Kowalski, S. A. Kucharski, and M. Musial,** *Where does the planar-to-nonplanar turnover occur in small gold clusters?*, J. Am. Chem. Soc. 127, 1049 (2005).
13. **M. Hou, M. E. Azzaouri, H. Pattyn, J. Verheyden, G. Koops, and G. Zhang,** *Growth and lattice dynamics of CO nanoparticles embedded in Ag: a combined molecular dynamics simulation and Mossbauer study*, Phys. Rev. B 62, 5117 (2000).
14. **H. Cheng and L. Wang,** *Dimer growth, structural transition, and antiferromagnetic ordering of small chromium clusters*, Phys. Rev. Lett. 77, 51 (1996).

15. **L. Wang, H. Wu, and H. Cheng,** *Photoelectron spectroscopy of small chromium clusters: observation of even-odd alternations & theoretical interpretation*, Phys. Rev. B 55, 12884 (1997).
16. **K. Lee, J. Callaway, and S. Dhar,** *Electronic structure of small iron clusters*, Phys. Rev. B 30, 1724 (1984).
17. **K. Lee, J. Callaway, K. Kwong, R. Tang, and A. Ziegler,** *Electronic structure of small clusters of nickel and iron*, Phys. Rev. B 31, 1796 (1985).
18. **M. Castro, D. R. Salahub**, *Density functional calculations for small iron clusters*, Phys. Rev. B 49, 11842 (1984).
19. **E. L. Muetterties,** *Molecular Metal Clusters*, Science 196, 839 (1977).
20. **H. Conrad, G. Ertl, H. Knozinger, J. Kuppers, and E. E. Latta,** *Polynuclear metal carbonyl compounds and chemisorption of CO on metal surfaces*, Chem. Phys. Lett. 42, 115 (1976).
21. **K. M. Neyman, R. Sahnoun, C. Inntam, S. Hengrasmee, and N. Rosch,** *Computational Study of Model Pd-Zn Nanoclusters and Their Adsorption Complexes with CO Molecules*, J. Phys. Chem. B 108, 5424 (2004).
22. **S. Chretien, M. S. Gordon, and H. Metiu,** *Binding of propene on small gold clusters and on Au(111): rules for binding sites and relative binding energies*, J. Chem. Phys. 121, 3756 (2004).
23. **R. Whyman**, *Transition Metal Clusters*, ed. B. F. G. Johnson (Wiley & Sons, New York, 1980).
24. **H. Haberlandt**, *Theoretical Aspects of Heterogeneous Catalysis*, edited by J. B. Moffat (Van Nostrand Reinhold, New York, 1990).
25. **A. R. Yavari,** *A new order for metallic glasses,* Nature 439, 405 (2006).
26. **H. W. Sheng, W. K. Luo, F. M. Alamgir, J. M. Bai, and E. Ma,** *Atomic packing and short-to-medium-range order in metallic glasses,* Nature 439, 419 (2006).
27. **M. S. Daw and M. I. Baskes,** *Semiempirical, Quantum Mechanical Calculation of Hydrogen Embrittlement in Metals,* Phys. Rev. Lett. 50, 1285 (1983); Phys. Rev. B 29, 6443 (1984).
28. **M. W. Finnis and J. E. Sinclair,** *A simple empirical N-body potential for transition metals,* Philos.Mag. A 50, 45 (1984).
29. **R. P. Gupta**, *Lattice relaxation at a metal surface*, Phys. Rev. B 23, 6265 (1985).
30. **D. Tomanek, A. A. Aligia, and C. A. Balseiro,** *Calculation of elastic strain and electronic effects on surface segregation,* Phys. Rev. B 32, 5051 (1985).
31. **S. M. Foiles, M. I. Baskes, and M. S. Daw,** *Embedded-atom-method functions for the fcc metals Cu, Ag, Au, Ni, Pd, Pt, and their alloys,* Phys. Rev. B 33, 7983 (1986).
32. **F. Ercolessi, M. Parrinello, and E. Tosatti**, *Simulation of gold in the glue model,* Philos. Mag. A 58, 213 (1988).
33. **R. A. Johnson,** *Analytic nearest-neighbor model for fcc metals,* Phys. Rev. B 37, 3924 (1988); 39, 12554 (1989); 41, 9717 (1990).
34. **A. P. Sutton, J. Chen**, *Long-range Finnis–Sinclair potentials,* Philos.Mag.Lett. 61, 139 (1990).
35. **M. I. Baskes,** *Modified embedded-atom potentials for cubic materials and impurities,* Phys. Rev. B 46, 2727 (1992).
36. **S. R. Nishitani, S. Ohgushi, Y. Inoue, and H. Adachi**, *Grain boundary energies of Al simulated by environment-dependent embedded atom method,* Mater. Sci. Eng. A309, 490 (2001).
37. **A. W. Jasper, P. Staszewski, G. Staszewska, N. E. Schultz, and D. G. Truhlar,** *Analytic Potential Energy Functions for Aluminum Clusters,* J. Phys. Chem. B 108, 8996 (2004);
38. **A. W. Jasper, N. E. Schultz, and D. G. Truhlar**, *Analytic Potential Energy Functions for Simulating Aluminum Nanoparticles*, J. Phys. Chem. B 109, 3915 (2005).
39. **Z. H. Li, D. Bhatt, N. E. Schultz, J. I. Siepmann, and D. G. Truhlar,** *Free Energies of Formation of Metal Clusters and Nanoparticles from Molecular Simulations: Al_n with n = 2–60,* J. Phys. Chem. C 111, 16227 (2007).

40. **A. C. T. van Duin, S. Dasgupta, F. Lorant, and W. A. Goddard III**, *ReaxFF:A Reactive Force Field for Hydrocarbons,* J. Phys. Chem. A 105, 9396 (2001).
41. **A. Strachan, A. C. T. van Duin, D. Chakraborty, S. Dasgupta, and W. A. Goddard III**, *Shock waves in high-energy materials: the initial chemical events in nitramine RDX,* Phys. Rev. Lett. 91, 098301 (2003).
42. **A. C. T. van Duin, A. Strachan, S. Stewman, Q. Zhang, X. Xu, and W. A. Goddard III**, J. Phys. Chem. A 107, 3083 (2003).
43. **Q. Zhang, T. Cagin, A. C. T. van Duin, W. A. Goddard III, Y. Qi, L. G. Hector**, *Adhesion and nonwetting-wetting transition in the Al/α-Al_2O_3 interface,* Phys. Rev. B 69, 045423 (2004).
44. **K. D. Nelson, A. C. T. van Duin, J. Oxgaard, W.-Q. Deng, and W. A. Goddard III**, *Development of the ReaxFF Reactive Force Field for Describing Transition Metal Catalyzed Reactions, with Application to the Initial Stages of the Catalytic Formation of Carbon Nanotubes,* J. Phys. Chem. A 109, 493 (2005).
45. **Y. Qi, T. Cagin, Y. Kimura, and W. A. Goddard III**, *Molecular dynamics simulations of glass formation and crystallization in binary liquid metals: Cu-Ag and Cu-Ni,* Phys. Rev. B 59, 3527 (1999).
46. **H.-J. Lee, T. Cagin, W. L. Johnson, and W. A. Goddard III**, *Criteria for formation of metallic glasses: the role of atomic size ratio,* J. Chem. Phys. 119, 9858 (2003).
47. **J. Tersoff,** *New empirical approach for the structure and energy of covalent systems,* Phys. Rev. B 37, 6991 (1988).
48. **D. W. Brenner**, *Empirical potential for hydrocarbons for use in simulating the chemical vapor deposition of diamond films,* Phys. Rev. B 42, 9458 (1990).
49. **C. Zhou, J. Wu, L. Chen, Y. Wang, H. Cheng, and R. C. Forrey,** *Force Field for Copper Clusters and Nanoparticles*, Journal of Computational Chemistry 30, 2255 (2009).
50. **N. Legenski, C. Zhou, Q. Zhang, B. Han, J. Wu, L. Chen, H. Cheng, and R. C. Forrey,** *Force fields for metallic clusters and nanoparticles,* J. Computational Chemistry 32, 1711 (2011).
51. **D. Kahaner, and M. B. Cleve**, *Numerical Methods And Software*, Englewood Cliffs, N.J.: Prentice Hall, (1989).
52. **Lewars, Errol G**., *Introduction to Quantum Mechanics in Computational Chemistry*, Springer Netherlands. (2011)
53. **Hartree DR**, Proc Camb Phil Soc 24:89, (1928)

Figures, Charts, and Tables

Table 1: Number of Training Binary Clusters

System	Number of Benchmark Clusters
Cu + Ag	4,930
Au + Ag	7,696
Au + Cu	7,958

Table 2: Number of training monatomic clusters

System	Number of Benchmark Clusters

Cu	3,526
Ag	7,240
Au	12,383

Table 3: FF1 Parameters

	i	D_i(eV)	c_i	α_i(Å)	p_i	q_i
Cu	0	0.13961	3.96788	2.68496	11.26408	10.24328
Cu	1	0.82566	1.93253	2.22857	7.96061	3.15717
Ag	0	0.25590	2.70378	2.86178	11.07230	10.42481
Ag	1	1.02952	1.51273	2.46059	8.68235	3.84122
Au	0	0.84105	1.67526	2.54055	12.74993	10.68084
Au	1	1.88295	1.14705	2.43205	9.83923	4.84162
Cu +Ag	2	0.72848	2.02081	2.28822	8.02453	2.48090
Cu + Au	2	0.92499	2.01610	2.38057	7.96796	2.67050
Ag + Cu	2	1.11780	1.57238	2.52451	8.00608	4.34362
Ag + Au	2	1.09787	1.70548	2.73388	8.61485	3.48297
Au + Cu	2	1.96652	1.29905	2.43940	9.35256	7.98669
Au + Ag	2	2.07570	1.42034	2.36455	9.48098	10.61060

Table 4: FF0 Parameters

	D_i(eV)	c_i	α_i(Å)	p_i	q_i
Cu	0.97251	1.25718	2.03707	12.51465	2.54495
Ag	0.52735	1.67790	2.47532	12.45291	1.80458
Au	0.65415	1.82580	2.54173	12.31934	3.55212

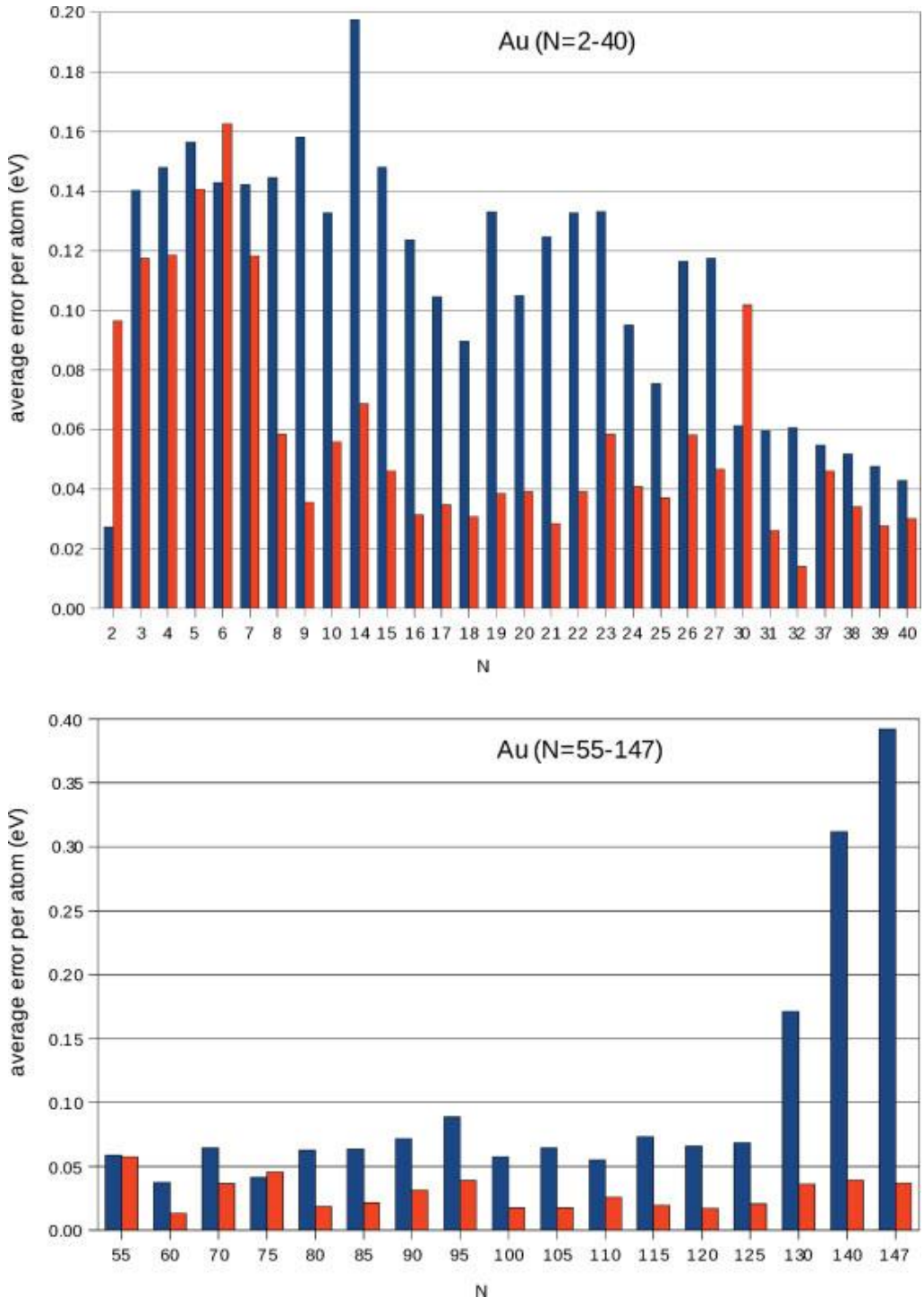


Figure 7: PEF Gold Results

These graphs show the error of our derived PEF when we compare it to our benchmark DFT data. The blue results are from a PEF with no coordination number dependency, and the red results are from our PEF with all parameters dependent on the coordination number. We observe that, with the exception of a few small clusters, a coordination number dependency greatly increases the accuracy of the PEF for Au systems.

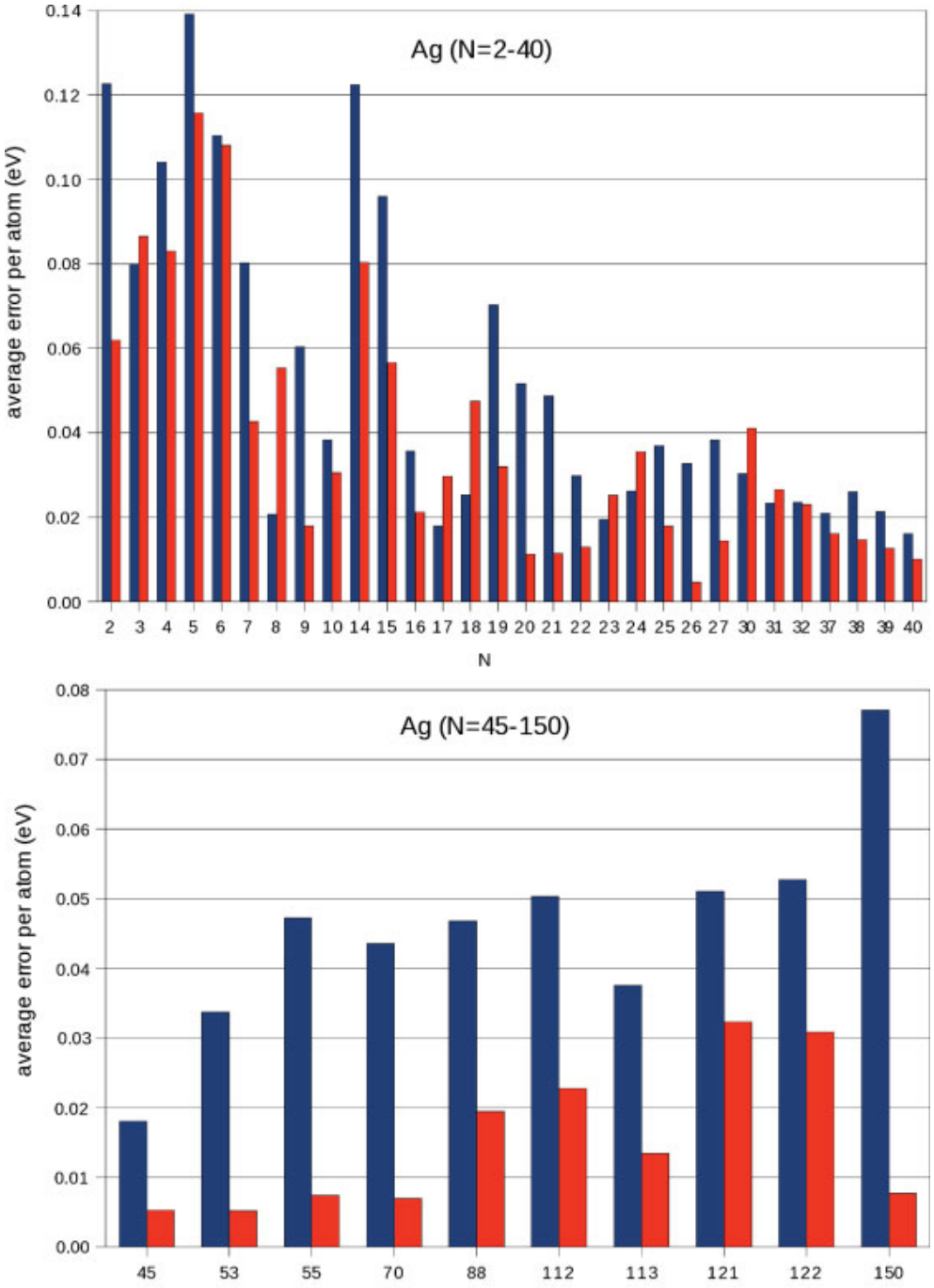


Figure 8: PEF Silver Results

These graphs show the error of our derived PEF when we compare it to our benchmark DFT data. The blue results are from a PEF with no coordination number dependency, and the red results are from our PEF with all parameters dependent on the coordination number. We observe that the coordination number dependence increases the accuracy of the PEF, especially for larger cluster sizes.

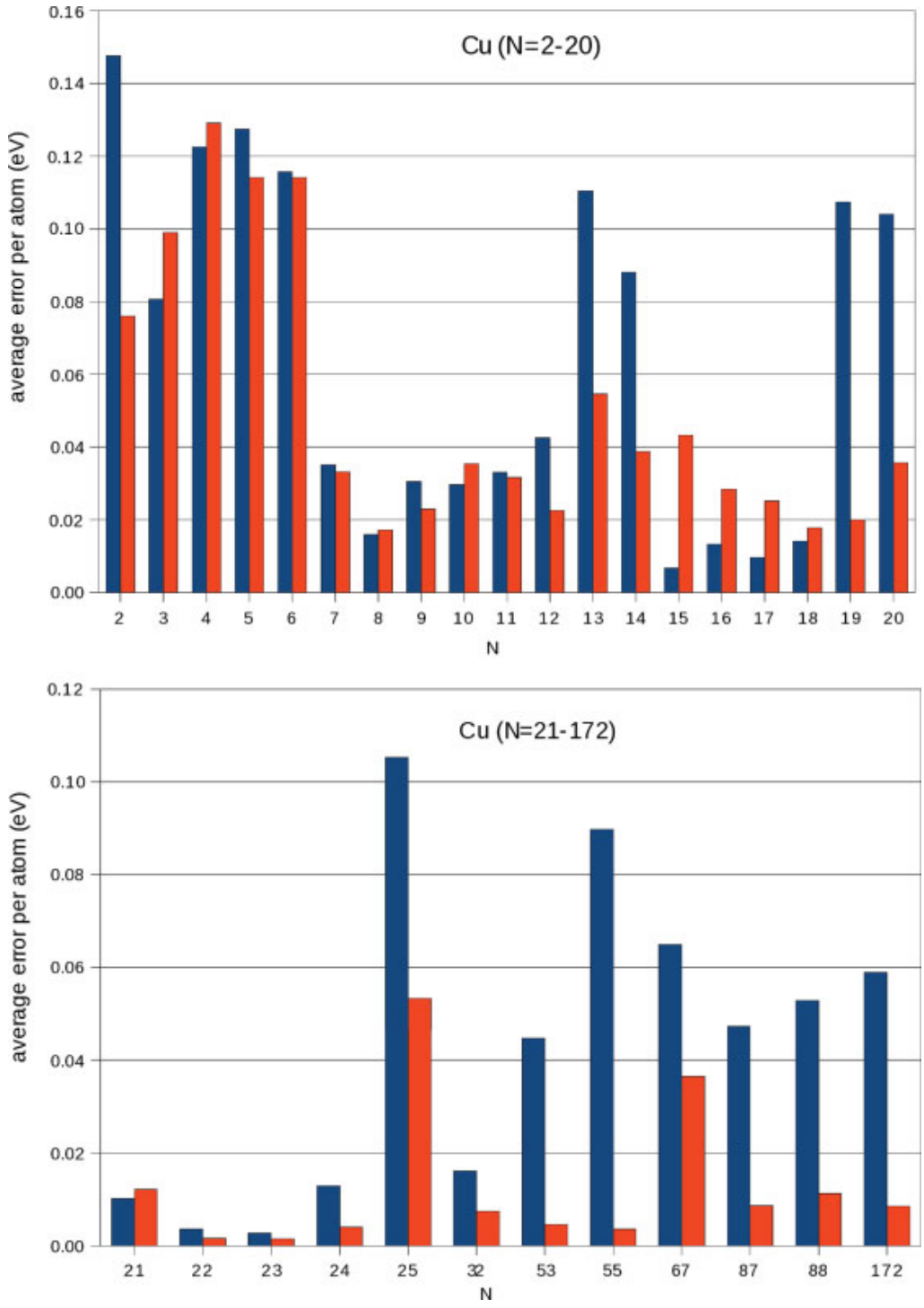


Figure 9: PEF Copper Results

These graphs show the error of our derived PEF when we compare it to our benchmark DFT data. The blue results are from a PEF with no coordination number dependency, and the red results are from our PEF with all parameters dependent on the coordination number. We observe that the coordination number dependence increases the accuracy of the PEF, especially for larger cluster sizes.

Further Computational Algorithms

Code that derives Force Fields

```
c*********************************************************************
c
c  Compute forcefield (FF) for Au-Cu cluster
c  nsize - size of cluster
c  ndata - number of cluster configurations
c  x,y,z - coordinates of atoms in the cluster
c  energy - potential energy at (x,y,z)
c  ffx,ffy,ffz - components of the FF
c234567
      implicit none
      integer i,j,n,nsize,nmax,ndata
      parameter (nmax=4000,ndata=1000)
      integer ntype(nmax)
      real*8 x(nmax),y(nmax),z(nmax),edft,energy,
     #       ffx(nmax),ffy(nmax),ffz(nmax),
     #       e(2,3),c(2,3),a(2,3),p(2,3),q(2,3),
     #       rmin(2,3),rmax(2,3)
      character*2 atom
      common/params/e,c,a,p,q,rmin,rmax

      open(10,file='parameters',status='old')
      open(20,file='ffield.in',status='old')
      open(30,file='ffield.out',status='unknown')
      open(40,file='ffield.dump',status='unknown')

      do i=1,2
      read(10,*)
      do j=1,3
        read(10,'(7(f10.5))')e(i,j),c(i,j),a(i,j),p(i,j),q(i,j),
     #                   rmin(i,j),rmax(i,j)
      end do
      end do

      do 100 n=1,ndata

      read(20,*)nsize
      read(20,*)edft   !  DFT energy
      if(nsize.gt.nmax)then
        write(*,*)'increase nmax and try again'
        stop
      endif

      do i=1,nsize
        read(20,*)atom,x(i),y(i),z(i)
      if(atom.eq.'Cu')then
```

```
        ntype(i)=1
      elseif(atom.eq.'Au')then
        ntype(i)=2
      else
        write(*,*)'error in atom type'
        stop
      endif
      end do

      call ffield1(nsize,ntype,x,y,z,energy,ffx,ffy,ffz)  ! numerical
c     call ffield2(nsize,ntype,x,y,z,energy,ffx,ffy,ffz)  ! analytical

      write(30,*)n,nsize,edft/nsize,energy
      do i=1,nsize
        write(40,'(6(e10.1,2x))')x(i),y(i),z(i),ffx(i),ffy(i),ffz(i)
      end do

  100  continue

      stop
      end

c*********************************************************************
c
c  Code to evaluate the FF for a binary cluster
c  Uses finite differences to approximate gradients
c234567
      subroutine ffield1(nsize,ntype,x,y,z,energy,ffx,ffy,ffz)
      implicit none
      integer i,nmax,nsize,ntype(nsize)
      parameter (nmax=4000)
      real*8 x(nsize),y(nsize),z(nsize),energy,ex,ey,ez,
     #       ffx(nsize),ffy(nsize),ffz(nsize),
     #       xp(nmax),yp(nmax),zp(nmax),eps
      parameter (eps=1.d-6)

      if(nsize.gt.nmax)write(*,*)'dimension error in ffield'

      call pef(nsize,ntype,x,y,z,energy)   ← Calculates potential energy for coordinates x, y, and z
      do i=1,nsize
      xp(i)=x(i)
      yp(i)=y(i)
      zp(i)=z(i)
      end do
      do i=1,nsize
      xp(i)=x(i)+eps
      yp(i)=y(i)+eps
      zp(i)=z(i)+eps
      call pef(nsize,ntype,xp,y,z,ex)   ← Calculates potential energy for coordinates (x+eps), y, z
      call pef(nsize,ntype,x,yp,z,ey)   ← Calculates potential energy for coordinates x, (y+eps), z
```

```
      call pef(nsize,ntype,x,y,zp,ez)    ← Calculates potential energy for coordinates x, y, (z+eps)
      ffx(i)=-(ex-energy)/eps    ←Calculates the derivative of potential energy (FF) in x-direction
      ffy(i)=-(ey-energy)/eps    ←Calculates the derivative of potential energy (FF) in y-direction
      ffz(i)=-(ez-energy)/eps    ←Calculates the derivative of potential energy (FF) in z-direction
      xp(i)=x(i)
      yp(i)=y(i)
      zp(i)=z(i)
      end do

      return
      end

c*****************************************************************
c
c  Code to evaluate the FF for a binary cluster
c  Uses analytic formulas to evaluate gradients
c234567
      subroutine ffield2(nsize,ntype,x,y,z,energy,ffx,ffy,ffz)
      implicit none
      integer i,j,k,m,nmax,nsize,ntype(nsize)
      parameter (nmax=4000)
      real*8 x(nsize),y(nsize),z(nsize),energy,ex,ey,ez,
     #     ffx(nsize),ffy(nsize),ffz(nsize),rho(nmax),
     #     r(nmax,nmax),fc(nmax,nmax),gc(nmax,nmax),
     #     rhigh,rsq,gcm(nmax,nmax),gcn(nmax,nmax),
     #     mc(nmax),mcx(nmax),mcy(nmax),mcz(nmax),
     #     nc(nmax),ncx(nmax),ncy(nmax),ncz(nmax),
     #     etmp1(nmax),etmp2(nmax),ctmp1(nmax),ctmp2(nmax),
     #     e(2,3),c(2,3),a(2,3),p(2,3),q(2,3),rmin(2,3),
     #     rmax(2,3),rmin1,rmax1,rmin2,rmax2,rmin3,rmax3,
     #     eii,cii,e0,c0,a0,p0,q0,rij,eij,cij,aij,pij,qij,
     #     e1,c1,a1,p1,q1,e2,c2,a2,p2,q2,scale1,scale2,
     #     ei,ci,ai,pi,qi,ej,cj,aj,pj,qj,xij,yij,zij,
     #     sum,sum0,sum1,sum2,sum3,fcutoff,gcutoff,
     #     vr,va,dvax,dvay,dvaz,dvrx,dvry,dvrz,
     #     tmp1x,tmp1y,tmp1z,tmp2x,tmp2y,tmp2z,tmp,
     #     tmpx(nmax,nmax),tmpy(nmax,nmax),tmpz(nmax,nmax),
     #     vtmp1(nmax,nmax),vtmp2(nmax,nmax)
      common/params/e,c,a,p,q,rmin,rmax

      if(nsize.gt.nmax)write(*,*)'dimension error in ffield'

      do i=1,nsize
      mc(i)=0.d0
      nc(i)=0.d0
      mcx(i)=0.d0
      mcy(i)=0.d0
      mcz(i)=0.d0
      ncx(i)=0.d0
      ncy(i)=0.d0
```

```
ncz(i)=0.d0
rmin1=rmin(ntype(i),1)
rmax1=rmax(ntype(i),1)
rmin2=rmin(ntype(i),2)
rmax2=rmax(ntype(i),2)
rmin3=rmin(ntype(i),3)
rmax3=rmax(ntype(i),3)
do j=1,nsize
 rsq=(x(i)-x(j))**2+(y(i)-y(j))**2+(z(i)-z(j))**2
 r(i,j)=sqrt(rsq)
 rij=r(i,j)
 fc(i,j)=0.d0
 gc(i,j)=0.d0
 gcm(i,j)=0.d0
 gcn(i,j)=0.d0
 if(j.ne.i .and. rij.lt.rmax1)then
   fc(i,j)=fcutoff(rij,rmin1,rmax1)
   gc(i,j)=gcutoff(rij,rmin1,rmax1)
 endif
 if(j.ne.i .and. rij.lt.rmax2 .and. ntype(i).eq.ntype(j))then
   mc(i)=mc(i)+fcutoff(rij,rmin2,rmax2)
   gcm(i,j)=gcutoff(rij,rmin2,rmax2)
   xij=x(i)-x(j)
   yij=y(i)-y(j)
   zij=z(i)-z(j)
   mcx(i)=mcx(i)+gcm(i,j)*xij/rij
   mcy(i)=mcy(i)+gcm(i,j)*yij/rij
   mcz(i)=mcz(i)+gcm(i,j)*zij/rij
 endif
 if(j.ne.i .and. rij.lt.rmax3 .and. ntype(i).ne.ntype(j))then
   nc(i)=nc(i)+fcutoff(rij,rmin3,rmax3)
   gcn(i,j)=gcutoff(rij,rmin3,rmax3)
   xij=x(i)-x(j)
   yij=y(i)-y(j)
   zij=z(i)-z(j)
   ncx(i)=ncx(i)+gcn(i,j)*xij/rij
   ncy(i)=ncy(i)+gcn(i,j)*yij/rij
   ncz(i)=ncz(i)+gcn(i,j)*zij/rij
 endif
end do
end do

do i=1,nsize
sum=0.d0
ai=a(ntype(i),1)
qi=q(ntype(i),1)
do j=1,nsize
  aj=a(ntype(j),1)
  qj=q(ntype(j),1)
  aij=(ai+aj)/2.d0
  qij=(qi+qj)/2.d0
```

```
  rij=r(i,j)
  if(j.ne.i)sum=sum+fc(i,j)*(aij/rij)**qij
end do
rho(i)=dsqrt(sum)  ! note change
end do

sum0=0.d0
do i=1,nsize
if(ntype(i).eq.1)rhigh=rmax(1,1)
if(ntype(i).eq.2)rhigh=rmax(2,1)

scale1=min(12,mc(i))/12.d0
scale2=min(12,nc(i))/12.d0

e0=e(ntype(i),1)
c0=c(ntype(i),1)
a0=a(ntype(i),1)
p0=p(ntype(i),1)
q0=q(ntype(i),1)
e1=e(ntype(i),2)
c1=c(ntype(i),2)
a1=a(ntype(i),2)
p1=p(ntype(i),2)
q1=q(ntype(i),2)
e2=e(ntype(i),3)
c2=c(ntype(i),3)
a2=a(ntype(i),3)
p2=p(ntype(i),3)
q2=q(ntype(i),3)
ei=e0+scale1*(e1-e0)+scale2*(e2-e0)
ci=c0+scale1*(c1-c0)+scale2*(c2-c0)
ai=a0
pi=p0
qi=q0

if(scale1.lt.1)then
  etmp1(i)=(e1-e0)/12.d0
  ctmp1(i)=(c1-c0)/12.d0
else
  etmp1(i)=0.d0
  ctmp1(i)=0.d0
endif

if(scale2.lt.1)then
  etmp2(i)=(e2-e0)/12.d0
  ctmp2(i)=(c2-c0)/12.d0
else
  etmp2(i)=0.d0
  ctmp2(i)=0.d0
endif
```

```
vr=0.d0
va=0.d0
dvrx=0.d0
dvax=0.d0
dvry=0.d0
dvay=0.d0
dvrz=0.d0
dvaz=0.d0
do j=1,nsize
rij=r(i,j)
xij=x(i)-x(j)
yij=y(i)-y(j)
zij=z(i)-z(j)

if(j.ne.i .and. rij.lt.rhigh)then

scale1=min(12,mc(j))/12.d0
scale2=min(12,nc(j))/12.d0

e0=e(ntype(j),1)
c0=c(ntype(j),1)
a0=a(ntype(j),1)
p0=p(ntype(j),1)
q0=q(ntype(j),1)
e1=e(ntype(j),2)
c1=c(ntype(j),2)
a1=a(ntype(j),2)
p1=p(ntype(j),2)
q1=q(ntype(j),2)
e2=e(ntype(j),3)
c2=c(ntype(j),3)
a2=a(ntype(j),3)
p2=p(ntype(j),3)
q2=q(ntype(j),3)
ej=e0+scale1*(e1-e0)+scale2*(e2-e0)
cj=c0+scale1*(c1-c0)+scale2*(c2-c0)
aj=a0
pj=p0
qj=q0

if(scale1.lt.1)then
  etmp1(j)=(e1-e0)/12.d0
  ctmp1(j)=(c1-c0)/12.d0
else
  etmp1(j)=0.d0
  ctmp1(j)=0.d0
endif

if(scale2.lt.1)then
  etmp2(j)=(e2-e0)/12.d0
  ctmp2(j)=(c2-c0)/12.d0
```

```
      else
        etmp2(j)=0.d0
        ctmp2(j)=0.d0
      endif

      eij=sqrt(ei*ej)
      cij=(ci+cj)/2.d0
      aij=(ai+aj)/2.d0
      pij=(pi+pj)/2.d0
      qij=(qi+qj)/2.d0
      cii=ci
      eii=ei

c compute FF for constant parameters

      vr=vr+fc(i,j)*eij*(aij/rij)**pij
      va=va+fc(i,j)*(cii*eii)**2*(aij/rij)**qij

      tmp=fc(i,j)*eij*pij*aij**pij/rij**(pij+2)
     #   -gc(i,j)*eij*aij**pij/rij**(pij+1)
      dvrx=dvrx+tmp*xij
      dvry=dvry+tmp*yij
      dvrz=dvrz+tmp*zij

      tmp=fc(i,j)*qij*aij**qij/rij**(qij+2)*ci*ei/rho(i)/2.d0
     #   +fc(i,j)*qij*aij**qij/rij**(qij+2)*cj*ej/rho(j)/2.d0
     #   -gc(i,j)*aij**qij/rij**(qij+1)*ci*ei/rho(i)/2.d0
     #   -gc(i,j)*aij**qij/rij**(qij+1)*cj*ej/rho(j)/2.d0
      dvax=dvax+tmp*xij
      dvay=dvay+tmp*yij
      dvaz=dvaz+tmp*zij

c add contribution from parameter derivatives

c attractive part

      tmp=etmp1(i)*gcm(i,j)+etmp2(i)*gcn(i,j)
      tmp1x=tmp*xij/rij
      tmp1y=tmp*yij/rij
      tmp1z=tmp*zij/rij

      tmp=ctmp1(i)*gcm(i,j)+ctmp2(i)*gcn(i,j)
      tmp2x=tmp*xij/rij
      tmp2y=tmp*yij/rij
      tmp2z=tmp*zij/rij

      dvax=dvax-rho(i)*(ci*tmp1x+ei*tmp2x)
      dvay=dvay-rho(i)*(ci*tmp1y+ei*tmp2y)
      dvaz=dvaz-rho(i)*(ci*tmp1z+ei*tmp2z)

      tmp=etmp1(j)*gcm(i,j)+etmp2(j)*gcn(i,j)
```

Calculates the derivatives of potential for constant parameters

Calculates derivatives for variable parameters that account for the attractive forces

```
      tmp1x=tmp*xij/rij
      tmp1y=tmp*yij/rij
      tmp1z=tmp*zij/rij

      tmp=ctmp1(j)*gcm(i,j)+ctmp2(j)*gcn(i,j)
      tmp2x=tmp*xij/rij
      tmp2y=tmp*yij/rij
      tmp2z=tmp*zij/rij

      dvax=dvax-rho(j)*(cj*tmp1x+ej*tmp2x)
      dvay=dvay-rho(j)*(cj*tmp1y+ej*tmp2y)
      dvaz=dvaz-rho(j)*(cj*tmp1z+ej*tmp2z)

      endif
      end do
      sum0=sum0+vr/2.d0-sqrt(va)
      sum1=dvrx-dvax
      sum2=dvry-dvay
      sum3=dvrz-dvaz
      energy=sum0/nsize
      ffx(i)=sum1/nsize
      ffy(i)=sum2/nsize
      ffz(i)=sum3/nsize
      end do


      return
      end

c*****************************************************************
c
c  Code to evaluate the PEF for a binary cluster
c
c234567
      subroutine pef(nm,ntype,x,y,z,energy)
      implicit none
      integer i,j,k,l,m,n,nmax,nm,ntype(*)
      parameter (nmax=4000)
      real*8 fcutoff,x(nm),y(nm),z(nm),r(nmax,nmax),
     #       fc(nmax,nmax),mc(nmax),nc(nmax),oc(nmax),
     #       vr,va,rsq,rij,eij,cij,aij,pij,qij,energy,
     #       e(2,3),c(2,3),a(2,3),p(2,3),q(2,3),sum,
     #       rmin(2,3),rmax(2,3),scale,rmin1,rmax1,
     #       rmin2,rmax2,rhigh,eii,cii,e0,c0,a0,p0,q0,
     #       e1,c1,a1,p1,q1,e2,c2,a2,p2,q2,scale1,
     #       ei,ci,ai,pi,qi,ej,cj,aj,pj,qj,scale2
      common/params/e,c,a,p,q,rmin,rmax

      if(nm.gt.nmax)write(*,*)'dimension error in pef'

      do i=1,nm
```

Calculates derivatives for variable parameters that account for the attractive forces

Combines derivatives due to attractive and repulsive forces

```
mc(i)=0.d0
nc(i)=0.d0
if(ntype(i).eq.1)m=1
if(ntype(i).eq.2)m=2
rmin1=rmin(m,1)
rmax1=rmax(m,1)
rmin2=rmin(m,2)
rmax2=rmax(m,2)
do j=1,nm
 rsq=(x(i)-x(j))**2+(y(i)-y(j))**2+(z(i)-z(j))**2
 r(i,j)=sqrt(rsq)
 rij=r(i,j)
 if(j.ne.i .and. rij.lt.rmax1)fc(i,j)=fcutoff(rij,rmin1,rmax1)
 if(j.ne.i .and. rij.lt.rmax2 .and. ntype(i).eq.ntype(j))then
   mc(i)=mc(i)+fcutoff(rij,rmin2,rmax2)
 endif
 if(j.ne.i .and. rij.lt.rmax2 .and. ntype(i).ne.ntype(j))then
   nc(i)=nc(i)+fcutoff(rij,rmin2,rmax2)
 endif
end do
end do

sum=0.d0
do i=1,nm
if(ntype(i).eq.1)rhigh=rmax(1,1)
if(ntype(i).eq.2)rhigh=rmax(2,1)

scale1=min(12,mc(i))/12.d0
scale2=min(12,nc(i))/12.d0
e0=e(ntype(i),1)
c0=c(ntype(i),1)
a0=a(ntype(i),1)
p0=p(ntype(i),1)
q0=q(ntype(i),1)
e1=e(ntype(i),2)
c1=c(ntype(i),2)
a1=a(ntype(i),2)
p1=p(ntype(i),2)
q1=q(ntype(i),2)
e2=e(ntype(i),3)
c2=c(ntype(i),3)
a2=a(ntype(i),3)
p2=p(ntype(i),3)
q2=q(ntype(i),3)
ei=e0+scale1*(e1-e0)+scale2*(e2-e0)
ci=c0+scale1*(c1-c0)+scale2*(c2-c0)
ai=a0+scale1*(a1-a0)+scale2*(a2-a0)
pi=p0+scale1*(p1-p0)+scale2*(p2-p0)
qi=q0+scale1*(q1-q0)+scale2*(q2-q0)

vr=0.d0
```

```
      va=0.d0
      do j=1,nm
        rij=r(i,j)
        if(j.ne.i .and. rij.lt.rhigh)then

      scale1=min(12,mc(j))/12.d0
      scale2=min(12,nc(j))/12.d0
      e0=e(ntype(j),1)
      c0=c(ntype(j),1)
      a0=a(ntype(j),1)
      p0=p(ntype(j),1)
      q0=q(ntype(j),1)
      e1=e(ntype(j),2)
      c1=c(ntype(j),2)
      a1=a(ntype(j),2)
      p1=p(ntype(j),2)
      q1=q(ntype(j),2)
      e2=e(ntype(j),3)
      c2=c(ntype(j),3)
      a2=a(ntype(j),3)
      p2=p(ntype(j),3)
      q2=q(ntype(j),3)
      ej=e0+scale1*(e1-e0)+scale2*(e2-e0)
      cj=c0+scale1*(c1-c0)+scale2*(c2-c0)
      aj=a0+scale1*(a1-a0)+scale2*(a2-a0)
      pj=p0+scale1*(p1-p0)+scale2*(p2-p0)
      qj=q0+scale1*(q1-q0)+scale2*(q2-q0)

      eij=sqrt(ei*ej)
      cij=(ci+cj)/2
      aij=(ai+aj)/2
      pij=(pi+pj)/2
      qij=(qi+qj)/2
      cii=ci
      eii=ei

         vr=vr+fc(i,j)*eij*(aij/rij)**pij
         va=va+fc(i,j)*(cii*eii)**2*(aij/rij)**qij
        endif
      end do
      sum=sum+vr/2.d0-sqrt(va)
      end do

      energy=sum/nm

      return
      end

c*************************************************************
c
c234567
```

```
      real*8 function fcutoff(r,rminshift,rmaxshift)
      implicit none
      integer index
      real*8 r,rmin,rmax,pi,arg,re,rminshift,rmaxshift

      pi=dacos(-1.d0)

      rmin=0.0d0+rminshift
      rmax=0.0d0+rmaxshift

      if(r.le.rmin)then
        fcutoff=1.d0
      elseif(r.gt.rmin .and. r.lt.rmax)then
        arg=pi*(r-rmin)/(rmax-rmin)
        fcutoff=(1.d0+cos(arg))/2.d0
      else
        fcutoff=0.d0
      endif

      return
      end

c*************************************************************
c
c234567
      real*8 function gcutoff(r,rminshift,rmaxshift)
      implicit none
      integer index
      real*8 r,rmin,rmax,pi,arg,re,rminshift,rmaxshift

      pi=dacos(-1.d0)

      rmin=0.0d0+rminshift
      rmax=0.0d0+rmaxshift

      if(r.le.rmin)then
        gcutoff=0.d0
      elseif(r.gt.rmin .and. r.lt.rmax)then
        arg=pi*(r-rmin)/(rmax-rmin)
        gcutoff=-pi/(rmax-rmin)*sin(arg)/2.d0
      else
        gcutoff=0.d0
      endif

      return
      end
```

Nicole V. Legenski | **2012**

[309 East Beaver Ave, State College, PA, 16801] [484-663-4956] [nvl5026@psu.edu]

Area of Study	Electrical Engineering Control and Power Engineering
Degree	B.S. in Electrical Engineering The Pennsylvania State University, State College, PA - Honors in Physics - International Undergraduate Research Experience
Publications	N. Legenski, C. Zhou, Q. Zhang, B. Han, J. Wu, L. Chen, H. Cheng, and R. C. Forrey, *Force fields for metallic clusters and nanoparticles*, J. Computational Chemistry 32 1711(2011).
Research	Atomic Force Field development