# SANDIA REPORT

SAND2007-WXYZ
Unlimited Release
Printed October 2007

# Development of an Inter-Atomic Potential for the Pd-H Binary System

Jonathan A. Zimmerman, Xiaowang Zhou, Joshua Griffin, Bryan M. Wong and Jeffrey J. Hoyt

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94-AL85000.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

**NOTICE:** This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

| | |
|---|---|
| Telephone: | (865) 576-8401 |
| Facsimile: | (865) 576-5728 |
| E-Mail: | reports@adonis.osti.gov |
| Online ordering: | http://www.osti.gov/bridge |

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

| | |
|---|---|
| Telephone: | (800) 553-6847 |
| Facsimile: | (703) 605-6900 |
| E-Mail: | orders@ntis.fedworld.gov |
| Online ordering: | http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online |

SAND2007-WXYZ
Unlimited Release
Printed October 2007

# Development of an Inter-Atomic Potential for the Pd-H Binary System

Jonathan A. Zimmerman and Xiaowang Zhou
Mechanics of Materials Department
jzimmer@sandia.gov, xzhou@sandia.gov

Joshua Griffin
Computational Sciences and Mathematics Department
jgriffi@sandia.gov

Bryan M. Wong
Materials Chemistry Department
bmwong@sandia.gov

Sandia National Laboratories
P.O. Box 969
Livermore, CA 94551

Jeffrey J. Hoyt
Department of Materials Science and Engineering
McMaster University
Hamilton, Ontario, Canada L8S 4L7
hoytj@mcmaster.ca

**Abstract**

Ongoing research at Sandia National Laboratories has been in the area of developing models and simulation methods that can be used to uncover and illuminate the material defects created during He bubble growth in aging bulk metal tritides. Previous efforts have used molecular dynamics calculations to examine the physical mechanisms by which growing He bubbles in a Pd metal lattice create material defects. However, these efforts focused only on the growth of He bubbles in pure Pd and not on bubble growth in the material of interest, palladium tritide (PdT), or its non-radioactive isotope palladium hydride (PdH). The reason for this is that existing inter-atomic potentials do not adequately describe the thermodynamics of the Pd-H system, which includes a miscibility gap that leads to phase separation of the dilute (alpha) and concentrated (beta) alloys of H in Pd at room temperature. This document will report the results of research to either find or develop inter-atomic potentials for the Pd-H and Pd-T systems, including our efforts to use experimental data and density functional theory calculations to create an inter-atomic potential for this unique metal alloy system. This research was funded by the Physics and Engineering Models sub-program of the Advanced Simulation and Computing program conducted at Sandia National Laboratories.

**Keywords: hydrogen, tritium, palladium, molecular dynamics, atomistic simulation, inter-atomic potential, force field, miscibility gap**

# Acknowledgment

The authors gratefully acknowledge input provided by discussions with M.I. Baskes, D.F. Cowgill, S.M. Foiles, C.D. Lorenz, M.G. Martin, T.K. Mattsson, S.L. Robinson, R.B. Schwarz, C.S. Snow, G.C. Story and R.T. Walters. Funding for this effort was generously provided by the Advanced Scientific Computing Physics & Engineering Models program.

# Contents


**1 Introduction** **15**

**2 Previous Models for the Pd-H System** **19**

2.1 Embeded Atom Method by Daw and Baskes . . . . . 19

2.2 Multi-body potentials by Puska *et al.*and Zhong *et al.* . . . . . 20

2.3 EAM model by Wolf *et al.* . . . . . 22

**3 Modifications to Daw and Baskes EAM** **29**

3.1 Approach . . . . . 29

3.2 Physical properties of the modified potential . . . . . 30

3.3 Mechanical properties of the modified potential . . . . . 32

3.4 Further comments . . . . . 37

**4 Numerical Optimization of EAM Potential Parameters** **41**

4.1 Methods . . . . . 41

4.1.1 DAKOTA . . . . . 42

4.1.1.1 Genetic algorithms . . . . . 43

4.1.1.2 Pattern searches . . . . . 43

4.1.1.3 Surrogate models . . . . . 45

4.1.1.4 Gradient based algorithms . . . . . 46

4.1.2 Towhee . . . . . 47

4.2 Results . . . . . 47

4.3 Concluding comments . . . . . 58

**5 Analytical Approach to EAM Parameter Fitting** **59**

5.1 Embedded Atom Method Potential . . . . . 60

5.2 Palladium potential . . . . . 61

5.3 Hydrogen Potential and Pair Interaction between Palladium and Hydrogen . . . . . 62

5.4 Palladium-Hydrogen Solid Solution Model . . . . . 64

5.4.1 Gibbs Free Energy of Mixing . . . . . 64

5.4.2 Cohesive Energy as a Function of Composition . . . . . 65

5.4.3 Bulk modulus B, and Elastic Constants $C'$ and $C_{44}$ . . . . . 68

5.5 Fitting Procedures . . . . . 69

5.6 Characteristics of the Potential . . . . . 74

5.6.1 EAM Functions . . . . . 74

5.6.2 Energy as a Function of Lattice Spacing . . . . . 74

5.6.3 Equilibrium Lattice Spacing . . . . . 77

5.6.4 Cohesive energy . . . . . 77

5.6.5 Gibbs Free Energy of Mixing . . . . . 78

5.6.6 Elastic Constants . . . . . 80

5.6.7 Mechanical Strength . . . . . 80

5.7 Molecular Dynamics Simulations . . . . . 83

5.8 Conclusions . . . . . 87

**6 Conclusions and Future Work** **89**

6.1 Conclusions . . . . . 89

6.2 Future work . . . . . 90

**References** **91**

# Appendix

**A FitEAM** **97**

**B User's Manual for FitEAM** **175**

B.1 Overview . . . . . 175

B.2 Terminology . . . . . 178

B.3 Structure of Mathematica notebook . . . . . 178

B.4 User's input values . . . . . 181

B.5 Trial-and-Error . . . . . 190

# List of Figures


1.1 The phase diagram for the Pd-H system. . . . . 16

2.1 Properties of the Pd-H system for the Daw-Baskes EAM potential at T=300K (a) Concentration vs Chemical Potential (b) Lattice Parameter vs Chemical Potential . . . . . 21

2.2 Monte Carlo results of hydrogen concentration vs. chemical potential for the Zhong *et al.*potential at 300K. The solid line is a guide to the eye and there is no indication of a miscibility gap. . . . . 23

2.3 Energy Minimization of PdH using the potential by Zhong *et al.*(a) Atomic configuration before minimization (b) Atomic configuration after minimization . . . . . 24

2.4 Pressure-composition isotherms for the $PdH_x$ alloy system produced with the Wolf *et al.*potential. (taken from [16]) . . . . . 25

2.5 Pressure-composition isotherms at 300K for the $PdH_x$ (□) and $PdT_x$ (+) alloy systems produced with the Wolf *et al.*potential. (taken from [16]) . . . . . 25

3.1 Unit cells for the stochiometries of $x = 0.25$, 0.5, 0.75 and 1 of the $PdH_x$ binary system. . . . . 29

3.2 Pair function for the Pd-H interaction . . . . . 32

3.3 Lattice parameter for the $PdH_x$ system. . . . . 33

3.4 Cohesive energy per unit cell for PdH. The black curve shows the results for the fitted potential while the red curve shows the VASP values. . . . . 33

3.5 Hydrogen concentration versus chemical potential for the fitted Pd-H potential, clearly showing a miscibility gap. . . . . 34

3.6 Atomic configuration of PdH after 100 ps at T = 300 K. . . . . 34

3.7 Atomic configurations for the stochiometries of $x = 0.25$, 0.5, 0.75 and 1 of the $PdH_x$ binary system. The axes shown for each configuration denotes the $< 100 >$ crystal directions. . . . . 35

3.8 (a) Shear $C'$ and (b) bulk $B$ elastic constants measured by Schwarz *et al.*[27] for the $PdH_x$ (△), $PdD_x$ (◯) and $PdT_x$ (□) alloy systems. (taken from [27]) . . . . . 37

3.9 System energy as a function of lattice parameter for H placed at tetrahedral and octahedral interstitial sites. (a) Dilute concentration of 1 H atom and 32 Pd atoms. (b) $PdH_{0.25}$ concentration with 1 H atom and 4 Pd atoms. . . . . 39

4.1 Schematic of a genetic algorithm optimization sequence. . . . . 43

4.2 Schematic of a pattern search optimization sequence. . . . . 44

4.3 Schematic of a surrogate optimization sequence. . . . . 45

4.4 First iteration for EAM of Pd-H system (a) $\phi_{\text{Pd-H}}$ (b) $\phi_{\text{H-H}}$ (c) $f_{\text{H}}$ (d) $F_{\text{H}}$ . . . . . 48

4.5 Energy per atom versus lattice parameter for first iteration of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H . . . . . 49

4.6 Energy per atom versus lattice parameter for second iteration of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H . . . . . 51

4.7 Energy of mixing as a function of H concentration. . . . . 52

4.8 Energy per atom versus lattice parameter for various optimization methods applied to fitting of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H . . . . . 53

4.9 A one-dimensional hypothetical objective function that when far from the global minimum is easy to reduce, but becomes increasingly difficult for smaller values of $OF$. . . . . 54

4.10 Energy per atom versus lattice parameter for fitting of EAM of Pd-H system (a) 50% H (b) 75% H . . . . . 55

4.11 Energy per atom versus lattice parameter for fitting of EAM of Pd-H system (a) 25% H, (b) 50% H, (c) 75% H, (d) 100% H. . . . . 57

5.1 Two $PdH_x$ structures with (a) hydrogen at octahedral sites and (b) hydrogen at tetrahedral sites. . . . . 67

5.2 Shear moduli (a) $C$' and (b) $C_{44}$ measured by Schwarz *et al.*[27] for the $PdH_x$ (△), $PdD_x$ (◯) and $PdT_x$ (□) alloy systems. . . . . 72

5.3 EAM function curves for (a) embedding energy; (b) electron density; and (c) pair energy. . . . . 75

5.4 Energy as a function of lattice spacing for OC lattices. . . . . 76

5.5 Lattice spacing as a function of composition for OC lattices. . . . . 77

5.6 Cohesive energy as a function of composition for OC and TE lattices. . 78

5.7 Gibbs free energy of mixing as a function of mole fraction for the OC structure. . . . . 79

5.8 Elastic constants as a function of composition for OC structure. (a) B; (b) $C'$; and (c) $C_{44}$. Curved, solid lines are shown as guides for connecting the simulated compositions. . . . . 81

5.9 Tensile stress as a function of stretch. . . . . 82

5.10 Tensile strength as a function of composition for OC structures. . . . . 82

5.11 Shear stress as a function of applied strain. . . . . 83

5.12 Shear strength as a function of composition for OC structures. . . . . 84

5.13 Change of atomic configurations of a $PdH_{0.784}$ phase due to MD annealing at 300 K for one nanosecond followed by MS energy minimization. (a) The TE structure before MD+MS relaxation; and (b) After MD+MS energy relaxation. . . . . 85

5.14 Snapshots of single hydrogen atom diffusion in a palladium matrix. (a) time 0.000 ps, (b) time 0.043 ps; and (c) time 0.063 ps. The size of the hydrogen atom is magnified for clarity. . . . . 86

# List of Tables


3.1 Cohesive Energies for lattices of the $PdH_x$ binary system calculated using VASP. . . . . 31

3.2 Elastic constants for lattices of the $PdH_x$ binary system calculated using ParaDyn and the fitted potential. . . . . 36

3.3 Tensile strengths for lattices of the $PdH_x$ binary system calculated using ParaDyn and the fitted potential. . . . . 38

3.4 Tensile strengths for PdH for different crystal directions calculated using ParaDyn and the fitted potential. . . . . 38

4.1 Parameter values for the second iteration of EAM of the Pd-H system. . 50

4.2 Objective Function values for various optimization methods used in the EAM fitting process. . . . . 53

4.3 Parameter values for the an EAM potential of the Pd-H system. . . . . 56

4.4 Parameter values for an EAM potential of the Pd-H system. . . . . 58

5.1 Interpolated data from Figure 5 of [27] for $C'$ as a function of H-isotope concentration. . . . . 70

5.2 Interpolated data from Figure 6 of [27] for $C_{44}$ as a function of H-isotope concentration. . . . . 71

5.3 Adjusted DFT properties. . . . . 72

5.4 EAM parameters. . . . . 73

5.5 Predicted values of lattice constant a (Å), cohesive energy $E_c$ (eV/atom), and elastic constants $C_{11}$, $C_{12}$, and $C_{44}$ (GPa). . . . . 76

This page intentionally left blank.

# Chapter 1

# Introduction

Ongoing research at Sandia National Laboratories has been in the area of developing models and simulation methods that can be used to uncover and illuminate the material defects created during He bubble growth in aging bulk metal tritides. Helium (He) is a by-product of the decay of tritium, and the insolubility of He with metals allows He atoms to be retained within the material leading to the formation of nano-scale bubbles that grow as the concentration of He within the metal increases. The development of these bubbles has been observed experimentally within palladium (Pd) tritide alloys [1, 2, 3], and analytical [4, 5, 6, 7, 8] and computational [9, 10, 11, 12, 13] models have been developed to study the processes by which these bubbles form, grow and deform the surrounding metal.

Two of the efforts listed above have used molecular dynamics calculations to examine the physical mechanisms by which growing He bubbles in a Pd metal lattice create material defects. Foiles and Hoyt [10] performed simulations that displayed the formation of dislocation threads that interconnected the bubbles, as well as other material defects such as vacancies and stacking fault tetrahedra. Their simulations utilized full periodic boundary conditions in order to represent a region of bulk crystal. Follow-on work to this was done by Zimmerman [11], who compared simulations of a bulk system with a model system containing bubbles near a free surface. His results clearly show the influence of the free surface on defect creation and evolution through examining the formation of inter-bubble dislocations as well as bubble pressure and swelling characteristics as functions of He to metal concentration. Also, Zimmerman's results revealed the need for improving the implementation of non-periodic boundary conditions in these simulations.

While the results by Foiles, Hoyt and Zimmerman are useful, they are focused only on the growth of He bubbles in pure Pd and not on bubble growth in the material of interest, palladium tritide (PdT), or its non-radioactive isotope palladium hydride (PdH). The reason for this is that existing inter-atomic potentials do not adequately describe the thermodynamics of the Pd-H system, which includes a miscibility gap that leads to phase separation of the dilute (alpha) and concentrated (beta) alloys of H in Pd at temperatures below a critical temperature of approximately 565K (at a pressure of 20 bar) [14]. This feature is apparent in the phase diagram of the Pd-H system, shown in Figure 1.1. At room temperature and ambient pressure, the

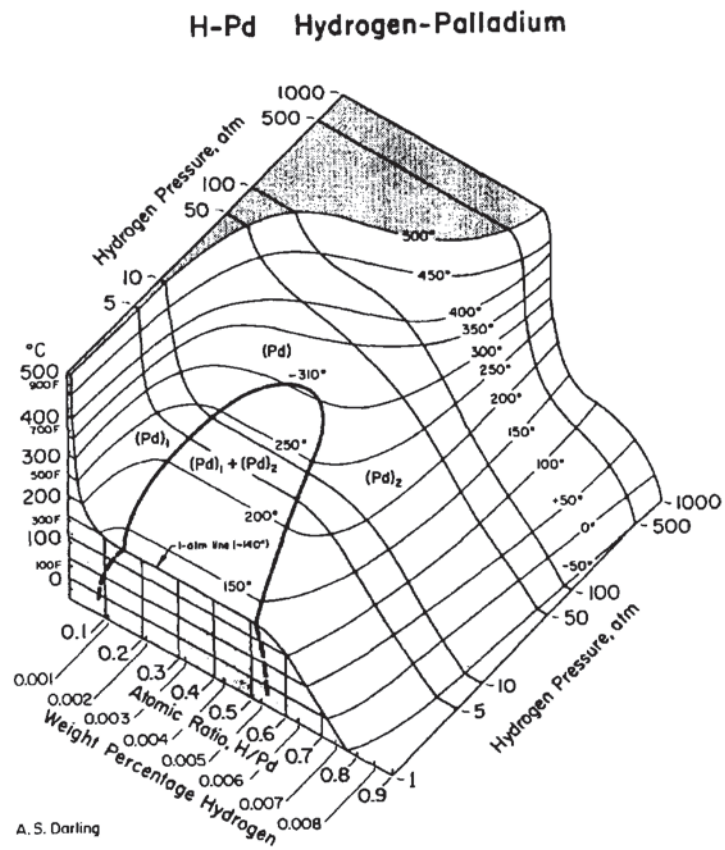



**Figure 1.1.** The phase diagram for the Pd-H system.

alpha and beta phases exist at approximate concentrations of H/Pd = 0.03 and 0.60, respectively [15].

The goal of this research is to either find or develop inter-atomic potentials for the Pd-H and Pd-T systems, and to use such potentials to study He bubble growth in the alloy material. As will be discussed in more detail in Chapter 2, while a number of inter-atomic potentials have been designed for the Pd-H system, only one was found to capture the physical characteristic of a miscibility gap [16] and the publications that discuss this potential do not contain sufficient detail to implement the potential in a working molecular dynamics simulation. As such, Chapters 3, 4 and 5 describe our group's efforts to use experimental data and density functional theory calculations to create an inter-atomic potential for this unique metal alloy system. Chapter 3 describes our initial attempts to construct a pair potential for the Pd-H interaction while using an Embedded Atom Method (EAM) formalism [17, 18] for modeling the Pd itself. Chapter 4 discusses our use of numerical optimization methods to fit a

combined EAM potential for the Pd-H system while Chapter 5 presents our most recent efforts that use analytical expressions and physical reasoning combined with the numerical optimization methods implemented within the software Mathematica [19].

This page intentionally left blank.

# Chapter 2

# Previous Models for the Pd-H System

As mentioned in Chapter 1, the goal of our research was to either find or develop interatomic potentials for the Pd-H system for use in molecular dynamics simulations of He bubble growth. Examination of the available scientific literature revealed several potentials, which will be discussed in this chapter. In particular, the potentials will be evaluated with regard to whether they produce a miscibility gap and other physical characteristics of the Pd-H system.

## 2.1 Embeded Atom Method by Daw and Baskes

The EAM potential was created by Daw and Baskes [20, 17] to be a multi-body interatomic potential capable of simulating face-centered-cubic (FCC) metals and their impurities. For a system of such a material, the total potential energy is

$$E = \frac{1}{2}\sum_{\alpha=1}^{N}\sum_{\beta\neq\alpha}^{N}\phi_{\alpha\beta}(r^{\alpha\beta}) + \sum_{\alpha=1}^{N}F_{\alpha}(\rho_{\alpha}), \tag{2.1}$$

where there are $N$ atoms within the system. In this expression, each atom's potential energy is derived from two separate contributions. The first is the summation of pair potential energy between a given atom $\alpha$ and its neighbors $\beta$, $\phi_{\alpha\beta}$. These pair potential energies depend only on the radial separation of the two atoms, $r^{\alpha\beta}$ , and are representative of the repulsion between the ionic cores of atoms. The second contribution, $F_{\alpha}$, is the energy necessary to "embed" atom $\alpha$ in an electron gas of some density $\rho_{\alpha}$ composed of contributions from all the neighbors $\beta$ of the atom. The electron density is assumed to be a linear superposition of spherically-averaged atomic charge densities,

$$\rho_{\alpha} = \sum_{\beta\neq\alpha}^{N} f_{\beta}\left(r^{\alpha\beta}\right), \tag{2.2}$$

where $f_{\beta}$ is an atomic charge density function. For molecular dynamics (MD) simulations, it is essential to calculate the force on each atom, which can be determined

by taking the derivative of $E$ with respect to its position vector. Thus,

$$\boldsymbol{F}^{\alpha} = -\sum_{\beta\neq\alpha}^{N} \left[ \left( F'_{\alpha}(\rho_{\alpha})\, f'_{\beta}\left(r^{\alpha\beta}\right) + F'_{\beta}(\rho_{\beta})\, f'_{\alpha}\left(r^{\alpha\beta}\right) \right) + \phi'_{\alpha\beta}\left(r^{\alpha\beta}\right) \right] \frac{\boldsymbol{r}^{\alpha\beta}}{r^{\alpha\beta}} \tag{2.3}$$

In their early work, Daw and Baskes use a screened coulomb-like term for the pair function $\phi_{\alpha\beta}$,

$$\phi_{\alpha\beta}(r) = \frac{Z_{\alpha}(r) Z_{\beta}(r)}{r}, \tag{2.4}$$

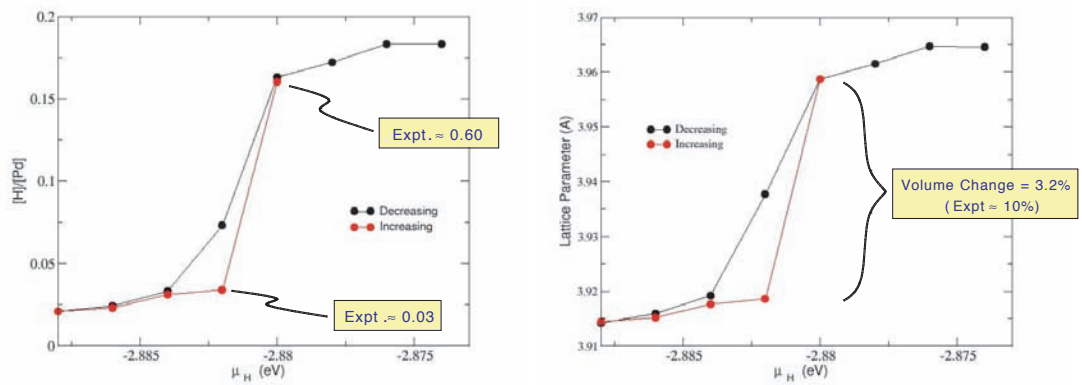
and the atomic charge density functions $f_{\beta}(r)$ are determined from the $s$- and $d$-like densities computed by Clementi and Roetti [21]. The embedding functions $F^{\alpha}(\rho)$ are determined using specific qualitative forms and quantitatively fitting them to the experimentally-observed cohesive behavior of the pure metal. Additional details can be found in [17], which provides parameters and functions for both the Ni-H and Pd-H systems.

In [17], Daw and Baskes analyze the interaction of H with Pd. For dilute amounts of H, they accurately reproduce physical characteristics such as migration and trapping energies of H atoms in a bulk Pd lattice. They also note that they find a reasonable lattice expansion (4%) for the PdH alloy as compared with the value of 3.5% observed for the beta-phase concentration of H. However, in this paper they did not explore whether a miscibility gap exists for the potential they created, nor did they characterize mechanical properties (*e.g.* elastic moduli, tensile strength, shear strength) for comparison with experiment.

Our own simulations reveal that this potential displays only a very small miscibility gap with boundaries at H/Pd ratios of 0.03 and 0.17, as shown in Figure 2.1. Simulation results also incorrectly predict that, at high H concentrations, interstitial hydrogen atoms preferentially lie at tetrahedral sites, not octahedral sites as observed in experiment. In Chapter 3, we will discuss our efforts to modify this potential to improve these characteristics.

## 2.2 Multi-body potentials by Puska *et al.* and Zhong *et al.*

At approximately the same time that Daw and Baskes developed EAM, Puska and Nieminen published a similar multi-body inter-atomic potential later to be called Effective Medium Theory (EMT) [22, 23, 24]. EMT is similar in form to EAM, but uses functions and parameters that are wholely derived or calculated using quantum mechanics. EMT does not fit any parameters to empirically-determined information such as lattice constants and elastic moduli measured experimentally. In [23], Puska and Nieminen do a complete evaluation of self-trapped hydrogen in both FCC and




(a) (b)

**Figure 2.1.** Properties of the Pd-H system for the Daw-Baskes EAM potential at T=300K (a) Concentration vs Chemical Potential (b) Lattice Parameter vs Chemical Potential

body-centered-cubic (BCC) metals. However, they too do not explore the existence of a miscibility gap for the Pd-H system, nor do they consider large concentrations of H in bulk Pd.

Sometime later, another multi-body potential was developed by Zhong, Li and Tománek [25, 26]. The potential by Zhong *et al.* is also similar to the EAM potential, but has some significant differences. Specific forms for $F^\alpha(\rho)$, $f_\beta(r)$ and $\phi_{\alpha\beta}(r)$ are used:

$$F^\alpha(\rho) = -\sqrt{\rho} \tag{2.5}$$

$$f_\beta(r) \rightarrow f_{\alpha\beta}(r) = \xi^2_{0,\alpha\beta} \exp\left[-2q_{\alpha\beta}\left(\frac{r^{\alpha\beta}}{r_0^{\alpha\beta}} - 1\right)\right] \tag{2.6}$$

$$\phi_{\alpha\beta}(r) = \epsilon^R_{0,\alpha\beta} \exp\left[-p_{\alpha\beta}\left(\frac{r^{\alpha\beta}}{r_0^{\alpha\beta}} - 1\right)\right] \tag{2.7}$$

In these expressions,

$$\xi_{0,\alpha\beta} = \frac{E_{\text{coh}}(\text{bulk})}{(1 - q/p)\sqrt{Z_{\text{bulk}}}} \tag{2.8}$$

and

$$\epsilon^R_{0,\alpha\beta} = \frac{\xi_{0,\alpha\beta}}{\sqrt{Z_{\text{bulk}}}}\frac{q}{p}. \tag{2.9}$$

These expressions are distinctly different from the usual EAM format in that the atomic charge density function contains terms that take into account the specific

environment for a given atom. Thus, the electron density at that atom's position depends not only on the arrangement and types of atoms neighboring that atom, but on what type of atom that particular atom is. Also, this approach does not assume local charge neutrality in alloys. Finally, the formalism for this potential considers only nearest-neighbor interactions, both to evaluate the expressions in equations (2.8) and (2.9) and for any given configuration. While the EAM format typically does utilize some set radial cutoff for interactions, the specific distance is not prescribed and interactions usually range from third- to fifth-nearest neighbors, if not beyond. The longer-range of EAM enables it to more accurately reproduce surface and stacking fault energies and free surface relaxations than most nearest-neighbor potentials with only moderate increases in computational cost.

The articles authored by Zhong *et al.* [25, 26] discuss a number of features of their potential. Analysis of equilibrium structure and binding energy for pure Pd and PdH, as well as H-free and H-covered Pd(001) and Pd(110) surfaces shows good agreement with available experimental data. In addition, they note that bulk modulus decreases as a function of H concentration for the $PdH_x$ alloy system, although they also state that Young's modulus values are unchanged by the presence of H, a feature that may not consistent with experiment [27]. They also mention a reduction of mechanical stability due to H that originates from *increased* ductility and plasticity in regions of high H concentration. It is not clear that this explanation is consistent with the known phenomenon of "hydrogen embrittlement", and the authors do note the necessity of further experimental confirmation.

In their articles, Zhong *et al.* do not mention the miscibility gap for the Pd-H system, nor do they conduct simulations to explore whether their potential produces one. Our simulation work reveals that it indeed does not produce such a gap. Figure 2.2 shows Monte Carlo results of H concentration vs. the applied chemical potential of Hydrogen at a temperature of 300K, where the solid line is a guide to eye. Clearly there is no indication of a discontuous jump in chemical potential, indicative of a miscibility gap, predicted by the Zhong et al potential. In addition to the incorrect phase equilibrium behavior, energy minimization of the PdH alloy shows that the potential also incorrectly predicts, at high concentrations, H moving to tetrahedral interstitial sites when initially placed at octahedral sites, as seen in Figure 2.3.

## 2.3 EAM model by Wolf *et al.*

Since the potentials created by Daw and Baskes, Puska *et al.* and Zhong *et al.* fail to reproduce the miscibility gap for the Pd-H system, one may be tempted to question whether an EAM-like formalism is even capable of exhibiting this behavior. However, the work by Wolf, Lee and coworkers at the Savannah River National Laboratory[1] and

[1] At the time of their publications, this facility was known as the Westinghouse Savannah River Company and/or the Savannah River Technology Center.

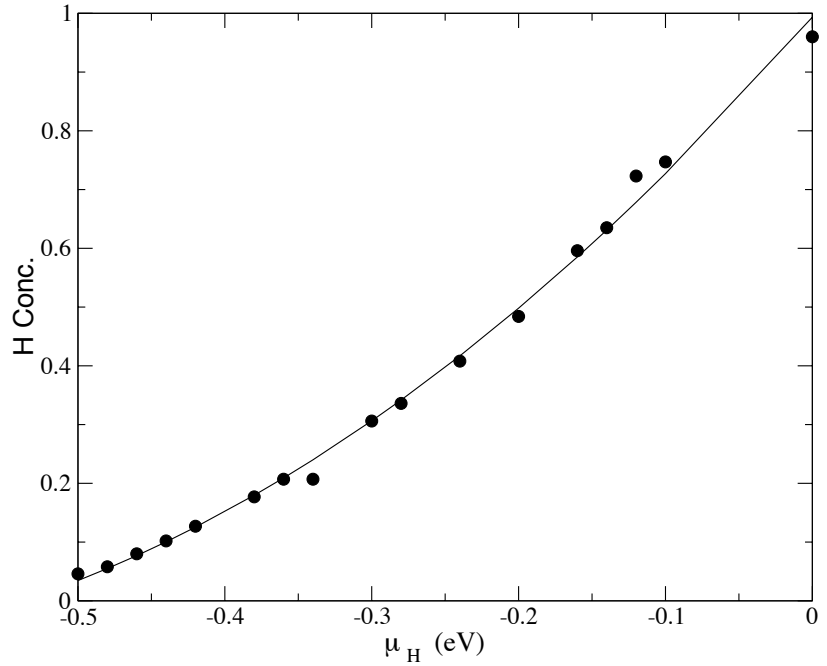
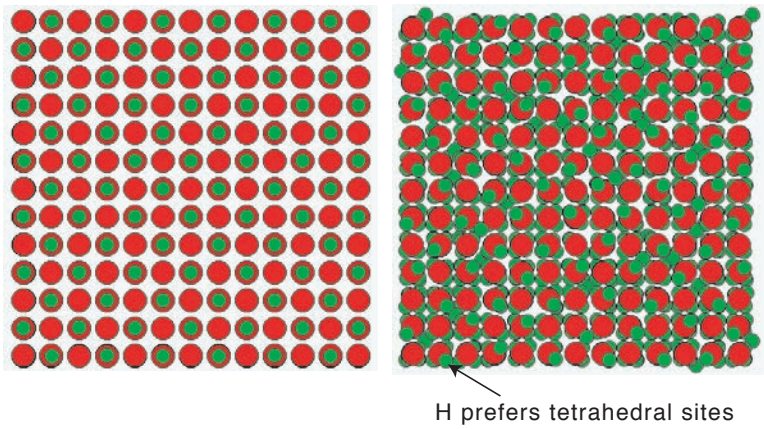



**Figure 2.2.** Monte Carlo results of hydrogen concentration vs. chemical potential for the Zhong *et al.* potential at 300K. The solid line is a guide to the eye and there is no indication of a miscibility gap.

their collaborators at Clemson University shows that this possibility exists. In their papers [28, 16, 29, 30], these authors show the ability of their potential to reproduce the miscibility gap through the use of pressure-composition isotherms. For the $PdH_x$ alloy system, their potential shows behavior that qualitatively matches experimental results, as shown in Figure 2.4. This figure not only shows the constant pressure plateau characteristic of a miscibility gap, but shows several such plateaus for different isotherms with behavior consistent with a critical point of 565K, the experimentally measured value. Reference [16] also presents room temperature (T = 300K) isotherms for the palladium-tritide system, seen in Figure 2.5. Both $PdH_x$ and $PdT_x$ systems show similar alpha and beta phase boundaries, at values of H/Pd approximately equal to 0.14 and 0.84, respectively. The authors do note that these values are larger than what is experimentally observed, and promise future refinements in the potential related to volume expansion as a function of H/Pd to improve the accuracy of these properties.

Unfortunately, this potential could not be implemented within our molecular dynamics simulation code because the functions and parameters that dictate the Pd-H and




(a) (b)

**Figure 2.3.** Energy Minimization of PdH using the potential by Zhong *et al.* (a) Atomic configuration before minimization (b) Atomic configuration after minimization

H-H interactions do not appear in the publications by Wolf *et al.* In their first publication [28], the authors do present functions and parameters to model pure Pd. The pair potential $\phi$ is assumed to be a Morse potential,

$$\phi(r) = D\left(\exp\left[-2\alpha\left(r - r_0\right)\right] - 2\exp\left[-\alpha\left(r - r_0\right)\right]\right), \tag{2.10}$$

and the atomic charge density function $f$ is

$$f(r) = r^6\left(\exp\left[-\beta r\right] - 512\exp\left[-2\beta r\right]\right). \tag{2.11}$$

The embedding function $F(\rho)$ is found by equating the total system potential energy as a function of lattice parameter, for a hydrostatically deformed configuration, to the equation of state developed by Rose *et al.* [31],

$$E(a) = -E_{\text{coh}}\left(1 + a^*\right)\exp\left[-a^*\right] \tag{2.12}$$

where

$$a^* = \left(\frac{a}{a_0} - 1\right)\sqrt{\frac{9B\Omega}{E_{\text{coh}}}}, \tag{2.13}$$

$a_0$ is the equilibrium lattice constant, $B$ is the zero temperature bulk modulus, and $\Omega$ is the equilibrium atomic volume ($= a_0^3/4$). Additional details and values for parameters appear in [28]. The authors proceed to compare both third-and fifth-nearest neighbor versions of their potential with experimental properties such as thermal expansion, specific heat, melting temperature and elastic moduli. The authors conclude that the fidelity of the third-nearest neighbor potential is sufficient for most purposes.

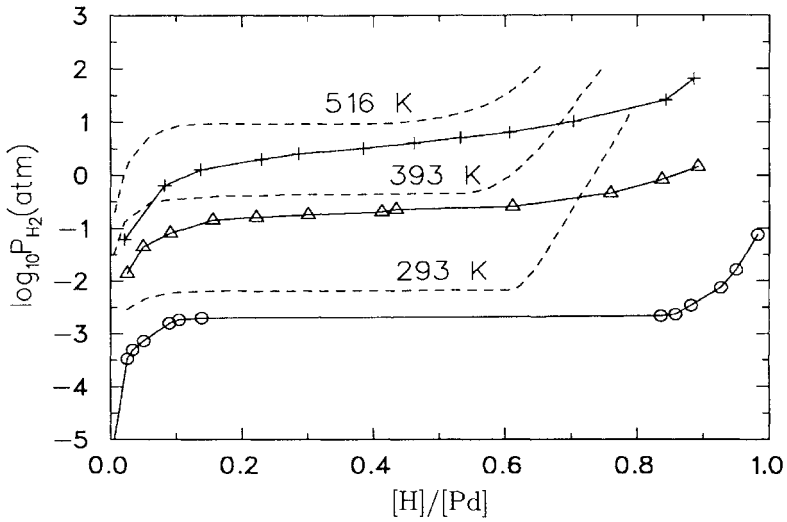



**Figure 2.4.** Pressure-composition isotherms for the $PdH_x$ alloy system produced with the Wolf *et al.* potential. (taken from [16])

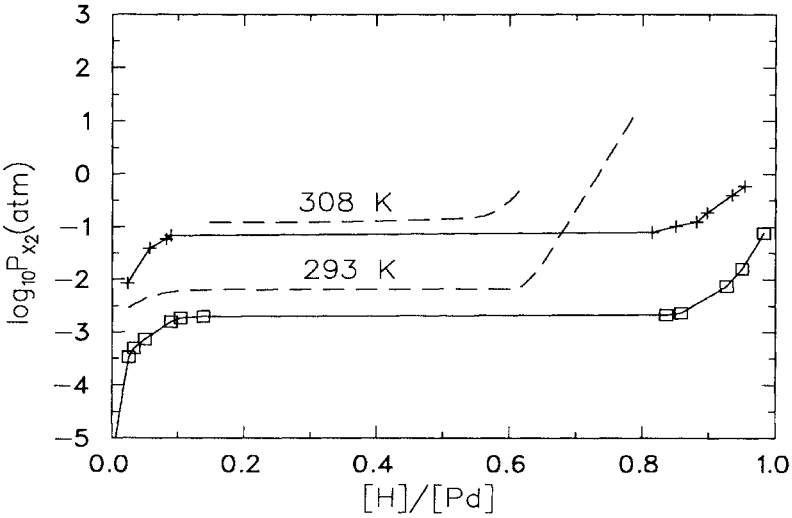



**Figure 2.5.** Pressure-composition isotherms at 300K for the $PdH_x$ (□) and $PdT_x$ (+) alloy systems produced with the Wolf *et al.* potential. (taken from [16])

In their next article [16], the authors revisit their third-nearest neighbor potential and expand it for use on the Pd-H system. However, in this article the authors do not list any additional functional forms that may have been used, or, assuming the same functional forms are used as in [28], any parameter values for the Pd-H and H-H interactions. The authors only note that additional pair and embedding functions are needed for these interactions, discuss the methods for performing their simulation and potential development work, and present the pressure-composition isotherms already discussed. Later papers by these authors [29, 30] also do not list any functions or parameters, and only cite the earlier articles for any references to the Pd-H potential.

Additional efforts were made by our research group to obtain the necessary information to replicate the Wolf *et al.* potential. Phone contact was made with Ralph Wolf, John Ray, Khalid Mansour, and Patrick Fay, but none of these individuals had retained this information. Unfortunately, the sole individual that may have possessed such information, Myung Lee, had passed away sometime during the late 1990's. The lead author of this report traveled to SRNL in August of 2006 to go through all remaining documentation created by Wolf, Lee and Mansour during their employment at SRNL. No quantitative information was found on the parameters or functions that could be definitively connected to the work presented in [16]. Some notes were recovered that showed the use of an embedding function

$$F(\rho) = b_1\rho + b_2 + \frac{1}{b_3\rho + b_4}, \tag{2.14}$$

and the values for H are listed as $b_1$ = 20.44510, $b_2$ = -2.897859, $b_3$ = 52.89785 and $b_4$ = 0.412562 where $F$ is in units of eV. However, other notes showed this functional form was only used for H while the forms

$$F(\rho) = a_1\frac{\rho}{\rho_0} + a_2\left(\frac{\rho}{\rho_0}\right)^2 + a_3\left(\frac{\rho}{\rho_0}\right)^3 + a_4\left(\frac{\rho}{\rho_0}\right)^4 \tag{2.15}$$

and

$$F(\rho) = c_1\rho \tag{2.16}$$

were used for Pd and He, respectively, where $a_1$ = -8.47412, $a_2$ = 5.55418, $a_3$ = -2.51868, $a_4$ = 0.741624, $\rho_0$ = 0.01518 and $c_1$ = 42. In addition, these secondary notes

list the following functions for Pd, H and He:

$$Z_{\mathrm{Pd}}(r) = 10\left(1+0.0595r\right)\exp\left[-1.295r\right] \tag{2.17}$$

$$Z_{\mathrm{H}}(r) = \begin{cases} \left(1-\frac{r}{2}\right)^{3.83} & r<2, \\ 0 & r\geq 2. \end{cases} \tag{2.18}$$

$$Z_{\mathrm{He}}(r) = \begin{cases} 1.6875\left(1-\frac{r}{2}\right)^{3.83} & r<2, \\ 0 & r\geq 2. \end{cases} \tag{2.19}$$

$$\rho^{a}_{\mathrm{Pd}}(r) = 0.8478\rho^{as}_{\mathrm{Pd}}(r) + (10-0.8478)\rho^{ad}_{\mathrm{Pd}}(r) \tag{2.20}$$

$$\rho^{as}_{\mathrm{H}}(r) = \frac{1}{\pi 0.5292^3}\exp\left[-\frac{2r}{0.5292}\right] \tag{2.21}$$

$$\rho^{as}_{\mathrm{He}}(r) = \frac{1}{4\pi}\left\{1.79478\exp\left[-2.91093r\right] + 2.95761\exp\left[-1.45363r\right]\right\}^2 \tag{2.22}$$

It is not apparent whether these functions are related to the work in [16]. The expressions for atomic charge density functions seem to indicate the use of Clementi and Roetti's tables [21], as was done by Daw and Baskes, for Pd. However, this does not match what was done in [28]. There is some indication that these functions were used in earlier efforts by Mansour and Wolf, as they appear in internal technical reports [32, 33]. However, functions and parameters for H and He do not appear in these reports. The listing of functions $Z(r)$ indicate the pair potential form used by Daw and Baskes, shown in equation (2.4). However, both reports document the use of the Morse pair potential, and, while parameters are listed for Pd, none are listed for either H or He. In any event, no evidence was found that connects either the discovered notes or the technical reports to the published work by Wolf *et al.*

Some presentation material was discovered that showed Wolf and Lee were able to improve the potential that appears in [16] as the beta phase concentration was reduced to a value of H/Pd approximately equal to 0.6. However, this material did not contain any details related to the improved potential itself, only the resulting pressure-composition isotherm.

Finally, contact was made with Thomas Graham, a student of John Ray's at Clemson University who used the Wolf *et al.* potential in his doctoral dissertation [34]. However, this dissertation does not contain quantitative information on the Wolf *et al.* potential, only citations to [28] and [16]. Phone and e-mail contact was made with Dr. Graham, but he had not retained this information.

With no other avenues for further investigation, our group proceeded to develop a new potential capable of accurately modeling the physical behavior of the Pd-H system. Our efforts will be detailed in the chapters that follow.

This page intentionally left blank.

# Chapter 3

# Modifications to Daw and Baskes EAM

In light of the inability of the potentials discussed in Chapter 2 to reproduce the physical trait of a miscibility gap for the Pd-H system at room temperature, our first efforts were directed towards adapting the Daw and Baskes EAM potential to incorporate this behavior. In this chapter, we discuss our approach in performing this adaptation, and the resulting physical and mechanical behavior possessed by the potential we created.

## 3.1 Approach

We began by using the density functional theory [35] simulation code VASP [36] to analyze atomic systems of specific stochiometries for the $PdH_x$ alloy system, *i.e.* specific values of $x$.[1] The values chosen were $x = 0.25$, 0.5, 0.75 and 1, and the unit cells for these systems are shown in Figure 3.1. These systems were composed of a

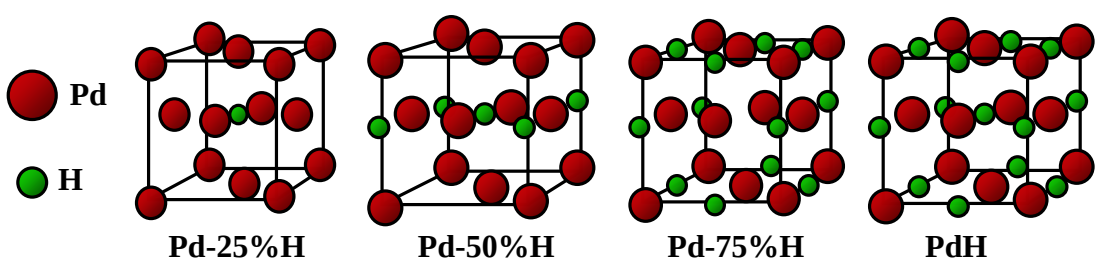



**Figure 3.1.** Unit cells for the stochiometries of $x = 0.25$, 0.5, 0.75 and 1 of the $PdH_x$ binary system.

total of 27 unit cells, 3 unit cells for each $< 100 >$ direction, and contain between 135

[1]These calculations were performed for the report authors by Thomas K. Mattsson, a technical staff member of the HEDP Theory department at Sandia National Laboratories, Albuquerque, NM.

and 216 atoms in total, depending on the value of $x$. VASP was used to calculate the energy per unit cell for values of lattice parameter ranging from 3.9 Å to 4.25 Å. VASP was also used to calculate energy values for a pure Pd system. All of these values can be found in Table 3.1. It is immediately noticeable from the Pd calculation that DFT overestimates the magnitude of energy per unit cell as compared with experimental estimates. For example, at the measured equilibrium lattice parameter of 3.89 Å, the cohesive energy of Pd is known to be -3.91 eV/atom [18], or -15.64 eV/cell. However, interpolation of the values listed in Table 3.1 yields an estimate of -20.814326 eV/cell at a somewhat larger lattice parameter of approximately 3.9596 Å. This discrepancy requires that an energy offset be applied to the VASP data so that the use of, and agreement with, existing EAM models for Pd is possible. A more thorough discussion of this offset will be presented in Chapters 4 and 5.

Once we established a database of cohesive energy values to fit to, we chose specific functions to model the Pd-H and H-H interactions. As a first approximation, we set the atomic charge density function for hydrogen, $f_{\mathrm{H}}$, equal to zero. This eliminated the need to determine the hydrogen embedding function, $F_{\mathrm{H}}$, and necessitates only fitting the pair interactions $\phi_{\text{Pd-H}}$ and $\phi_{\text{H-H}}$. A Morse pair potential of the form $\phi(r) = A\exp\left[-2\alpha\left(r-r_0\right)\right] + B\exp\left[-\alpha\left(r-r_0\right)\right]$ was first used to fit the Pd-H interaction, while the H-H interaction was the same pair function used by Daw and Baskes [17]. Fitted values determined were $\alpha = 3.4$ Å$^{-1}$, $A =$ -0.255 eV, $B =$ -2$A$ and $r_0 = 2.0$ Å. However, it was soon discovered that the FCC phase for PdH was not stable unless a minimum is added to $\phi_{\text{Pd-H}}$ at the second nearest neighbor distance, $r = 4.5$ Å. This was done using a Gaussian function centered at this value of neighbor distance. The resulting pair function is shown in Figure 3.2.

## 3.2 Physical properties of the modified potential

This modified potential produces some interesting properties that do indicate partial success. First, it yields and equilibrium lattice parameter of 4.1322 Å for the PdH alloy, which is in agreement with the experimental measured value of 4.12 Å for the beta phase concentration. We also notice that the potential gives a very close prediction of the dependency of lattice parameter on H content, as seen in Figure 3.3 In this Figure, the predicted values of lattice parameter for the 25% and 75% concentrations lie just slightly above a straight line that connects the 0% and 100% concentrations (The 50% structure, as shown in Figure 3.1, possesses tetrahedral symmetry. Therefore, the idealized crystal for the 50% concentration has two lattice parameters and no attempt was made to compare with experiment.). It is interesting to note that the experimental value of lattice parameter for the beta phase concentration lies on the straight line itself.

Another comparison of the fitted potential with the VASP database is shown in Figure 3.4, which plots the unit cell energy (relative to its minimum) as a function of

**Table 3.1.** Cohesive Energies for lattices of the $PdH_x$ binary system calculated using VASP.

| $x$ = H/Pd | Number of Atoms per unit cell | Lattice Parameter (Å) | Energy per unit cell (eV) |
|---|---|---|---|
| 0 | 4 | 3.66 | -18.421469 |
| | | 3.76 | -19.860436 |
| | | 3.86 | -20.593786 |
| | | 3.96 | -20.808887 |
| | | 4.06 | -20.653975 |
| | | 4.16 | -20.239540 |
| 0.25 | 5 | 3.90 | -24.031153 |
| | | 3.95 | -24.235094 |
| | | 4.00 | -24.324118 |
| | | 4.05 | -24.315371 |
| | | 4.10 | -24.223130 |
| | | 4.15 | -24.060911 |
| 0.5 | 6 | 4.00 | -27.647505 |
| | | 4.05 | -27.928507 |
| | | 4.10 | -27.911924 |
| | | 4.15 | -27.810875 |
| | | 4.20 | -27.638371 |
| 0.75 | 7 | 4.00 | -31.224208 |
| | | 4.05 | -31.411258 |
| | | 4.10 | -31.482005 |
| | | 4.15 | -31.452637 |
| | | 4.20 | -31.337810 |
| | | 4.25 | -31.149215 |
| 1 | 8 | 4.00 | -34.419099 |
| | | 4.05 | -34.716911 |
| | | 4.10 | -34.881462 |
| | | 4.15 | -34.930072 |
| | | 4.20 | -34.879271 |
| | | 4.25 | -34.529706 |

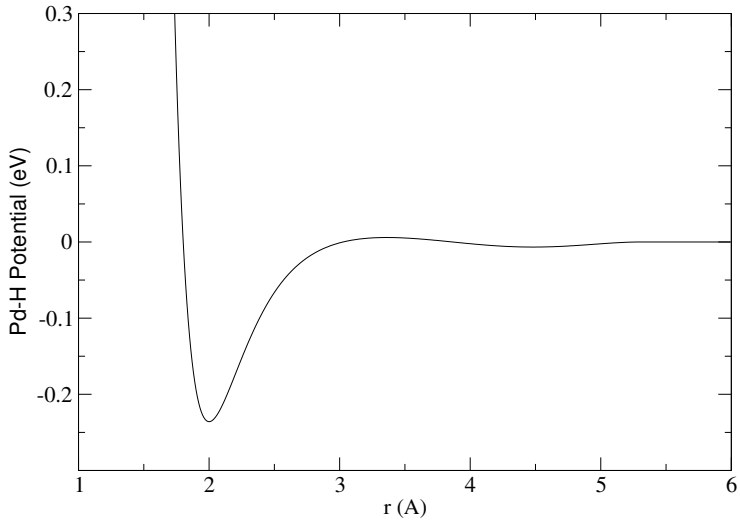



**Figure 3.2.** Pair function for the Pd-H interaction

lattice parameter for PdH. While these curves display a small offset, it can be seen that their curvatures are nearly identical. The VASP values for PdH give a curvature of 1.28 times the curvature of pure Pd, while the fitted potential produces a value of 1.27. This agreement suggests that predictions of bulk modulus for PdH would be accurate since curvature of the lattice energy curve is related to bulk modulus.

Regarding the prediction of a miscibility gap, Figure 3.5 shows that one does indeed exist for this fitted potential. However, the miscibility gap boundaries for the fitted potential are essentially at 0 and 100% H. Thus, all intermediate compositions, including the beta phase concentration, would tend to phase separate and thus be unstable. Some refinements to this potential were done in an attempt to correct this phase separation behavior, but this only had the effect of destabilizing the FCC structure for the PdH alloy combination. A test configuration at a temperature of 300 K can be seen in Figure 3.6.

## 3.3 Mechanical properties of the modified potential

As mentioned in the previous section, the agreement between the fitted potential and the VASP database regarding the curvature of the energy versus lattice parameter

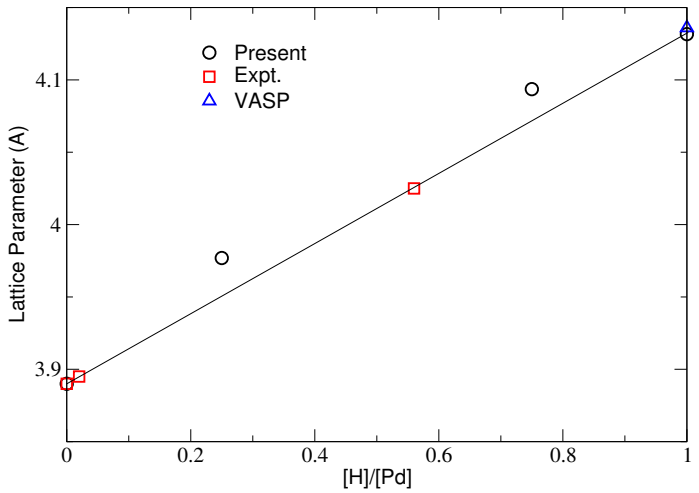



**Figure 3.3.** Lattice parameter for the $PdH_x$ system.

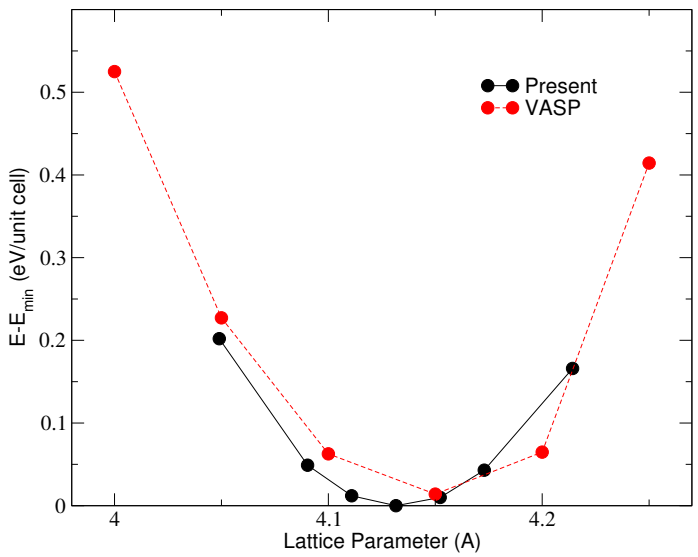



**Figure 3.4.** Cohesive energy per unit cell for PdH. The black curve shows the results for the fitted potential while the red curve shows the VASP values.

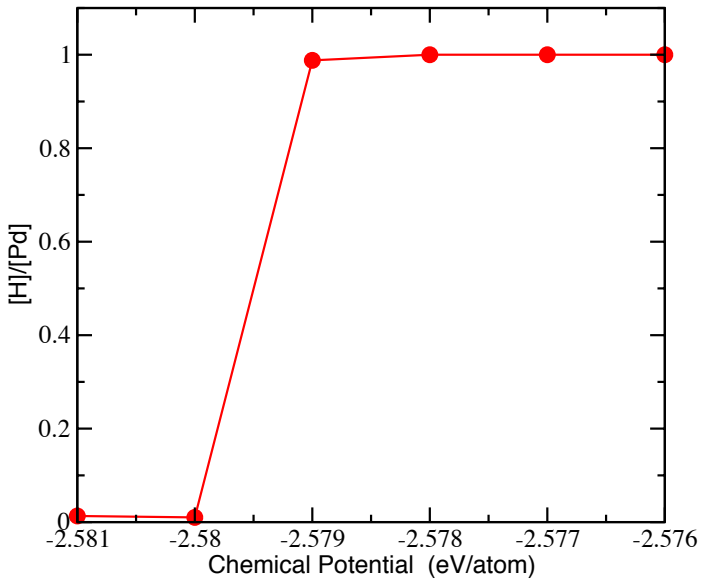



**Figure 3.5.** Hydrogen concentration versus chemical potential for the fitted Pd-H potential, clearly showing a miscibility gap.

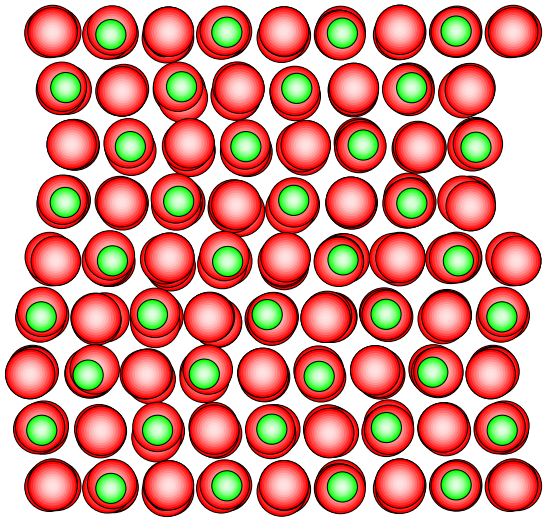


**Figure 3.6.** Atomic configuration of PdH after 100 ps at T = 300 K.

curve for PdH suggested that the potential would do a decent job at representing the bulk modulus of this material. We also used molecular statics calculations to evaluate other elastic constants, *i.e.* the values of elastic moduli at zero strain, and tensile strengths and how both vary with the concentration of hydrogen.

Figure 3.7 shows the four configurations for which mechanical properties were evaluated. These systems are 10 unit cells on a side and contain 5000, 6000, 7000, and

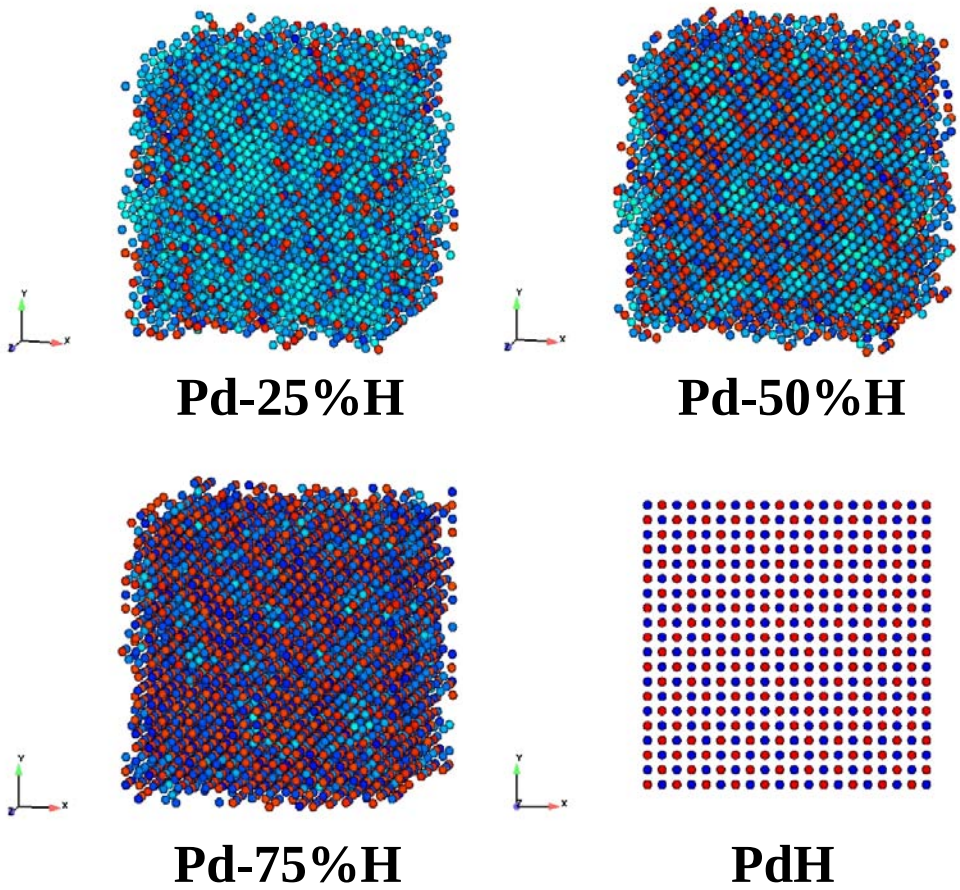



**Figure 3.7.** Atomic configurations for the stochiometries of $x = 0.25$, 0.5, 0.75 and 1 of the $PdH_x$ binary system. The axes shown for each configuration denotes the $< 100 >$ crystal directions.

8000 atoms, respectively, for the 25%, 50%, 75% and 100% H concentrations. Simulations were performed using the molecular dynamics simulation code ParaDyn [37] and periodic boundary conditions were used on all faces.

Values for the elastic constants $C_{11}$ and $C_{12}$ are shown in Table 3.2. Also shown in this table are the estimated values for bulk modulus, $B = (C_{11} + 2C_{12})/3$, and the

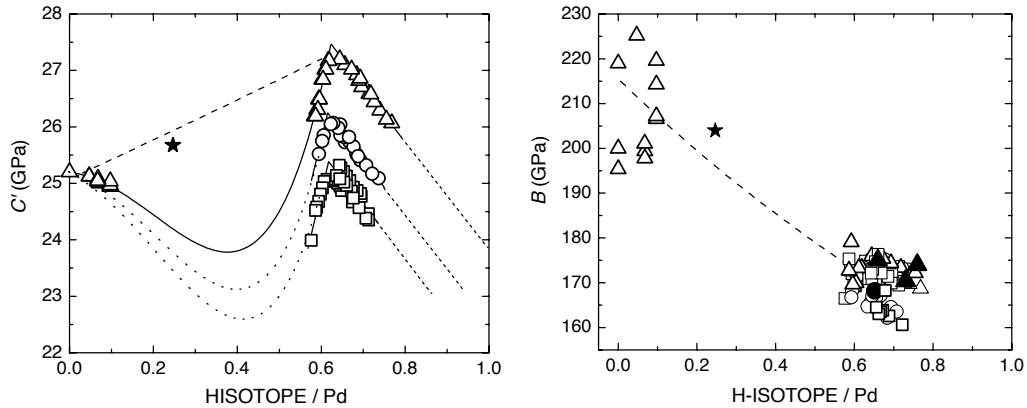
**Table 3.2.** Elastic constants for lattices of the $PdH_x$ binary system calculated using ParaDyn and the fitted potential.

| $x$ = H/Pd | $C_{11}$ (GPa) | $C_{12}$ (GPa) | $B$ (GPa) | $C'$ (GPa) |
|---|---|---|---|---|
| 0 | 239.47 | 173.67 | 195.60 | 32.9 |
| 0.25 | 267.32 | 156.20 | 193.24 | 55.56 |
| 0.5 | 308.62 | 143.35 | 198.44 | 82.64 |
| 0.75 | 367.79 | 133.12 | 211.34 | 117.34 |
| 1 | 452.03 | 126.59 | 235.07 | 162.72 |

alternative shear modulus, $C' = (C_{11} - C_{12})/2$.

Table 3.2 shows that as H concentration increase, both $C_{11}$ and $C'$ increase monotonically while $C_{12}$ decreases monotonically. Bulk modulus is observed to decrease slightly at a concentration of 25%, but then increases for the larger concentrations. In his Sandia technical report, Wolfer [38] experimentally observed the trend of decreasing $C_{12}$ with increasing tritium content, in agreement with our simulation results. However, Wolfer also notes this behavior for $C_{11}$, which is opposite to the trend seen in our simulation results. Schwarz *et al.* performed more recent measurements for $PdH_x$, $PdT_x$ and palladium-deuteride alloys ($PdD_x$) using ultrasonic experimental techniques [27]. In this article, the authors observe complex behavior for both $C'$ and $B$, shown here in Figure 3.8. While these graphs only show measured data for H concentrations outside of the alpha-beta miscibility gap, it is apparent that the trends noticed in our simulations do not agree with these experimental measurements. It is important to note that, although the methods each used to measure elastic constants were quite different, the data presented by both Wolfer [38] and Schwarz *et al.* [27] are consistent with one another.

We also used our simulation method to evaluate the tensile strength of these systems when subjected to uniaxial stress loading conditions, and the results are shown in Table 3.3. This table shows tensile strength decreasing with increasing H content of the alloy. We also simulated similar systems of different crystal orientations in order to evaluate the tensile strengths of PdH for the [110] and [111] crystal directions. The results are shown in Table 3.4. While not conclusive, the results from both of these tables support the hypothesis that metal-hydrides are more vulnerable to failure due to brittle fracture than their pure metal counterparts. Thus, while fracture is not observed in the He bubble growth in pure Pd simulations by Foiles and Hoyt [10] and Zimmerman [11], it may occur in metal-hydride materials, thereby justifying the effort spent in developing an accurate material model.




(a) (b)

**Figure 3.8.** (a) Shear $C'$ and (b) bulk $B$ elastic constants measured by Schwarz *et al.* [27] for the $PdH_x$ (△), $PdD_x$ (◯) and $PdT_x$ (□) alloy systems. (taken from [27])

## 3.4 Further comments

It was noted in Chapter 2 that the inter-atomic potentials by both Daw and Baskes [20, 17] and Zhong *et al.* [25, 26] predict that interstitial hydrogen in a $PdH_x$ alloy alloy at high concentrations preferentially lie at tetrahedral sites on the FCC lattice, and that this prediction contradicts the experimentally observed behavior of hydrogen at octahedral interstitial sites. Hydrogen site selection has proven to be a major hurdle in the development of an accurate inter-atomic potential. As described in section 3.2, potential schemes that capture the correct octahedral occupancy at higher concentrations tend to overestimate the width of the miscibility gap. On the other hand, narrowing the concentration limits by increasing the energy of the octahedral site relative to the tetrahedral tends to destabilize the FCC structure. The crystal distortion shown in Figure 3.6 is a direct result of an energy that favors tetrahedral occupancy.

An attempt was made to determine the origin of this erroneous behavior. Density functional calculations were performed[2] using VASP for two systems. The first system is composed of 32 Pd atoms and 1 H atom, a "dilute" concentration of hydrogen in Pd. The second system is composed of 4 Pd atoms and 1 H atom, a $PdH_{0.25}$ stochiometry. For both of these systems, system energy was evaluated as a function of lattice parameter, with the H atom placed at either a tetrahedral or an octahe-

[2]These calculations were performed for the report authors by Thomas K. Mattsson, a technical staff member of the HEDP Theory department at Sandia National Laboratories, Albuquerque, NM.

**Table 3.3.** Tensile strengths for lattices of the $PdH_x$ binary system calculated using ParaDyn and the fitted potential.

| $x$ = H/Pd | $\sigma_{100}$ (GPa) |
|---|---|
| 0 | 11.10 |
| 0.25 | 9.42 |
| 0.5 | 8.18 |
| 0.75 | 6.14 |
| 1 | 4.95 |

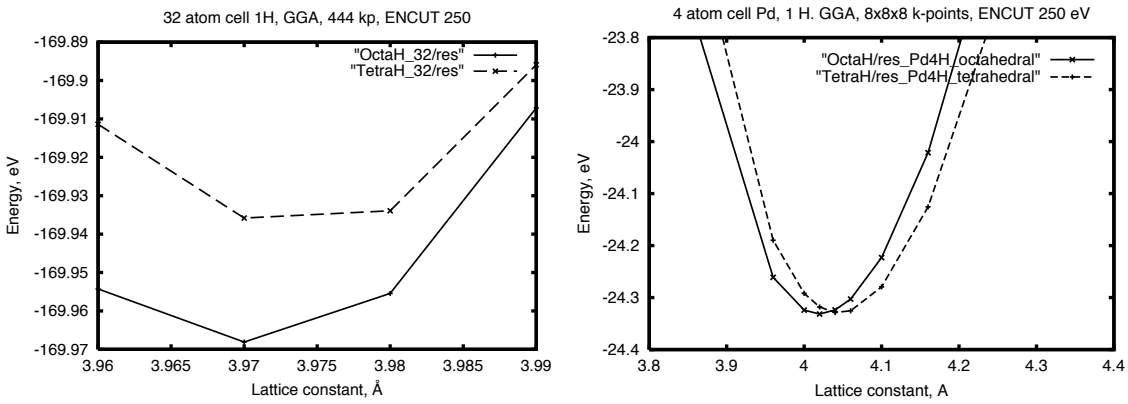
**Table 3.4.** Tensile strengths for PdH for different crystal directions calculated using ParaDyn and the fitted potential.

| Crystal Direction | $\sigma$ (GPa) |
|---|---|
| [100] | 4.95 |
| [110] | 0.298 |
| [111] | 0.112 |

dral interstitial site. The results of these calculations can be viewed graphically in Figure 3.9.

Figure 3.9(a) clearly shows that for dilute amounts of H in Pd, the octahedral site is energetically favorable to the tetrahedral site at all values of lattice parameter within the range examined. However, Figure 3.9(b) incorrectly predicts that for larger concentrations of H, the interstitial configuration is dependent on lattice parameter. For small values of lattice parameter, *i.e.* states of compression, the octahedral site is preferable; for large values, *i.e.* states of tension, the tetrahedral site is preferable. Figure 3.9(b) also shows that the two configurations have approximately equal energy in their minimized states, and that the octahedral configuration reaches its minimum at a slightly lower lattice parameter than the tetrahedral configuration does at its minimum.

An explanation for this anomalous behavior was recently put forth by Caputo and Alavi [39]. They too performed DFT calculations for the range of alloy compositions our group is interested in ($x = 0.25$ to 1), and discovered that tetrahedral site occupation was energetically preferable to octahedral occupation. Furthermore, they showed that this tetrahedral occupancy leads to a zincblende structure for the PdH stochiometry. To explain this, they hypothesize that the use of DFT omits vibrational




(a) (b)

**Figure 3.9.** System energy as a function of lattice parameter for H placed at tetrahedral and octahedral interstitial sites. (a) Dilute concentration of 1 H atom and 32 Pd atoms. (b) $PdH_{0.25}$ concentration with 1 H atom and 4 Pd atoms.

zero-point energy (ZPE), and that inclusion of ZPE is responsible for the octahedral occupation observed in experimental systems. This quantum effect is significant for light elements such as hydrogen, whereas the additional amount of ZPE may not be as significant for heavier elements that act as interstitial impurities.

While not rigorous, the work by Caputo and Alavi does show that care needs to be applied when using DFT results in the potential fitting process. The site selection problem is very subtle and is a concern in first principles modeling. Furthermore, construction of a physically realistic potential may require that additional actions be taken to ensure dominance of octahedral sites in the $PdH_x$ alloy system. This knowledge is used in our subsequent potential construction efforts, as will be shown.

This page intentionally left blank.

# Chapter 4

# Numerical Optimization of EAM Potential Parameters

In the previous chapter, we discussed our initial effort at building a Pd-H inter-atomic potential consisting of modifying the EAM potential by Daw and Baskes through the alteration of the pair potential to describe the Pd-H interaction. We showed how this effort did not meet with success, as the resulting potential is unable to display a physically-correct miscibility gap for a stable FCC lattice structure, shows mechanical behavior inconsistent with experimental observations, and incorrectly predicts tetrahedral site occupation for interstitial hydrogen. Our next step is to use and fit parameters for both the Pd-H and H-H pair interactions, $\phi_{\text{Pd-H}}$ and $\phi_{\text{H-H}}$, and for the atomic charge density function $f_{\text{H}}$ and the embedding energy function $F_{\text{H}}$ for hydrogen. To perform this fitting we used Sandia's optimization code DAKOTA [40] with the open source Monte Carlo code Towhee [41]. This chapter will discuss the methods we used and the results obtained.

## 4.1 Methods

We used the Sandia-originated optimization code DAKOTA [40] to search for an optimal fitting of potential parameters. This is achieved by minimizing an optimization function defined as

$$OF \equiv \sqrt{\frac{1}{n}\sum_{s=1}^{n}\left(E_s^{\text{A}} - E_s^{\text{VASP}}\right)^2}, \tag{4.1}$$

where $n$ configurations are considered, $E_s^{\text{VASP}}$ is the potential energy for configuration $s$ calculated using the DFT code VASP and $E_s^{\text{A}}$ is the potential energy for configuration $s$ calculated using an atomistic simulation code with the EAM potential and a trial set of fitted parameters. For our analysis, we used the Monte Carlo code Towhee [41] to evaluate the EAM potential and fitted parameters.

Table 3.1 contains the data used for the terms $E_s^{\text{VASP}}$. As mentioned earlier, these values were modified such that the equilibrium cohesive energy for a pure Pd system ($x = 0$) equals -15.64 eV/cell (-3.91 eV/atom) at a lattice parameter of 3.89 Å, values

consistent with experimental observation. Initially, this modification was to add a constant, offset amount of energy per cell from each of the systems listed in Table 3.1. For the fitting itself, only data from the $x = 0.25$, 0.5, 0.75 and 1 concentrations were used, a total of $n = 23$ configurations. Our goal in the fitting process was to optimize the potential parameters such that $OF < 0.01$ eV/atom, while still resulting in a potential that would reproduce a stable FCC lattice structure and a miscibility gap.

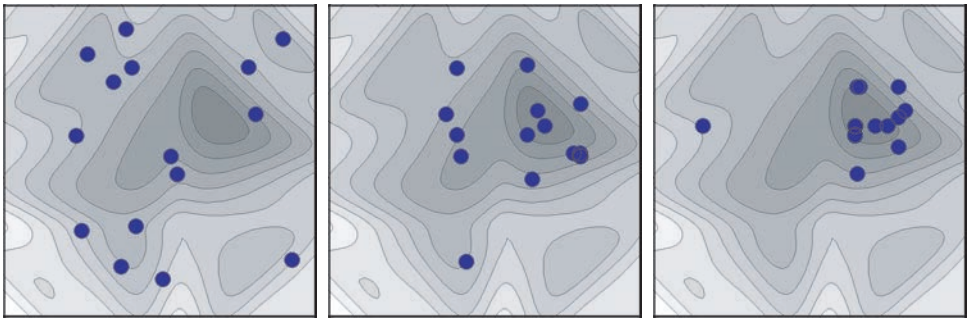
### 4.1.1 DAKOTA

DAKOTA (Design Analysis Kit for Optimization and Terascale Applications) is a C++ general purpose software toolkit used for the integration of commercial and in-house simulation capabilities with broad classes of systems analysis tools. The DAKOTA toolkit is intended as a flexible, extensible interface between simulation codes and iterative systems analysis methods. In addition to optimization methods, DAKOTA implements uncertainty quantification with sampling, reliability, and stochastic finite-element methods, parameter estimation with nonlinear least squares methods, and sensitivity/variance analysis with design of experiments and parameter study capabilities. These capabilities may be used on their own or as components within advanced strategies such as surrogate-based optimization, mixed integer nonlinear programming, or optimization under uncertainty. By employing object-oriented design to implement abstractions of the key components required for iterative systems analyses, the DAKOTA toolkit provides a flexible and extensible problem-solving environment as well as a platform for rapid prototyping of advanced methodologies which focus on increasing robustness and efficiency for computationally complex engineering problems. More information on Dakota can be found at the website http://www.cs.sandia.gov/DAKOTA/, and in references [42], [43] and [44].

The DAKOTA toolkit contains a variety of optimization strategies that can be used for parameter fitting. These include non-gradient based algorithms such as genetic algorithms, pattern search and surrogate optimization methods, and gradient-based algorithms such as finite-difference Newton and Gauss-Newton nonlinear least squares. Two optimization algorithms not offered by DAKOTA were also applied using the software package SNOPT [45] and Implicit Filtering [46, 47]. SNOPT is a state-of-the-art optimization software package that uses sequential quadratic programming to solve smooth optimization problems. Though this software package is developed to encompass large-scale optimization problems, it has been fine-tuned and rigorously tested on a large class of general test problems and hence is extremely reliable for smooth optimization problems. Implicit filtering is a steepest-descent algorithm for noisy problems and provides a unique blend of pattern search and finite-difference Newton. In the following subsections, we describe the basic premise for several of these algorithms, and their advantages and disadvantages.

#### 4.1.1.1 Genetic algorithms

The basic premise of a genetic algorithm (GA) is that the a population of points, *i.e.* trial parameter values, are followed from generation to generation using a "survival of the fittest" strategy to select new points. Also, the concept of "mutation", random and non-sequential modifications in the population, is used to break free from local minima. A schematic of how a GA works is shown in Figure 4.1.



(a) (b) (c)

**Figure 4.1.** Schematic of a genetic algorithm optimization sequence.

GA's are attractive in that they can optimize non-differentiable, noisy, discontinuous objective functions and can be shown under certain conditions to converge to the global optimum. Moreover, in practice GA's tend to perform quite well on a wide-range of problems. A disadvantage of GA's is that they tend to require substantially more function evaluations than pattern-search, can be slow to converge, and provide no mechanism for communicating to the user when a minimum has been found. Typically the user must run a GA several times, stopping the optimization process once improvements to the objective function become negligible or a budget of function evaluations has been expended.

#### 4.1.1.2 Pattern searches

Pattern search (PS) algorithms use a positive-spanning set of search directions to explore the feasible region. By doing so they provably converge to a local minimum by always containing at least one search direction that lies within 90° of the current steepest descent direction. PS algorithms do not require derivatives to exists in order to continue optimizing the objective and can in general handle noisy problems.

However, unlike a genetic algorithm, PS algorithms are not global in that they can get stuck at local solutions. This is primarily due to the lack of point mutations that can jump a genetic algorithm away from a local minimum to continue a global search. Despite this, PS algorithms in practice often avoid getting stuck at local solutions simply because they use many more search directions than a steepest descent algorithm and do not generate search directions based upon local information. Hence it is quite likely, especially during the early stages of optimization when the algorithm's step size is quite large, for PS algorithms to simply "step over" local minimums on the way to finding a more global solution.

A schematic for the sequence of a PS algorithm is shown in Figure 4.2. As this figure

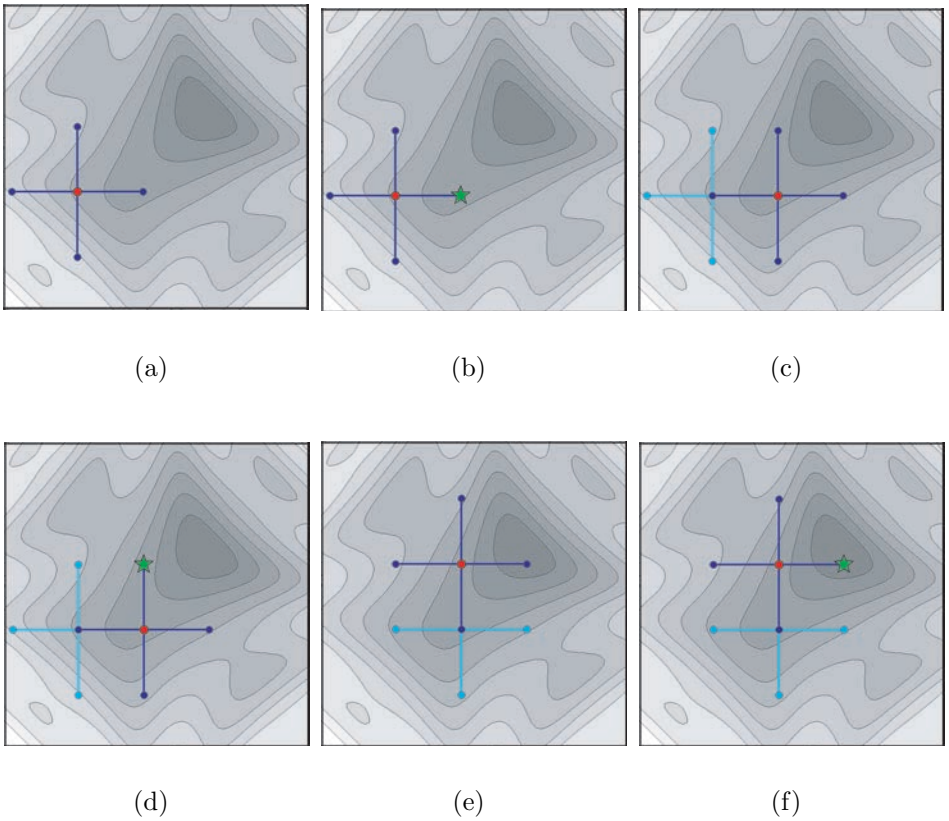



**Figure 4.2.** Schematic of a pattern search optimization sequence.

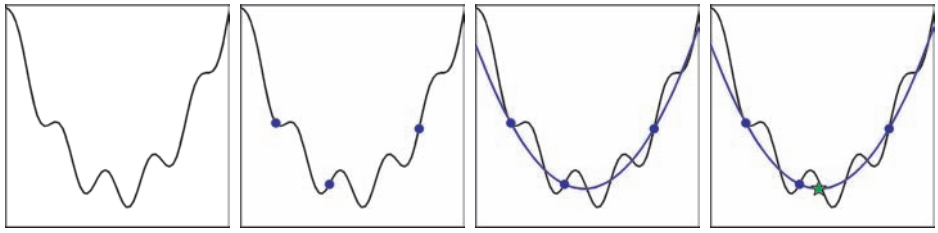
shows, at each step in the algorithm, the best point (i.e. the set of parameters that currently yields the best value of $OF$) is used to generate new trial points in many different "directions". These new points are evaluated and if a better point is found reducing the value of $OF$, the algorithm updates and moves to this point. Otherwise, the step size is reduced and new points are generated closer to the current best. This

process repeats iteratively until the current step size drops below a user specified tolerance. A characteristic unique to pattern search is that it can be shown that the step-size forms an implicit bound on the local optimality conditions.

PS algorithms tend to make good initial progress, but are slow to converge as the solution set approaches a local minimum. However, one advantage of PS algorithms that can dramatically reduce solve time is that evaluations can be performed asynchronously in parallel using multiple processors; for this purpose, the software package APPSPACK (Asynchronous Parallel Pattern Search) has been developed at Sandia and incorporated into DAKOTA [48].

#### 4.1.1.3 Surrogate models

In surrogate models (SM), points are sampled about the current best point (set of parameter values) to form an analytical model of the objective function. This model is then optimized to find a new best point. A schematic of a SM is shown in Figure 4.3.



(a) (b) (c) (d)

**Figure 4.3.** Schematic of a surrogate optimization sequence.

While the use of a surrogate model introduces an approximation that may oversimplify the optimization problem being considered, they nonetheless can be effective for engineering design problems with non-smooth objective functions that involve noise, slope discontinuities, multiple local minima, and other inhomogeneities found in the parameter space of the $OF$. However, unlike GA's and PS, undefined points and exponential growth in the objective functions can cause serious problems for surrogate-based optimization methods where the model is either undefined for a given set of points or must fit function values that vary dramatically in magnitude for relatively small changes in parameter values.

#### 4.1.1.4 Gradient based algorithms

In addition to the non-gradient based algorithms already discussed, some gradient based algorithms was also considered and used. Examples of such algorithms that were used for this research include finite-difference Newton (FDN) and Gauss-Newton nonlinear least squares (GN) algorithms, as well as the hybrid approach of implicit filtering (IF) that essentially combines FDN with PS.

FDN approximates first derivatives of the $OF$ using finite-differencing schemes in order to determine steepest descent directions in parameter space. FDN algorithms are excellent for finding local minima for smooth parameter spaces, and converge significantly faster than GA and PS approaches as the algorithm approaches a solution.

GN also relies on finite-differencing, but exploits the structure of nonlinear least squares. As with FDN, GN is excellent for finding local minima and has accelerated convergence under certain conditions.

IF exploits the feature that the standard compass search directions typically used by PS for bound constrained problems form a stencil of points similar to that used by FDN algorithm to approximate first-derivatives. Thus, by taking a step-size along each search direction that is smaller than normal for PS but larger than normal for FDN they are able to simultaneously perform PS and approximately (albeit with less accuracy than FDN) first derivatives. This ostensibly creates an algorithm that borrows from the best of both worlds, being able to fall back on pattern search when FDN would normally break-down (i.e. when encountering high levels of noise, undefined points, etc.), but obtain fast rates of convergence despite low levels of noise. A drawback of this approach is that currently no parallel implementation exists and hence evaluations must be done in serial.

SNOPT suffers similar drawbacks to IF in that currently no parallel implementation exists. As mentioned earlier, SNOPT is a sequential quadratic programming method that has been shown to be highly effective for solving constrained optimization problems on a wide range of problems. SNOPT is a highly-sophisticated derivative-based algorithm and approximates first derivatives in a similar fashion to FDN. The primary difference between the two algorithms being how the first derivatives are used to generate new trial points. We would expect in general for FDN and SNOPT to offer similar performance if the problem is sufficiently smooth. Thus SNOPT was primarily included to offer a "second opinion" on the derivative-based approach and to see if another algorithm might be able to improve performance. Again, SNOPT's primary drawback for our purposes was the fact that function evaluations must be performed in serial. (Note that SNOPT was specifically designed for problems with quick function evaluations and thousands of variables where the computational bottleneck results from the underlying linear algebra; thus there has been little effort attempting to parallelize this software. Though for our purposes it was the methods polished robustness on smooth problems that was appealing.)

### 4.1.2 Towhee

Towhee[41] is a Monte Carlo molecular simulation code originally designed for the prediction of fluid phase equilibria using atom-based force fields and the Gibbs ensemble with particular attention paid to algorithms addressing molecule conformation sampling. The code has subsequently been extended to several ensembles, many different force fields, and solid (or at least porous) phases. Towhee has made open-source code available through the SourceForge website. Information on using and developing the Towhee source code can be found at http://towhee.sourceforge.net/.

Towhee was used as the analysis code executed from within a DAKOTA optimization run. Towhee contains the flexibility to use both tables for inter-atomic potential functions and a number of analytical expressions for which parameters are read-in. This feature enabled us to use a tabular file containing an EAM potential for pure Pd[10], along with parameters used for remaining functions governing the Pd-H and H-H interactions. The choices of the specific functions that were used will be discussed later in this chapter. The parameters used with a Towhee execution are generated from each optimization iteration of the encapsulating DAKOTA execution.

## 4.2 Results

Initially, the genetic algorithm (GA) optimization method was used with Morse functions for Pd-H and H-H pair potentials,

$$\phi_{\text{Pd-H}}(r) = D_{\text{Pd-H}}\left\{\exp\left[-2\alpha_{\text{Pd-H}}\left(r - r_{\text{Pd-H}}\right)\right] - 2\exp\left[-\alpha_{\text{Pd-H}}\left(r - r_{\text{Pd-H}}\right)\right]\right\} \tag{4.2}$$

$$\phi_{\text{H-H}}(r) = D_{\text{H-H}}\left\{\exp\left[-2\alpha_{\text{H-H}}\left(r - r_{\text{H-H}}\right)\right] - 2\exp\left[-\alpha_{\text{H-H}}\left(r - r_{\text{H-H}}\right)\right]\right\}, \tag{4.3}$$

and an exponential function for the H atomic charge density function,

$$f_{\text{H}}(r) = C\exp\left[-\delta r\right], \tag{4.4}$$

where the constants $D_{\text{Pd-H}}$, $\alpha_{\text{Pd-H}}$, $r_{\text{Pd-H}}$, $D_{\text{H-H}}$, $\alpha_{\text{H-H}}$, $r_{\text{H-H}}$, $C$ and $\delta$ were fitted. A tabular spline was used to fit the embedding function $F_{\text{H}}$ in order to match the energy values listed in Table 3.1. As mentioned above, the EAM potential created by Foiles and Hoyt [10] was used for the Pd atomic charge density and embedding functions, and the Pd-Pd pair potential. The resulting fitting is shown below in Figure 4.4 and the energy versus lattice parameter curves for the four stochiometries is shown in Figure 4.5. As Figure 4.5 shows, the optimized fit is quite poor. The energy curves produced from our fitted potential are significantly higher in energy than the VASP data used for the fitting. In addition, the curvatures of these curves, indicative of the bulk modulus for the material, do not match and the energy curve for the PdH stochiometry contains unphysical oscillations.

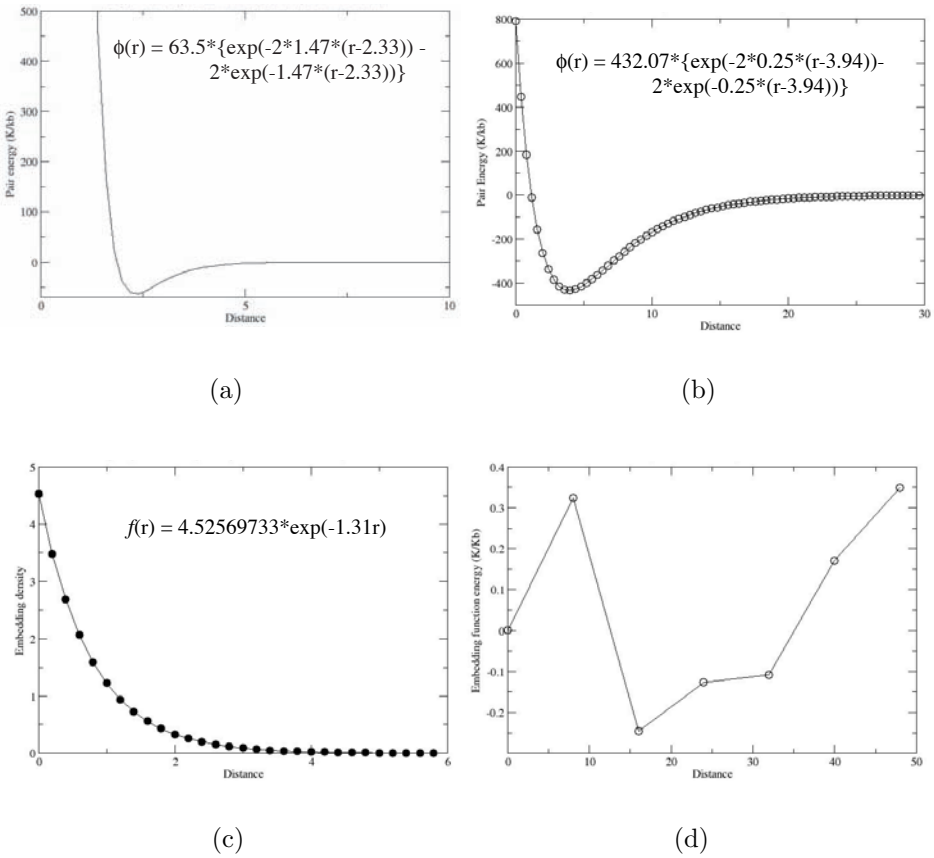



**Figure 4.4.** First iteration for EAM of Pd-H system (a) $\phi_{\text{Pd-H}}$ (b) $\phi_{\text{H-H}}$ (c) $f_{\text{H}}$ (d) $F_{\text{H}}$

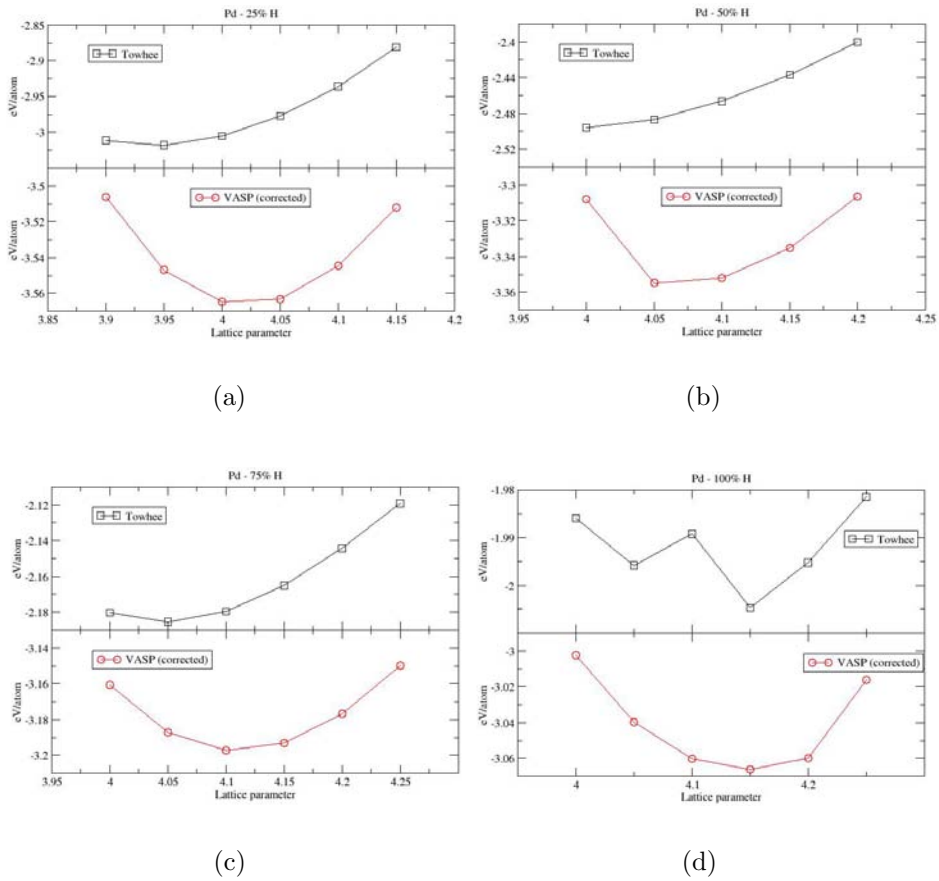



**Figure 4.5.** Energy per atom versus lattice parameter for first iteration of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H

Examination of these results and discussion regarding the fitting procedure resulted in improvements made to the fitting process. The first improvement was to use a more conventional, physically-based analytic expression for the H embedding function,

$$F(\rho) = \rho\left(A \ln[\rho] + B\right). \tag{4.5}$$

In addition, we also revised the fitting procedure to have a specific order. First, the Pd-H pair potential was fitted with all other H-related functions set to zero. Once this potential was optimally fitted, $F_{\mathrm{H}}$, $f_{\mathrm{H}}$ and $\phi_{\text{H-H}}$ were fitted with $\phi_{\text{Pd-H}}$ kept constant. After this second round of fitting, $\phi_{\text{Pd-H}}$ was re-fit to improve the overall fitting.

Using this process, the functions were refit to produce a potential of much higher accuracy. The resulting energy versus lattice parameter curves are shown in Figure 4.6, and the fitted parameters are listed in Table 4.1. This fitting produced an $OF =$

**Table 4.1.** Parameter values for the second iteration of EAM of the Pd-H system.

| Parameter | Value | Units |
|---|---|---|
| $D_{\text{Pd-H}}$ | 1.24061938021 | eV |
| $\alpha_{\text{Pd-H}}$ | 2.64074798 | $\text{Å}^{-1}$ |
| $r_{\text{Pd-H}}$ | 2.03390796 | Å |
| $D_{\text{H-H}}$ | 0.0 | eV |
| $\alpha_{\text{H-H}}$ | 0.0 | $\text{Å}^{-1}$ |
| $r_{\text{H-H}}$ | 0.0 | Å |
| $C$ | 104.405947 | |
| $\delta$ | 2.45154303 | $\text{Å}^{-1}$ |
| $A$ | $4.09822715253 \times 10^{-4}$ | eV |
| $B$ | $-1.29500616514 \times 10^{-3}$ | eV |

0.02455 eV/atom. Please note that for this fitting, the pair potential for the H-H interactions was zeroed out.

While the value of $OF$ achieved with this second fitting is not as low as the desired value, we did notice that this fitting produces an enthalpy of mixing curve that possesses a minimum in the neighborhood of 75% H, close to the beta phase boundary of 62%. This curve is shown in Figure 4.7, and indicates that this fitting could potentially produce a miscibility gap. However, application of the potential listed in Table 4.1 within a molecular dynamics simulation led to an unstable FCC structure for all of the H stochiometries.

Our next step was to fit all of the functions listed above (including the H-H pair potential) to our VASP database using a variety of optimization methods discussed in section 4.1.1. Figure 4.8, which revisits the energy per atom versus lattice parameter

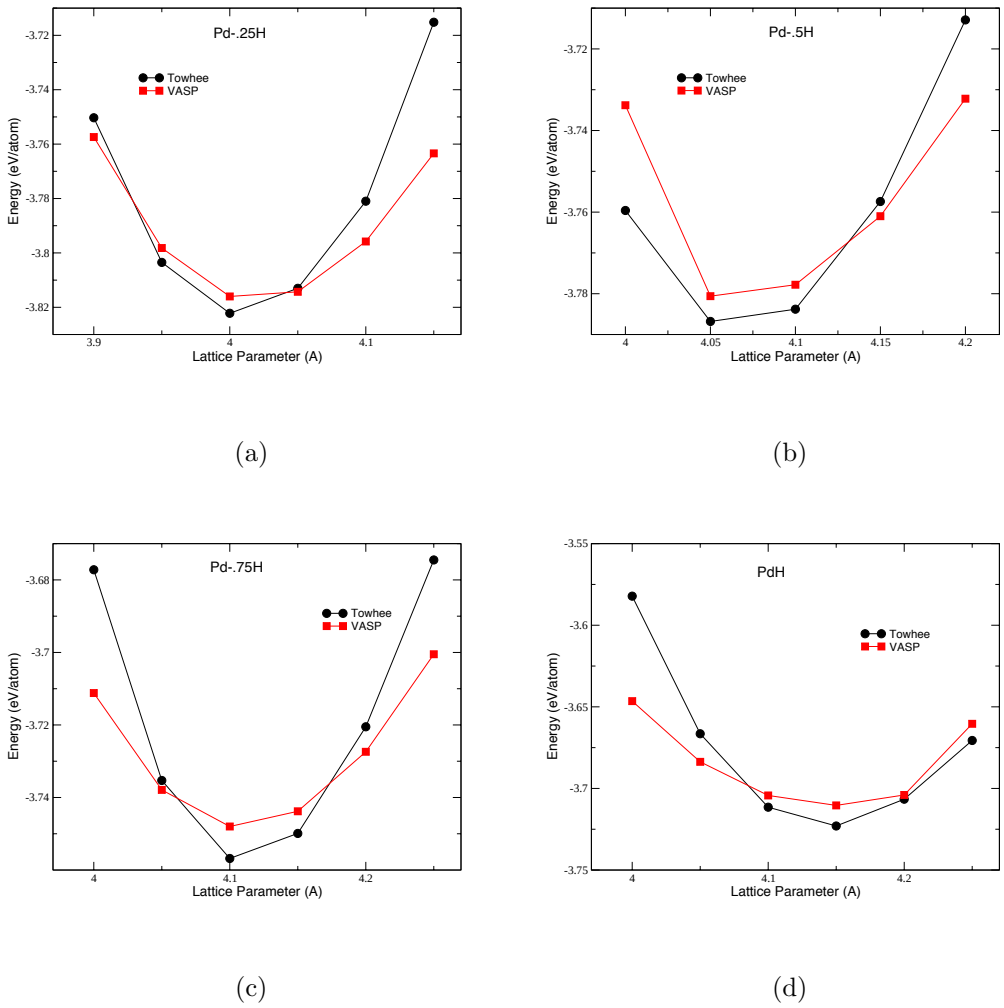



**Figure 4.6.** Energy per atom versus lattice parameter for second iteration of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H

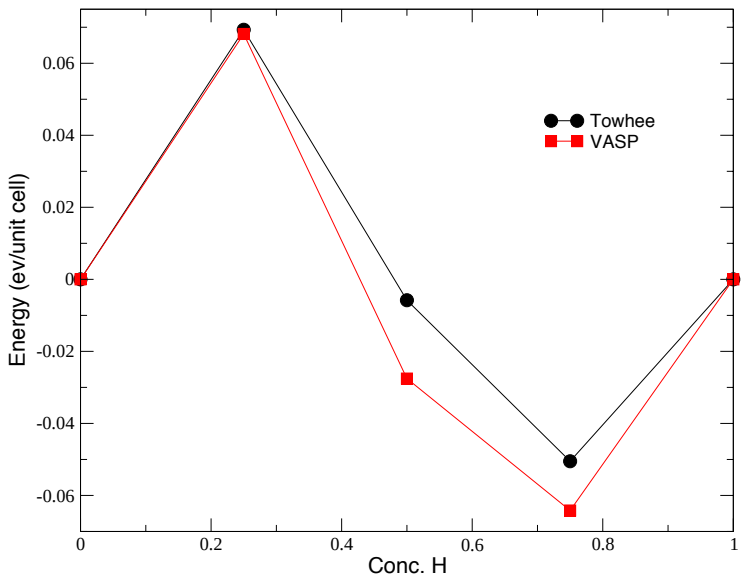



**Figure 4.7.** Energy of mixing as a function of H concentration.

curves for all four stochiometries, shows how the different methods compare with one another. Table 4.2 lists the $OF$ values for these different methods.

While Table 4.2 indicates that these revised fittings were not as good as our previous fit, both it and Figure 4.8 do show that the various methods are quite similar in the resulting energy curves they produce. The sole exception is the Surrogate Method, shown only in Figure 4.8(a). It is observed that the SM result is very much skewed as compared to the other methods, and produces a significantly higher $OF$ value. This behavior was noticed to be even more prominent for the higher H concentrations, thus the SM curves were eliminated in Figures 4.8(b), 4.8(c) and 4.8(d).

Another observation made in Figure 4.8 was that the energy curves for the fitted potentials were higher in energy as compared to the VASP data for the 50% and 75% H concentrations, but lower in energy for the 100% concentration. Combining this observation with the knowledge mentioned in section 3.4, it was hypothesized that the VASP data for PdH may be unphysical and its inclusion was negatively affecting the parameter fitting process. Additionally, the 25% H concentration was also viewed to be unphysical as experimental data shows that the alpha phase concentration is significantly lower than this.

We found the underlying optimization problem to be a difficult one for all optimization methods we applied. The objective function appeared to have convex behavior far

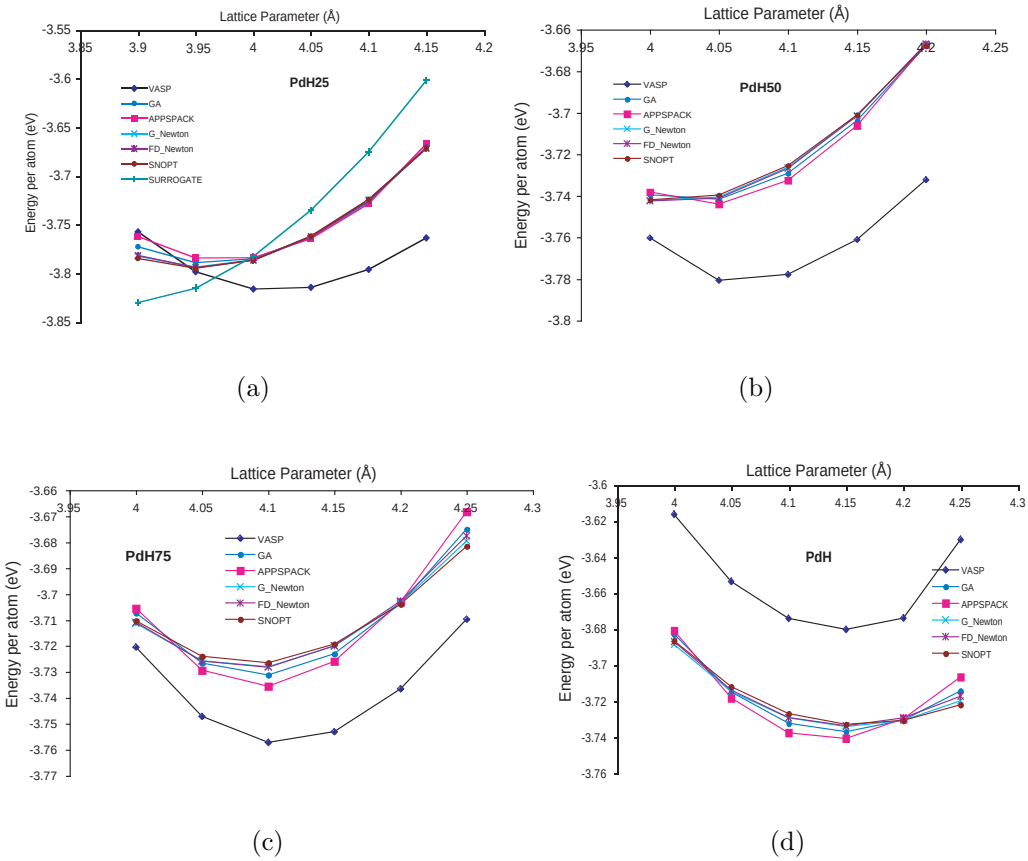



**Figure 4.8.** Energy per atom versus lattice parameter for various optimization methods applied to fitting of EAM of Pd-H system (a) 25% H (b) 50% H (c) 75% H (d) 100% H

**Table 4.2.** Objective Function values for various optimization methods used in the EAM fitting process.

| Method | $OF$ (eV/atom) |
|---|---|
| GA | 0.05056 |
| APPSPACK | 0.05063 |
| FDN | 0.05099 |
| SNOPT | 0.05135 |
| GN | 0.05148 |
| SM | 0.13589 |

from a minimum and hence all methods converge to points having similar quality in objective values. We suspect that the likely landscape of the objective functions looks a bit like Figure 4.2

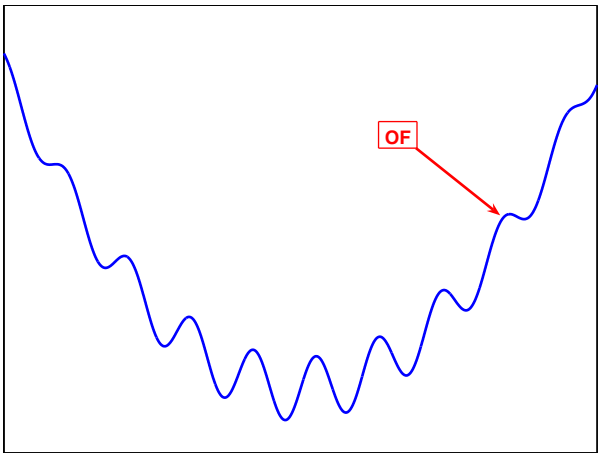



**Figure 4.9.** A one-dimensional hypothetical objective function that when far from the global minimum is easy to reduce, but becomes increasingly difficult for smaller values of $OF$.

Because the performance of the applied methods appeared to be similar for all methods save the surrogate-based approach (which we feel suffered from the exponential nature of our objective making it difficult to calculate and appropriate model) in Figure 4.5 we decided to use APPSPACK predominantly for future explorations. Evaluating the objective function of a given set of parameters took on average approximately 30 seconds. For a single run of the optimization algorithm we might perform more than 10,000 function evaluations, and we did hundreds of such runs repeatedly using different initial guesses, and different input parameters for each optimization algorithm. Thus, though the time to run Towhee was not prohibitive, it was neither a trivial amount of time when performing hundreds of thousands of such function evaluations. This present strong motivation for similarity in performance gains and the ability to perform all function evaluations asynchronously in parallel making it by far the fastest algorithm for this problem type.

As such, a reduced subset of the VASP database was created that contained only the 50% and 75% H concentration data. This subset, containing 11 values, was used in combination with parallel calculations performing asynchronous pattern searches (APPSPACK) to thoroughly explore the parameter space. We found that we were able to steadily reduce the objective value by keeping the bounds initially small (this affects scaling in APPSPACK), relatively close to the initial point, and moving a single upper or lower bound if ever it became active. An inequality constraint is considered to be active if the current point satisfies this constraint with equality. By keeping the parameter bounds relatively close to the current best point we were able

to restrict the PS to exploring the nearby region with relatively large steps. This seemed to provide the greatest gains when the value of the objectives dropped below .05. We suspect this is due to the way APPSPACK scales points in order to map the bound constrained region to the unit hypercube. Such a mapping is less sparse if the upper and lower bounds for a given set of parameters are closer together. Thus, perhaps counterintuitively, by using bounded that were closer together we were able to take much larger steps towards the best solution found.

Thus we iteratively solved problems of the form for an initial best guess $x_0$

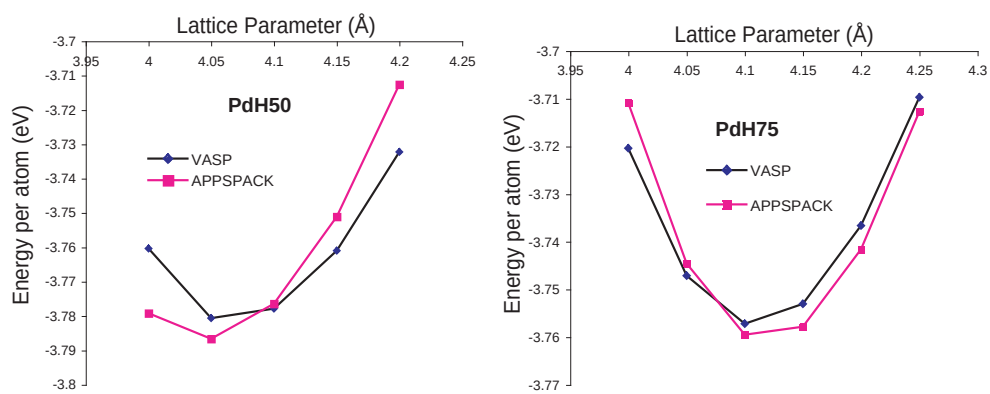
$$\begin{array}{ll} \text{minimize}_{x \in \mathbb{R}^{11}} & OF(x) \\ \text{subject to} & \ell_k \leq x \leq u_k \end{array}$$

where

$$\ell_k = x_0 - 10e \text{ and } u_k = x_0 + 10e.$$

Here $e$ is used to denote the vector of all ones. If the optimization process converged to a point $x^*$ such that $x^a st$ satisfied a lower or upper bound exactly, a new problem was then defined for $x_0 = x^*$ and $k = k + 1$. This process appeared to be an effective way of appropriately scaling the nearby region so that the optimization process could continue.

Figure 4.10 and Table 4.3. This fitting corresponded to $OF = 0.00971$ eV/atom,




(a) (b)

**Figure 4.10.** Energy per atom versus lattice parameter for fitting of EAM of Pd-H system (a) 50% H (b) 75% H

which meets one of our goals mentioned at the beginning of this chapter. However, application of these parameters produce an unstable potential, *i.e.* the FCC lattice

**Table 4.3.** Parameter values for the an EAM potential of the Pd-H system.

| Parameter | Value | Units |
|---|---|---|
| $D_{\text{Pd-H}}$ | 0.046308612 | eV |
| $\alpha_{\text{Pd-H}}$ | 2.7814474244 | $\text{Å}^{-1}$ |
| $r_{\text{Pd-H}}$ | 2.125 | Å |
| $D_{\text{H-H}}$ | 0.492445828 | eV |
| $\alpha_{\text{H-H}}$ | 0.78779185486 | $\text{Å}^{-1}$ |
| $r_{\text{H-H}}$ | 4.2905611785 | Å |
| $C$ | 1000.0 | |
| $\delta$ | -50.0 | $\text{Å}^{-1}$ |
| $A$ | 0.027293309 | eV |
| $B$ | -0.093327738 | eV |

structure is not stable, and predict a cohesive energy of hydrogen of -10 eV/atom or more. Calculations made with VASP [1] predict the correct value of hydrogen cohesive energy is between -1 and -2 eV/atom.

One last attempt was made to fit the full VASP database by starting with the parameter values listed in Table 4.3, and performing an additional optimization using the Gauss-Newton method. The results for this full database fitting are shown in Figure 4.11 and Table 4.4. This fitting corresponded to $OF = 0.01932$ eV/atom, a value not as accurate as our previous fitting (as expected), but very close to our goal value of 0.01 eV/atom. As Figure 4.11 shows, the resulting energy versus lattice parameter curves are not accurate to the VASP database. The inclusion of the 25% and 100% data creates a significant offset for the 75% curve, and only moderate agreement for the other three concentrations. Also apparent is the fact that refitting to the original database dramatically alters the optimized values of several of the parameters. In particular, $\alpha_{\text{Pd-H}}$, $D_{\text{H-H}}$, $\alpha_{\text{H-H}}$, and $r_{\text{H-H}}$ have changed significantly. Also, the value of $\delta$ decreases by an order of magnitude. While its value in Table 4.3 is such that the atomic charge density function $f_{\text{H}}$ is effectively zero, it now (Table 4.4) becomes quite significant although oddly the parameters associated with the H embedding function $F_{\text{H}}$ have changed very little.

[1] These calculations were performed for the report authors by Eric H. Majzoub, a technical staff member of the Energy Systems department at Sandia National Laboratories, Livermore, CA.

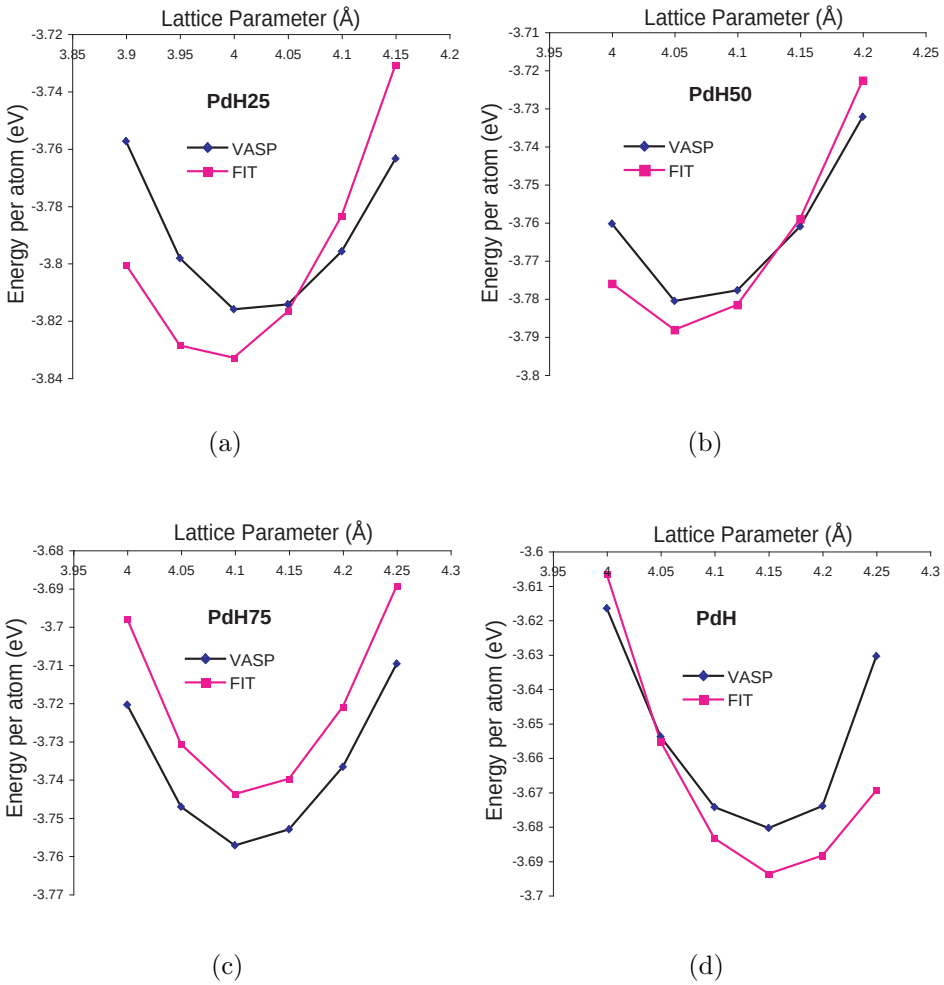



**Figure 4.11.** Energy per atom versus lattice parameter for fitting of EAM of Pd-H system (a) 25% H, (b) 50% H, (c) 75% H, (d) 100% H.

**Table 4.4.** Parameter values for an EAM potential of the Pd-H system.

| Parameter | Value | Units |
|---|---|---|
| $D_{\text{Pd-H}}$ | 0.041165488 | eV |
| $\alpha_{\text{Pd-H}}$ | 7.4929546066 | $\text{Å}^{-1}$ |
| $r_{\text{Pd-H}}$ | 2.0435229474 | Å |
| $D_{\text{H-H}}$ | 0.005678956 | eV |
| $\alpha_{\text{H-H}}$ | 0.10610757463 | $\text{Å}^{-1}$ |
| $r_{\text{H-H}}$ | 24.476985364 | Å |
| $C$ | 97.266879869 | |
| $\delta$ | -5.6509751317 | $\text{Å}^{-1}$ |
| $A$ | 0.028236149 | eV |
| $B$ | -0.095314319 | eV |

## 4.3 Concluding comments

In this chapter, work has been presented that demonstrates that application of numerical optimization techniques, namely the combined use of DAKOTA and Towhee, brings both power and speed to the parameter fitting process. Also clear is that these methods have improved the quality of the fit as evidenced by the improved accuracy in the curvatures for the energy curves shown in Figures 4.10 and 4.11 as compared with Figure 4.6, and the reduced value used as the metric for fitting ($OF$). Our goal of $OF < 0.01$ eV/atom was realized by fitting to the subset of the VASP database.

Ultimately, the underlying optimization problem as stated was apparently nonconvex (evidenced by the existence of multiple local minimums) and hence extremely difficult to solve. These local minima seemed to exist predominantly for parameters corresponding to small (but not sufficiently small) values of $OF$, i.e. when $OF \approx .05$. This had the effects of making the optimization problem exceedingly difficult as derivative-based methods tended to get stuck in one of these small local minimums while the derivative-free approaches continued to optimize at increasingly slower rates.

Furthermore, examination of the improved optimized parameter sets has shown a lack of consistency in fitted parameter values. This may indicate instabilities associated with using these functional forms with the VASP database. The underlying mathematical difficulties of solving such a problem compounded with the persistence of instabilities in the FCC lattice structures has motivated use to consider more sophisticated models that may be less challenging mathematically while offering a more stable and hence usable model. In the next chapter, we rethink our approach by using analytic expressions to overcome these instabilities from the beginning of the fitting process, as will be seen.

# Chapter 5

# Analytical Approach to EAM Parameter Fitting

Palladium possesses the capacity of absorbing a large amount of hydrogen to form a hydride.[49] This property enables many important applications such as hydrogen storage for clean portable energy, new refrigerator designs, catalytic converters, and nuclear radiation adsorption.[50, 51, 49, 52, 53, 54, 55] Palladium hydrides are also suited for safe and efficient processing of hydrogen isotopes such as tritium.[56, 57, 58] Materials used for these applications must have good mechanical properties. It has been well known, however, that the dissolution of even a small amount of hydrogen in metals significantly deteriorates their properties, resulting in embrittlement, crack propagation, and corrosion.[59, 60, 61] There is another significant problem for the application of tritides due to the natural radioactive decay of tritium to form helium atoms. These helium atoms are insoluble in palladium, and they tend to gather into bubbles of nanometric size.[62, 2] The high bubble pressure exerts significant stresses that are large enough to cause lattice swelling, crack formation, and quick release of helium due to the coalescence of defects.[63, 64] Understanding these phenomena is therefore critical in order to predict the time evolution (aging behavior) of material properties.

Hydrogen can be continuously pumped into the palladium lattice as the surrounding hydrogen gas pressure is increased. As the hydrogen content increases, the solid solution first forms a low hydrogen concentration $\alpha$ phase and then a high hydrogen concentration, but more defective $\beta$ phase. In both $\alpha$ and $\beta$ phases, palladium atoms fully occupy a face-centered-cubic (f.c.c.) lattice whereas hydrogen atoms partially occupy the octahedral interstitial sites of the palladium lattice. The pressure-composition curve exhibits a plateau (isotherm) in an intermediate composition regime where $\alpha$ and $\beta$ phases coexist. At room temperature and ambient pressure, the $\alpha$ and $\beta$ phases exist at approximate concentrations of H/Pd = 0.03 and 0.60, respectively.[15]

The pressure-composition isotherm is a central phenomenon for the application of the hydride[65], and may also affect mechanical properties. For instance, inserting hydrogen into the octahedral sites of the palladium f.c.c. lattice expands the lattice in linear proportion to the hydrogen content.[50] The dislocation density increases during the hydriding process for accommodating the lattice distortion induced by the

coexistence of phases, thereby causing the emergence of fracture.[66]

The technological importance has stimulated an interest to model $PdH_x$ hydride. Extensive first-principles calculations have been carried out.[54] Larger scale molecular dynamics (MD) or Monte Carlo (MC) simulations have not been widely used due to the lack of a high fidelity Pd-H interatomic potential that is applicable to the hydrogen composition range $0 \leq x \leq 1$ and is capable of predicting the pressure-composition isotherm (i.e., the $\alpha$ and $\beta$ phase miscibility gap). Numerous Pd-H potentials have been developed.[17, 20, 18, 67] However, these potentials are mainly designed for dilute hydrogen compositions, and none of them have been found to correctly predict the miscibility gap. Some potentials even incorrectly predict the occupancy of tetrahedral interstitial sites by hydrogen atoms.

To our knowledge, the Pd-H embedded atom method (EAM) interatomic potential developed by Wolf et al is the only one that has been successfully used in MC simulations to qualitatively predict the pressure-composition isotherm.[16] However, in their 1993 publication the authors do not present the functions and parameters that dictate the Pd-H and H-H interactions; rather, they refer only to their earlier 1992 publication[28] that presents information only on an EAM potential for pure Pd. Extensive efforts have been made to obtain this information[68], but to our knowledge specifics on the potential have been lost. In addition, no testing of this potential with regard to mechanical properties, e.g. elastic constants, tensile and shear strengths, has been performed.

In this chapter, we develop a Pd-H interatomic potential that is applicable to the entire hydrogen composition range $0 \leq x \leq 1$ and is capable of predicting the miscibility gap. Details regarding how model functions and parameters were obtained will be described, and physical and mechanical properties of the resulting potential will be explored.

## 5.1 Embedded Atom Method Potential

While there are a variety of mathematic formalisms for interatomic potentials, the EAM potential has an advantage of being both computationally efficient and suited to f.c.c. metal systems. EAM potential was initially developed by Daw and Baskes.[20, 17] It improves over pair potentials by incorporating the environment dependence of the atomic interactions. As a result, it has been successfully applied to a variety of metal and metal alloy systems.[69, 70] EAM has also been successfully applied in some of the metal hydride problems.[71] EAM is hence chosen as the format for our Pd-H potential.

In EAM, the cohesive energy of a computational system composed of N atoms is

expressed as

$$E_c = \frac{1}{N}\left[\sum_{i=1}^{N} F_i\left(\rho_i\right) + \frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} \phi_{ij}\left(r_{ij}\right)\right] \tag{5.1}$$

where $F_i\left(\rho_i\right)$ is an embedding energy at site of atom $i$ with a background electron density of $\rho_i$, and $\phi_{ij}\left(r_{ij}\right)$ is a pair energy between atoms $i$ and $j$ separated by a distance $r_{ij}$. Electron density at the site of atom i is calculated as

$$\rho_i = \sum_{\substack{i=1\\ j\neq i}}^{N} \rho_j^a\left(r_{ij}\right) \tag{5.2}$$

where $\rho_j^a\left(r_{ij}\right)$ is electron density contribution from atom $j$ that is $r_{ij}$ away from site $i$. This EAM potential requires three functions for each element (e.g., $i = Pd, H$): the embedding energy $F_i\left(\rho\right)$ as a function electron density $\rho$, the pair energy $\phi_{ii}\left(r\right)$ as a function of atom distance $r$, and the atomic electron density $\rho_i^a\left(r\right)$ as a function atom distance $r$. In addition to these three functions for each of the elements, the cross pair energy between dissimilar species $i$ and $j$, $\phi_{ij}\left(r\right)$, is required for the alloy system.

## 5.2 Palladium potential

Numerous EAM potentials have already been developed for palladium.[18, 69, 72] All of these potentials well captured the properties of palladium. In particular, the potential developed by Foiles and Hoyt[10], which incorporates the universal equation of state derived by Rose et al.[31], reproduces exactly the experimental lattice constant, cohesive energy, and bulk modulus of the f.c.c. palladium crystal. It also well predicts other elastic constants, vacancy formation energy and its relaxation volume, various defect energies and their relaxation volumes, defect migration energies, stacking fault energies, and various surface energies and surface relaxations. These properties are important for capturing the mechanical properties of palladium. The same palladium EAM potential has also been used to successfully study the effects of helium bubbles in palladium matrix.[10, 11] As a result, we construct our Pd-H potential based upon this palladium EAM potential. This palladium potential is available in tabular form and is included in molecular dynamics simulation packages released from Sandia National Laboratories[37].

The EAM potential has a special property in that an elemental EAM potential is invariant to a transformation to the embedding function,

$$G\left(\rho\right) = F\left(\rho\right) + k\cdot\rho \tag{5.3}$$

and a concurrent transformation to the pair potential

$$\psi\left(r\right) = \phi\left(r\right) - 2k\cdot\rho^a\left(r\right) \tag{5.4}$$

where $k$ is an arbitrary constant. While $k$ does not change the properties of the element, it does change the properties of the alloys when the elemental potentials are combined.[73, 74, 75, 17] During potential parameterization, an optimization procedure is used to minimize an objective function defined as the sum of the square deviation of predicted properties from the corresponding target (either experimental or ab initio calculated) properties. At first sight, it appears that $k$ can be treated as a fitting parameter so that the formulism has more freedom to better minimize the objective function. However, we found that this scheme caused a drift of $k$ to unphysical values, resulting in potentials that failed to predict the stable alloy crystal structures during MD simulations even though these structures were statically well fitted. This problem can be overcome by uniquely defining $k$ using a normalization condition that requires the equilibrium to be independently achieved for embedding and pair energy functions for the most stable elemental crystal phase. By setting the first derivative of the embedding energy to zero, $k$ can be solved and the normalized Pd embedding and pair energy functions are expressed as

$$F_{Pd}(\rho) = F_{Pd,u}(\rho) - F'_{Pd,u}(\rho_{0,Pd})\,\rho \tag{5.5}$$

$$\phi_{PdPd}(r) = \phi_{PdPd,u}(r) + 2F'_{Pd,u}(\rho_{0,Pd})\,\rho^{a}_{Pd}(r) \tag{5.6}$$

where $F_{Pd,u}(\rho)$ and $\phi_{PdPd,u}(r)$ are the original (unnormalized) palladium embedding and pair energy functions[10], $F'_{Pd,u}(\rho_{0,Pd})$ is the first derivative of the original palladium embedding energy evaluated at the equilibrium electron density of the most stable crystal phase of palladium, $\rho_{0,Pd}$, and $F_{Pd}(\rho)$ and $\phi_{PdPd}(r)$ are the corresponding normalized functions used in the present work.

## 5.3 Hydrogen Potential and Pair Interaction between Palladium and Hydrogen

With palladium potential known, the remaining work is to develop a hydrogen potential and a cross pair potential between palladium and hydrogen. It is more convenient to use analytical expressions. We first consider the elemental potential that does not require normalization.

The pair potential is approximated by

$$\phi_{HH,u}(r) = \beta_{HH}\cdot D_{HH}\cdot exp\left[-\alpha_{HH}(r - r_{0,HH})\right] - \alpha_{HH}\cdot D_{HH}\cdot exp\left[-\beta_{HH}(r - r_{0,HH})\right] \tag{5.7}$$

where $D_{HH}$, $\alpha_{HH}$, $\beta_{HH}$, and $r_{0,HH}$ are four positive constants, and in particular, $r_{0,HH}$ represents the equilibrium bond length of a dimer, and $D_{HH}(\beta_{HH} - \alpha_{HH})$ is its binding energy. Eq. 5.7 well approximates the pair energy because it is more general than a Morse potential, with the latter being a special case of Eq. 5.7 at $\alpha_{HH}/\beta_{HH} = 2$.

Within the interaction range usually encountered in atomistic simulations, electron density increases as atomic separation decreases. An exponential decay function well captures this behavior and is used to approximate the hydrogen electron density function:

$$\rho_H^a(r) = C_H \cdot exp(-\delta_H \cdot r) \tag{5.8}$$

where $C_H$ and $\delta_H$ are two positive constants.

The embedding energy must have a zero value at zero electron density where atoms are far separated. Previously used embedding energy functions are usually constructed by further requiring that they have a negative slope at small electron densities and a positive slope at large electron densities.[20, 17] The function $F(\rho) = A_1 \cdot \rho \cdot ln(\rho) + A_2 \cdot \rho$ satisfies these requirements, where $A_1$ (>0) and $A_2$ are two constants. A problem with this function is that it has a negative infinity slope at the point $\rho = 0$. Here we use an embedding energy function form that has a finite negative slope at $\rho = 0$ and a positive slope at large electron densities. It is expressed as

$$F_{H,u}(\rho) = -c_H \left( \frac{1}{2+d_H}\rho^{2+d_H} - \frac{a_H + b_H}{1+d_H}\rho^{1+d_H} + \frac{a_H \cdot b_H}{d_H}\rho^{d_H} \right) \tag{5.9}$$

where $a_H$, $b_H$, $c_H$, and $d_H$ are four positive constants. Here in particular, $a_H$ is the electron density at which the embedding energy is a minimum, and as long as $b_H$ is larger than any electron density that can be encountered during simulations, Eq. 5.9 does not have other minimum or maximum points within the electron density range simulated. The parameter $d_H$ is introduced to make the function more flexible to better fit the material properties.

Again for alloy systems, Eqs. 5.7 and 5.9 are normalized to give,

$$F_H(\rho) = F_{H,u}(\rho) - F'_{H,u}(\rho_{0,H}) \cdot \rho \tag{5.10}$$

$$\phi_{HH}(r) = \phi_{HH,u}(r) + 2F'_{H,u}(\rho_{0,H}) \cdot \rho_H^a(r) \tag{5.11}$$

The EAM model described in the above has another special property in that the elemental potential remains invariant to a transformation to the electron density contribution function

$$\mu_H^a(r) = \rho_t \cdot \rho_H^a(r) \tag{5.12}$$

as long as a concurrent transformation is made to the embedding energy

$$G_H(\rho) = F_H\left(\frac{\rho}{\rho_t}\right) \tag{5.13}$$

where $\rho_t$ is an arbitrary constant, and $\mu_H^a$ and $G_H$ are the transformed functions. This arises because the electron density is defined in a relative unit. While $\rho_t$ does not affect elemental properties, it does affect the alloy properties. Again we found that $\rho_t$ cannot be treated as a fitting parameter as the minimization of the objective function causes it to drift to an unphysical value that fails to stabilize the fitted crystal structures during MD simulations. It is recognized that the equilibrium electron densities of

different elements must be measured in the same unit. While their values are most likely to be different, they should not differ by orders of magnitude. We can therefore normalize the electron density by imposing the constraint

$$\rho_{min} \leq \rho_{0,i} \leq \rho_{max} \tag{5.14}$$

where $\rho_{0,i}$ is the equilibrium electron density in the most stable crystal structure of element $i$, and $\rho_{min}$ and $\rho_{max}$ are lower and higher bounds for electron density that is common to all elements. With proper electron density bounds, Eq. 5.14 ensures that the electron density units of different elements are compatible at least in terms of order of magnitude but also provides sufficient freedom for each element to adjust its own electron density. We found that with the two normalizations described above, well-behaved alloy potentials were always obtained.

Finally, the cross pair potential between palladium and hydrogen is simply approximated by the generalized Morse potential discussed earlier,

$$\phi_{PdH}(r) = \beta_{PdH} \cdot D_{PdH} \cdot exp\left[-\alpha_{PdH}\left(r - r_{0,PdH}\right)\right] - \alpha_{PdH} \cdot D_{PdH} \cdot exp\left[-\beta_{PdH}\left(r - r_{0,PdH}\right)\right] \tag{5.15}$$

## 5.4 Palladium-Hydrogen Solid Solution Model

### 5.4.1 Gibbs Free Energy of Mixing

The phase miscibility is determined by the Gibbs free energy of mixing per atom as a function of composition in units of mole fraction. Here we first examine the function of Gibbs free energy of mixing.

The Gibbs free energy of mixing is expressed as

$$\Delta G^{mix} = \Delta H^{mix} - \Delta S^{mix} \cdot T \tag{5.16}$$

where $\Delta H^{mix}$ and $\Delta S^{mix}$ are enthalpy and entropy of mixing (per atom) respectively, and $T$ is the temperature. The reference materials of the mixing can be arbitrary, but here it is convenient to define the mixing as the process of forming $PdH_x$ from two reference materials Pd and PdH. Consider the mixing reaction

$$\left[N_{Pd} \cdot x\right] PdH + \left[N_{Pd} \cdot (1 - x)\right] Pd \rightarrow \left[N_{Pd}\right] PdH_x \tag{5.17}$$

where $N_{Pd}$ is total number of Pd atoms involved in the mixing. As can be seen, the factors $N_{Pd} \cdot x$, $N_{Pd} \cdot (1-x)$, and $N_{Pd}$ in Eq. 5.17 represent the numbers of (chemical formula unit) the three corresponding materials PdH, Pd, and $PdH_x$. Assume that the cohesive energies (per atom) for PdH, Pd, and $PdH_x$ are $E_{PdH}$, $E_{Pd}$, and $E_{PdH_x}$ respectively, the total change of energy for Eq. 5.17 is

$$\Delta E_t = N_{Pd}\left(1 + x\right) E_{PdH_x} - 2N_{Pd} \cdot x \cdot E_{PdH} - N_{Pd}\left(1 - x\right) E_{Pd} \tag{5.18}$$

The change of energy per atom is then

$$\Delta E = \frac{(1+x)\,E_{PdH_x} - 2x \cdot E_{PdH} - (1-x)\,E_{Pd}}{1+x} \tag{5.19}$$

Replacing x with mole fraction X = x/(1+x), we have the heat of mixing

$$\Delta H^{mix} = E_{PdH_x} - 2\ X\ E_{PdH} - (1-2X)\ \ E_{Pd} \tag{5.20}$$

which is consistent with the conventional definition of heat of mixing.[76, 77]

The entropy of mixing can be approximated by the change of the configurational entropy. The $PdH_x$ lattice is composed of two sublattices occupied by palladium and hydrogen atoms respectively. There is no configurational entropy on the palladium sublattice as it is fully occupied. The hydrogen sublattice can be viewed as forming a substitutional H-vacancy "alloy". The total configurational entropy of the hydrogen sublattice is then

$$\Delta S_t = -N_{Pd} \cdot k\,[x \cdot ln\,(x) + (1-x)\,ln\,(1-x)] \tag{5.21}$$

where k is Boltzmann constant. Notice that the total number of palladium atoms $N_{Pd}$ equals the total number of sublattice sites (i.e., the total number of hydrogen atoms and vacancies). The configurational entropy per atom is then

$$\Delta S = -\frac{k\,[x \cdot ln\,(x) + (1-x)\,ln\,(1-x)]}{1+x} \tag{5.22}$$

Again replacing x with mole fraction X, we have an entropy of mixing expression

$$\Delta S^{mix} = -k\left[X \cdot ln\frac{X}{1-X} + (1-2X)\,ln\frac{1-2X}{1-X}\right] \tag{5.23}$$

Eqs. 5.16, 5.20, and 5.23 define a Gibbs free energy of mixing. It should be noted that in classic molecular dynamics simulations, the heat capacity is approximated as independent of materials. In that case, the temperature does not affect heat of mixing. As a result, Eq. 5.20 can be used for other temperatures even if it is derived at 0 K.

### 5.4.2 Cohesive Energy as a Function of Composition

While the cohesive energy of any phase can be calculated from the interatomic potential and Eq. 5.1 using a computational system, the approach is too computationally expensive when applied for a wide range of compositions and different structures (e.g., f.c.c. palladium with hydrogen at octahedral as well as tetrahedral interstitial sites). Here we use a solid solution model to derive an approximate analytical equation for cohesive energy as a function of composition and structure. This solid solution model assumes an ideally random distribution of atoms so that each palladium or hydrogen

atom sees the same environment. While this model greatly improved the calculation efficiency, it also produced results that were sufficiently close to those obtained from MD simulations of real systems.

At first sight, it was not clear whether hydrogen needed to be included in the potential parameterization, as our goal is to model various compositions of the $PdH_x$ alloy system rather than solid or molecular hydrogen. However, we found that if hydrogen properties were not included in the parameterization, the minimization of the objective function almost always drove the hydrogen cohesive energy to an unrealistically high magnitude, resulting in abnormal hydrogen segregation that distorted and unstabilized the fitted alloy phases during MD simulations. Here, we derive a cohesive energy equation that includes the hydrogen phase. For this purpose, we use a more general chemical formula for our compound system, $Pd_yH_x$, so that it includes both hydrogen ($y = 0$, $x = 1$) and $PdH_x$ ($y = 1$) phases.

In the formulation of Gibbs free energy of mixing described above, we considered only the equilibrium $PdH_x$ structure where hydrogen atoms occupy the octahedral interstitial sites of the palladium lattice. During the fitting of the cohesive energy, it is important to ensure that this equilibrium phase has the lowest cohesive energy. This means that the cohesive energies of both equilibrium and non-equilibrium phases must be included in the fitting. For this purpose, we assume a general $Pd_yH_x$ lattice containing several f.c.c. sublattices, where palladium atoms occupy one f.c.c. sublattice and hydrogen atoms occupy other f.c.c. sublattices. Two structures considered are shown in Fig. 5.1, where in Fig. 5.1(a), hydrogen atoms occupy the octahedral interstitial sites, and in Fig. 5.1(b) hydrogen atoms occupy the tetrahedral interstitial sites. Accordingly, we denote the structures shown in Figs. 5.1(a) and 5.1(b) by "OC" and "TE" respectively. It can be seen that there is only one f.c.c. sublattice for the octahedral interstitial sites so that the maximum composition for the OC structure is $x = 1$ (note that $x$ can also be viewed as the occupancy probability on the sublattice). On the other hand, there are two f.c.c. sublattices for the tetrahedral interstitial sites so that the maximum composition for the TE structure is $x = 2$. Alternatively, we can also represent the OC and TE structures by specifically including the number of hydrogen sublattices in the chemical formulae so that they become $PdH_x$ and $PdH_{x/2}H_{x/2}$ respectively. The use of $PdH_{x/2}H_{x/2}$ implies that hydrogen atoms randomly distribute between the two sublattices for the TE structure. Note that if hydrogen atoms fully occupy one of the tetrahedral sublattice, it forms a zinc-blende PdH crystal, $PdH_1H_0$.

In a crystal, an atom interacts with its neighbors that appear at discrete distances. These neighbors can be grouped into neighbor shells (e.g., the nearest neighbor shell, the second neighbor shell, etc.). Assume that within the cutoff distance of the interaction, there are up to $n$ neighbor shells. For convenience, we can view the lattice constant as uniquely determined by the lattice spacing, which is defined as the nearest neighbor distance in any of the f.c.c. sublattices. Suppose that in a structure $\eta$ ($\eta =$ OC, TE), a central atom of species $\mu$ has $z_i(\eta, \mu, \nu)$ atoms in the $i$th neighbor shell

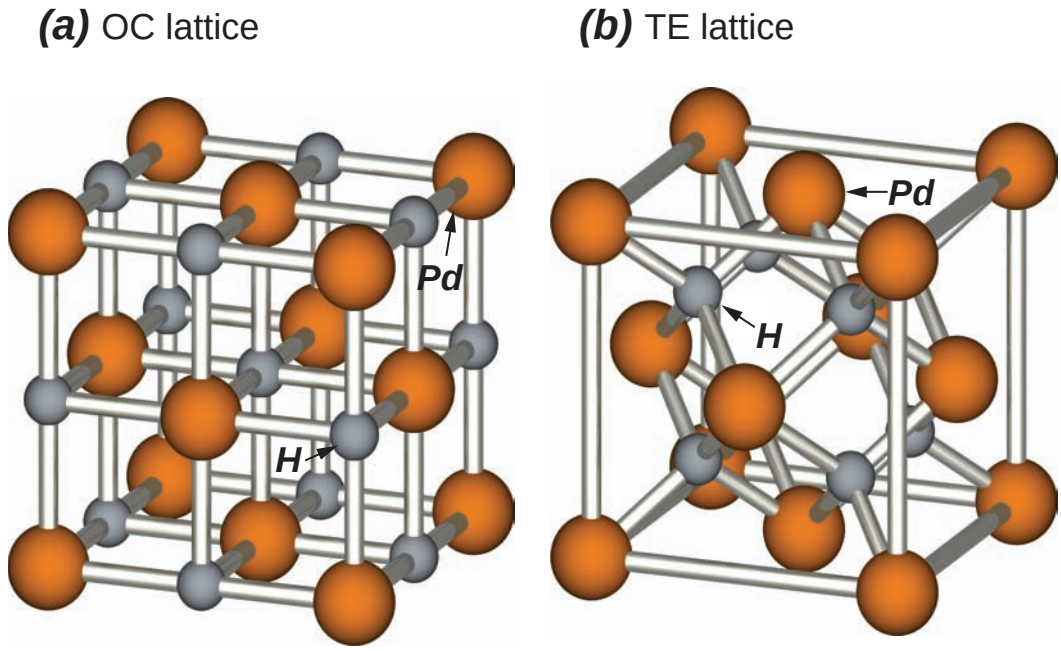



**Figure 5.1.** Two $PdH_x$ structures with (a) hydrogen at octahedral sites and (b) hydrogen at tetrahedral sites.

of species $\nu$. Obviously, $z_i$ is a function of $\eta$, $\mu$, and $\nu$ and can be pre-determined. The corresponding relative distance to the $i$th neighbor shell is $\gamma_i(\eta, \mu, \nu)$, where the relative distance means that the distance $r_i(\eta, \mu, \nu)$ is scaled by the lattice spacing $r_1$, $\gamma_i(\eta, \mu, \nu) = r_i(\eta, \mu, \nu)/r_1$, so that $\gamma_i$ is a pre-determined structure parameter and is independent of the lattice constant. Here the lattice spacing can be represented by $r_1$ because by definition, it is independent of the structure $\eta$ and species $\mu$ and $\nu$. With these concepts, the cohesive energy of the $\eta$ phase of the solid solution $Pd_yH_x$ can be written as

$$E_{c,\eta} = \frac{1}{(x+y)} \left\{ \begin{array}{l} y \cdot F_{Pd}(\rho_{Pd}) + x \cdot F_H(\rho_H) + \\ \frac{1}{2} \sum_{i=1}^{n_{Pd,Pd}} [y \cdot z_i(\eta, Pd, Pd) \cdot y \cdot \phi_{PdPd}(\gamma_i(\eta, Pd, Pd) \cdot r_1)] + \\ \frac{1}{2} \sum_{i=1}^{n_{H,H}} [x \cdot z_i(\eta, H, H) \cdot x \cdot \phi_{HH}(\gamma_i(\eta, H, H) \cdot r_1)] + \\ \sum_{i=1}^{n_{Pd,H}} [y \cdot z_i(\eta, Pd, H) \cdot x \cdot \phi_{PdH}(\gamma_i(\eta, Pd, H) \cdot r_1)] \end{array} \right. \tag{5.24}$$

where $n_{\mu,\nu}$ is total number of $\nu$ neighbor shells surrounding an $\mu$ atom. The electron density at a site of atom $\mu$ can be expressed as

$$\begin{aligned} \rho_\mu = & \sum_{i=1}^{n_{\mu,Pd}} [z_i(\eta, \mu, Pd) \cdot y \cdot \rho^a_{Pd}(\gamma_i(\eta, \mu, Pd) \cdot r_1)] + \\ & \sum_{i=1}^{n_{\mu,H}} [z_i(\eta, \mu, H) \cdot x \cdot \rho^a_H(\gamma_i(\eta, \mu, H) \cdot r_1)] \end{aligned} \tag{5.25}$$

With electron density defined by Eq. 5.25, Eq. 5.24 expresses the cohesive energy as a function of lattice spacing $r_1$ and fitting parameters. Because the expression does not

require calculations for each atom, it can be calculated efficiently. During potential parameterization, the cohesive energy at the equilibrium lattice constant (or lattice spacing) can be conveniently achieved by optimizing the pressure to zero. Pressure can be calculated as

$$P = -\frac{\partial E_c}{\partial V} = -\frac{r_1}{3V}\frac{\partial E_c}{\partial r_1} \tag{5.26}$$

where V is volume per atom.

### 5.4.3 Bulk modulus B, and Elastic Constants C′ and $C_{44}$

Cubic crystals have three independent elastic constants $C_{11}$, $C_{12}$, and $C_{44}$. Alternatively, the three independent elastic constants can be represented by bulk modulus B = ($C_{11}$ + 2$C_{12}$)/3, shear moduli C′ = ($C_{11}$ - $C_{12}$)/2 and $C_{44}$. Our approach uses analytical equations in Mathematica to fit material properties. Analytical equations for elastic constants can be easily derived.

The bulk modulus is simply calculated using Eq. 5.24 as

$$B = V\frac{\partial^2 E_c}{\partial V^2} = \frac{r_1^2}{9V}\frac{\partial^2 E_c}{\partial r_1^2} \tag{5.27}$$

The elastic constant $C_{ij}$ is defined as

$$C_{ij} = \frac{1}{V}\frac{\partial^2 E_c}{\partial \epsilon_i \partial \epsilon_j} \tag{5.28}$$

where $\epsilon_i$ and $\epsilon_j$ ($i$, $j$ = 1, 2, ..., 6) are strains represented by the contracted notation.

To calculate the elastic constant $C_{ij}$, the cohesive energy needs to be expressed as a function of $\epsilon_i$ (i = 1, 2, ..., 6). Assuming that a lattice vector from atom $i$ to atom $j$ in an equilibrium crystal is represented by three components $\Delta x_{ij,0}$, $\Delta y_{ij,0}$, and $\Delta z_{ij,0}$, the three components become $\Delta x_{ij}$, $\Delta y_{ij}$, and $\Delta z_{ij}$ after a $\epsilon_1$, $\epsilon_2$, ..., $\epsilon_6$ strain operation. This strain satisfies:

$$\begin{aligned}
\Delta x_{ij} &= \Delta x_{ij,0}\,(1+\epsilon_1) + 0.5\Delta y_{ij,0}\cdot\epsilon_6 + 0.5\Delta z_{ij,0}\cdot\epsilon_5 \\
\Delta y_{ij} &= 0.5\Delta x_{ij,0}\cdot\epsilon_6 + \Delta y_{ij,0}\,(1+\epsilon_2) + 0.5\Delta z_{ij,0}\cdot\epsilon_4 \\
\Delta z_{ij} &= 0.5\Delta x_{ij,0}\cdot\epsilon_5 + 0.5\Delta y_{ij,0}\cdot\epsilon_4 + \Delta z_{ij,0}\,(1+\epsilon_3)
\end{aligned} \tag{5.29}$$

The distance between atoms $i$ and $j$ is

$$r_{ij} = \sqrt{\Delta x_{ij}^2 + \Delta y_{ij}^2 + \Delta z_{ij}^2} \tag{5.30}$$

Substituting Eqs. 5.29 and 5.30 into Eqs. 5.1 and 5.2, the cohesive energy can be expressed as a function of strain and fitting parameters at the known lattice spacing.

It should be noted that estimates of elastic constants must be made using lattice deformations during which atoms in the same neighboring shell may not always have the same distance to the center atom (except hydrostatic strain). As a result, Eqs. 5.1 and 5.2 instead of Eqs. 5.24 and 5.25 were used in Eq. 5.28 to calculate elastic constants.

## 5.5 Fitting Procedures

The goal is to fit the EAM potential to the desired lattice constant, heat of mixing, bulk modulus, and other elastic constants ($C'$, $C_{44}$) of selected important structures as well as the relative energy difference between OC and TE phases. Four OC crystals $PdH_{0.250}$, $PdH_{0.500}$, $PdH_{0.750}$, and $PdH_{1.000}$, two TE crystals $PdH_{0.250}H_{0.250}$ and $PdH_{0.375}H_{0.375}$, and one f.c.c. hydrogen crystal $Pd_0H$ were considered. The desired properties of the TE structures were simply chosen to ensure that they are less stable than the corresponding (i.e., the same composition) OC structures. The desired properties of hydrogen were initially taken as the ones obtained from density function theory (DFT) calculations, and then adjusted during the iterations of the fitting. The desired bulk modulus and elastic constants $C'$ and $C_{44}$ for the OC structures were taken from experiments performed by Schwarz *et al.* .[27] The dependency of these elastic constants on hydrogen/deuterium/tritium content is depicted in Figure 5.2 and interpolated values are listed in Tables 5.1 and 5.2. The desired lattice constant and heat of mixing of the OC structures were integrated from both DFT calculations and experiments. First, the energy as a function of lattice spacing was determined from DFT calculations for the four OC structures $PdH_{0.250}$, $PdH_{0.500}$, $PdH_{0.750}$, and $PdH_{1.000}$.[78] As it is well known that cohesive energies determined from DFT calculations systematically deviate from experimental measurements, the DFT values were shifted to best match the experimental cohesive energies. The experimental cohesive energy of palladium element is known to be -3.91 eV. An experimental cohesive energy for the compound $PdH_{0.5}$ can be derived using the following reactions:

$$Pd_{(g)} \rightarrow Pd_{(s)} \tag{5.31}$$

$$Pd_{(s)} + \frac{1}{4}H_{2(g)} \rightarrow PdH_{0.5(s)} \tag{5.32}$$

and

$$\frac{1}{2}H_{(g)} \rightarrow \frac{1}{4}H_{2(g)} \tag{5.33}$$

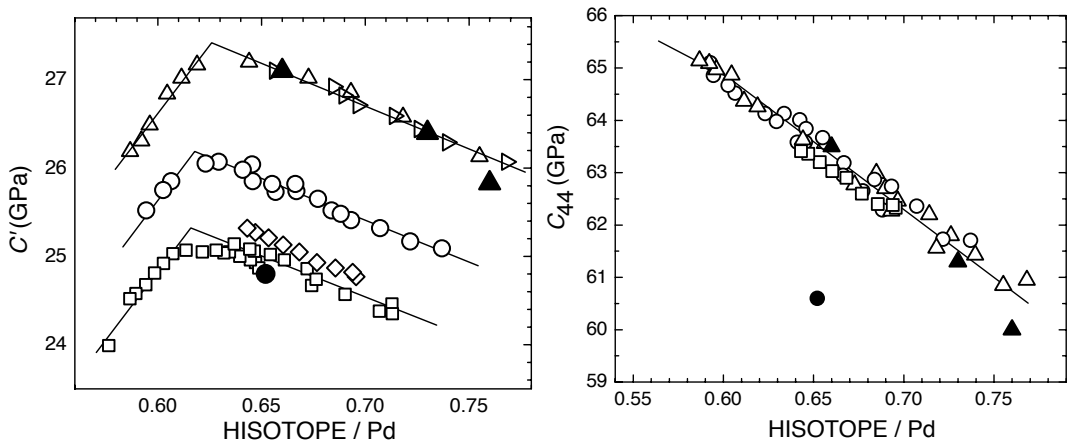
Here the subscripts $(g)$ and $(s)$ represent solid and gas states of the materials respectively. The energy change of Eq. 5.31 is equivalent to the palladium cohesive energy. The energy changes of Eqs. 5.32 and 5.33 have been measured experimentally.[79, 80] By summing Eqs. 5.31, 5.32 and 5.33, an experimental cohesive energy of the $PdH_{0.5}$ compound was determined as -3.4877 eV/atom. The two experimental cohesive energies for Pd and $PdH_{0.5}$ allow us to determine the respective energy shifts per palladium

**Table 5.1.** Interpolated data from Figure 5 of [27] for $C'$ as a function of H-isotope concentration.

| H/Pd | $C'$ (GPa) | D/Pd | $C'$ (GPa) | T/Pd | $C'$ (GPa) |
|---|---|---|---|---|---|
| 0.58667 | 26.18140 | 0.59456 | 25.52093 | 0.57646 | 23.99070 |
| 0.59203 | 26.30698 | 0.60249 | 25.74419 | 0.58679 | 24.51628 |
| 0.59619 | 26.48372 | 0.60646 | 25.84651 | 0.58956 | 24.58605 |
| 0.60453 | 26.84651 | | | 0.59432 | 24.68372 |
| 0.61147 | 27.02326 | 0.62310 | 26.05116 | 0.59868 | 24.81395 |
| 0.61900 | 27.16744 | 0.62944 | 26.06977 | 0.60285 | 24.92093 |
| | | 0.64112 | 25.98140 | 0.60721 | 25.02791 |
| 0.64415 | 27.20000 | 0.64587 | 26.04186 | | |
| 0.65682 | 27.09767 | 0.64567 | 25.85581 | 0.61375 | 25.06977 |
| 0.67266 | 27.01860 | 0.65537 | 25.82326 | 0.62147 | 25.05116 |
| 0.68513 | 26.92093 | 0.65675 | 25.73023 | 0.62820 | 25.06512 |
| 0.68988 | 26.82326 | 0.66646 | 25.82326 | 0.63176 | 25.04651 |
| 0.69324 | 26.86047 | 0.66665 | 25.73023 | 0.63473 | 25.05581 |
| 0.69660 | 26.71628 | 0.67714 | 25.65116 | | |
| 0.71422 | 26.59070 | 0.68347 | 25.52558 | 0.63692 | 25.13953 |
| 0.71818 | 26.58140 | 0.68822 | 25.48372 | 0.63948 | 25.00930 |
| 0.72629 | 26.44186 | 0.69297 | 25.41395 | 0.64424 | 25.07907 |
| 0.73995 | 26.29302 | 0.70722 | 25.32093 | 0.64443 | 24.96279 |
| 0.75499 | 26.13488 | 0.72167 | 25.16744 | 0.64681 | 25.05581 |
| 0.76825 | 26.07442 | 0.73672 | 25.08837 | 0.64720 | 24.94419 |
| | | | | 0.64898 | 24.86977 |
| | | | | 0.65434 | 25.01860 |
| | | | | 0.66126 | 24.95814 |
| | | | | 0.67195 | 24.86047 |
| | | | | 0.67412 | 24.66512 |
| | | | | 0.67630 | 24.73953 |
| | | | | 0.69015 | 24.57209 |
| | | | | 0.70678 | 24.38605 |
| | | | | 0.71272 | 24.46047 |
| | | | | 0.71292 | 24.34884 |

**Table 5.2.** Interpolated data from Figure 6 of [27] for $C_{44}$ as a function of H-isotope concentration.

| H/Pd | $C_{44}$ (GPa) | D/Pd | $C_{44}$ (GPa) | T/Pd | $C_{44}$ (GPa) |
|---|---|---|---|---|---|
| 0.58679 | 65.14296 | 0.59306 | 65.09750 | 0.64316 | 63.42057 |
| 0.59235 | 65.10730 | 0.59448 | 64.86958 | 0.64726 | 63.36599 |
| 0.59606 | 64.97181 | 0.60247 | 64.69049 | 0.65326 | 63.21316 |
| 0.60451 | 64.87911 | 0.60671 | 64.52769 | 0.66020 | 63.03851 |
| 0.61131 | 64.36567 | 0.62296 | 64.14103 | 0.66809 | 62.90751 |
| 0.61894 | 64.26583 | 0.62927 | 63.98821 | 0.67629 | 62.61278 |
| 0.64408 | 63.62403 | 0.63369 | 64.14103 | 0.68544 | 62.41629 |
| 0.65583 | 63.53132 | 0.64095 | 63.58431 | 0.69364 | 62.39446 |
| 0.67273 | 62.76829 | 0.64190 | 64.01004 | 0.69522 | 62.33987 |
| 0.68468 | 62.99649 | 0.64568 | 63.84630 | | |
| 0.68942 | 62.70411 | 0.64537 | 63.58431 | | |
| 0.69334 | 62.26911 | 0.65483 | 63.67164 | | |
| 0.69664 | 62.46165 | 0.66588 | 62.96209 | | |
| 0.71395 | 62.20493 | 0.66682 | 63.19133 | | |
| 0.71807 | 61.57026 | 0.67692 | 62.65644 | | |
| 0.72611 | 61.81272 | 0.68355 | 62.86385 | | |
| 0.73930 | 61.43477 | 0.68797 | 62.27438 | | |
| 0.75475 | 60.85002 | 0.69301 | 62.74377 | | |
| 0.76794 | 60.95698 | 0.70690 | 62.35079 | | |
| | | 0.72141 | 61.73949 | | |
| | | 0.73656 | 61.71766 | | |




(a) (b)

**Figure 5.2.** Shear moduli (a) $C$' and (b) $C_{44}$ measured by Schwarz *et al.* [27] for the $PdH_x$ (△), $PdD_x$ (◯) and $PdT_x$ (□) alloy systems.

**Table 5.3.** Adjusted DFT properties.

| Properties\structures | $PdH_{0.00}$ | $PdH_{0.25}$ | $PdH_{0.50}$ | $PdH_{0.75}$ | $PdH_{1.00}$ |
|---|---|---|---|---|---|
| lattice constant a (Å) | 3.960 | 4.020 | 4.077 | 4.109 | 4.147 |
| cohesive energy $E_c$ (eV/atom) | −3.910 | −3.647 | −3.488 | −3.363 | −3.261 |

and per hydrogen atom. These in turn allow us to adjust the cohesive energy of a structure based on its composition. The adjusted cohesive energies, along with the lattice constants, are shown in Table 5.3 for the five important OC structures.

An objective function was defined as a weighted sum of the square deviation of predicted properties from the target properties for all the structures being fitted. These properties include the heat of mixing, relative energy difference, pressure, bulk modulus, and elastic constants $C'$ and $C_{44}$. The target values for the pressure are set to zero to enforce equilibrium lattice constants. Some properties, such as those of the TE structures, do not require an exact match. We then use options "$>$" or "$<$" instead of "$=$". Only when the predicted properties violate these options when compared to their target values will the corresponding square deviation be summed into the objective function.

It is important to implement a cutoff distance $r_c$ for both atomic electron density

**Table 5.4.** EAM parameters.

| | | | | | |
|---|---|---|---|---|---|
| Embedding energy | $i$ | $c_i$ | $d_i$ | $a_i$ | $b_i$ |
| | $H$ | 0.000197047 | 1.18860 | 9.99780 | 60.0155 |
| Electron density | $i$ | $C_i$ | $\delta_i$ | – | – |
| | $H$ | 11.0025 | 1.30927 | – | – |
| Pair energy | $ij$ | $D_{ij}$ | $\alpha_{ij}$ | $\beta_{ij}$ | $r_{0,ij}$ |
| | $PdH$ | 0.2494540 | 4.82613 | 2.13158 | 1.50964 |
| | $HH$ | 0.0661496 | 3.67263 | 1.47797 | 2.51980 |
| Cut off | $ij$ | $r_{c,ij}$ | $r_{s,ij}$ | – | – |
| | $PdH$ | 4.9500000 | 0.30000 | – | – |
| | $HH$ | 5.3500000 | 0.30000 | – | – |

and pair energy functions. Two steps are used. First, we need to ensure that both functions naturally (i.e., by themselves) decay to small values when r gets close to $r_c$, say, r = $r_c$ - $r_s$, where $r_s$ is a small distance. This can be achieved by using the constraints

$$\frac{\phi\left(r_c - r_s\right)}{\phi\left(r_{1,e}\right)} < \epsilon \tag{5.34}$$

and

$$\frac{\rho^a\left(r_c - r_s\right)}{\rho^a\left(r_{1,e}\right)} < \epsilon \tag{5.35}$$

during optimization, where $\epsilon$ is a small number, and $r_{1,e}$ is the equilibrium lattice spacing of the equilibrium structure (a nominal atomic separation that is highly likely to be seen during simulations). The second step is to ensure that both pairwise functions indeed drop to zero at r = $r_c$. This can be done by multiplying them with a cutoff function $\frac{1}{2} + \frac{1}{2}\cos\frac{\pi(r-r_c+r_s)}{r_s}$ when r > $r_c$ - $r_s$.

With proper bounds for all parameters, optimization is done using four Mathematica[81] built-in methods: differential evolution, simulated annealing, default, and Nelder Mead algorithm. The use of four different methods increases the probability of finding a global minimum as the four methods usually produce different results (unless the initial parameters that are used for all methods are already close to the global minimum point). Iterative fitting is carried out by varying fitting conditions (e.g., cutoff distances, weights, target values, constraints, use of “>” or “<” fitting options instead of “=” option or vice versa etc.) until satisfactory results are obtained. The fitted parameters obtained through this process are listed in Table 5.4.

## 5.6 Characteristics of the Potential

### 5.6.1 EAM Functions

To characterize the fitted potential, we first examine all the EAM functions in Fig. 5.3. Smooth curves can be seen for all the functions. Due to the application of the two normalization approaches, Fig. 5.3(a) shows well-behaved embedding energy functions for palladium and hydrogen that reach a minimum at comparable electron densities. The Pd-Pd and H-H pair energies shown in Fig. 5.3(c) reach a minimum at the atomic spacings that are proportional to their lattice constants. The magnitudes of the pair energies are also comparable. The equilibrium separation of the Pd-H pair energy is shorter, and the corresponding binding energy is higher. Finally, the decay of the electron density functions shown in Fig. 5.3(b) appears reasonable and similar for both palladium and hydrogen over the atomic separation distance range commonly encountered in simulations [i.e., $\geq 2$ Å for palladium and $\geq 1.5$ Å for hydrogen, see Fig. 5.3(c)].

Based upon this potential, molecular statics simulations were used to minimize the energy of various OC structures as a function of composition using a cubic computational system containing 6×6×6 f.c.c. unit cells (corresponding to 864 palladium atoms) or more. The resulting crystals were used to evaluate the lattice constant, cohesive energy, and elastic constants $C_{11}$, $C_{12}$, and $C_{44}$ are shown in Table 5.5. It should be noted that the Pd elastic constants listed in Table 5.5 are slightly different from the values in reference.[10] This is because the tabular Pd potential functions[37] do not have smooth second derivatives, and we therefore approximated them with high order polynomial functions. Our polynomial functions have almost the same values and the same first derivatives to the original tabular functions, but have slightly different second derivatives. This causes differences in elastic constant predictions, but not the cohesive energy or the lattice constant.

### 5.6.2 Energy as a Function of Lattice Spacing

We further explore the behavior of the potential on predicting the system energy as the crystal is compressed or stretched. The energy per atom as a function of lattice spacing is plotted in Fig. 5.4 for two representative phases, $PdH_{0.25}$ and $PdH_{0.50}$. Again, smooth curves are seen in Fig. 5.4 which indicate that as the hydrogen composition increases, the lattice constant increases, and the magnitude of cohesive energy becomes lower, in good agreement with the trend shown in Table 5.5.

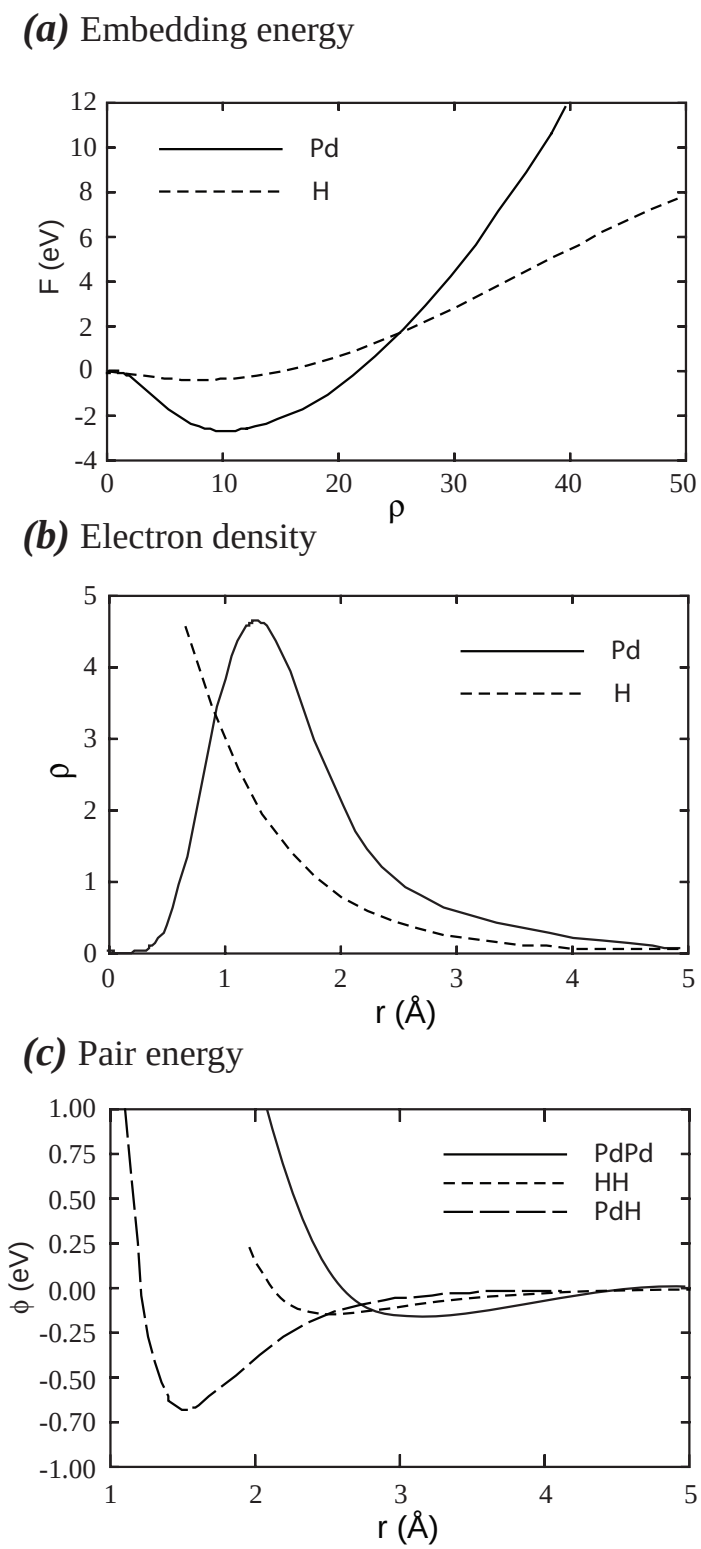



**Figure 5.3.** EAM function curves for (a) embedding energy; (b) electron density; and (c) pair energy.

**Table 5.5.** Predicted values of lattice constant a (Å), cohesive energy $E_c$ (eV/atom), and elastic constants $C_{11}$, $C_{12}$, and $C_{44}$ (GPa).

| Structures | $a$ | $E_c$ | $C_{11}$ | $C_{12}$ | $C_{44}$ |
|---|---|---|---|---|---|
| $Pd_1H_{0.000}$ | 3.885 | −3.910 | 245.8 | 199.8 | 54.7 |
| $Pd_1H_{0.087}$ | 3.923 | −3.770 | 226.0 | 186.5 | 44.7 |
| $Pd_1H_{0.185}$ | 3.965 | −3.638 | 207.7 | 173.4 | 38.9 |
| $Pd_1H_{0.292}$ | 4.010 | −3.520 | 194.1 | 162.7 | 32.7 |
| $Pd_1H_{0.378}$ | 4.045 | −3.439 | 186.0 | 156.2 | 28.9 |
| $Pd_1H_{0.476}$ | 4.083 | −3.361 | 181.4 | 152.0 | 26.7 |
| $Pd_1H_{0.558}$ | 4.113 | −3.303 | 183.9 | 152.5 | 27.5 |
| $Pd_1H_{0.666}$ | 4.148 | −3.238 | 208.3 | 163.1 | 33.5 |
| $Pd_1H_{0.784}$ | 4.180 | −3.176 | 241.5 | 176.7 | 36.4 |
| $Pd_1H_{0.885}$ | 4.202 | −3.129 | 245.9 | 181.9 | 33.2 |
| $Pd_1H_{1.000}$ | 4.225 | −3.081 | 241.7 | 190.3 | 25.5 |

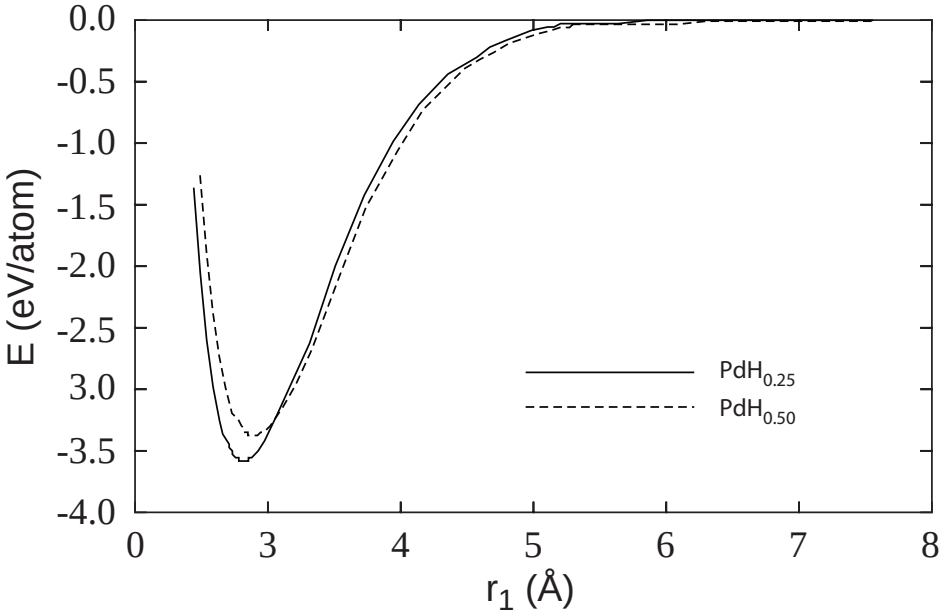



**Figure 5.4.** Energy as a function of lattice spacing for OC lattices.

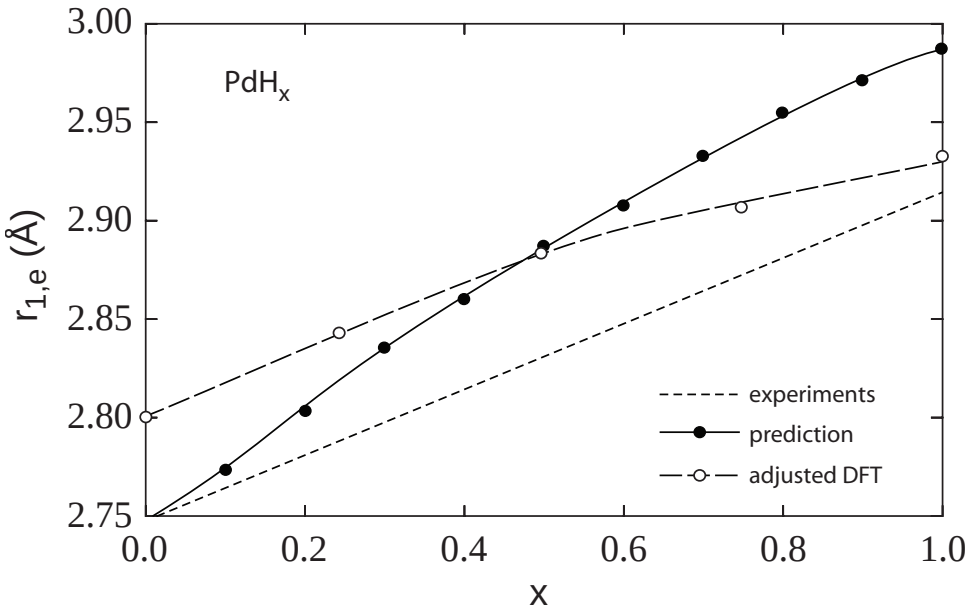



**Figure 5.5.** Lattice spacing as a function of composition for OC lattices.

### 5.6.3 Equilibrium Lattice Spacing

The equilibrium lattice constant as a function of composition is an important property as it causes stress when adjacent regions have different compositions or even different phases. To characterize the fitted potential, the predicted equilibrium lattice spacing as a function of composition is shown as filled circles in Fig. 5.5 for the OC $PdH_x$ structure. For comparison, the DFT data is shown as open circles, and the experimental values are shown as the short dashed line. To guide the eye, solid and long dashed lines are used to connect the data points. It can be seen that the trend of an increasing equilibrium lattice spacing with increasing hydrogen composition is predicted by our potential. The predicted equilibrium lattice spacing lies between experimental and DFT calculated values at small compositions, but is larger than both experimental and DFT calculated values at larger compositions. This difference arose as a trade-off to better fit the phase miscibility gap.

### 5.6.4 Cohesive energy

The shape of the cohesive energy as a function of hydrogen composition relates directly to the phase miscibility gap. In addition, the equilibrium OC structures must have lower cohesive energies than the TE structures over the entire composition range. Predicted cohesive energies as functions of composition $x$ are shown in Fig. 5.6 using

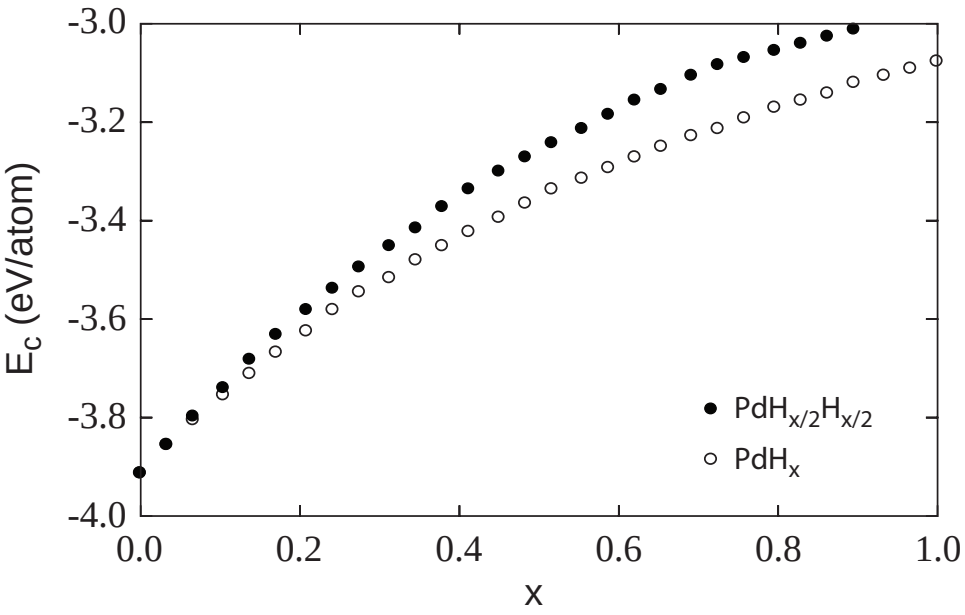



**Figure 5.6.** Cohesive energy as a function of composition for OC and TE lattices.

the filled circles for the TE structure and the open circles for the OC structures. Here cohesive energies of the OC structures were the minimum energy obtained from molecular statics simulations, listed in Table 5.5. The same approach cannot be used for unstable TE structures as they will transform to OC structures in simulations. As a result, the cohesive energies of TE structures were calculated using Eq. 5.24 at the equilibrium lattice spacing $r_{1,e}$. It can be seen from Fig. 5.6 that the potential correctly predicts more a stable OC structure than a TE structure over the entire composition range $0 \leq x \leq 1$.

### 5.6.5 Gibbs Free Energy of Mixing

To verify the existence of the phase miscibility gap and to estimate the compositions of the $\alpha$ and $\beta$ phases when they are in equilibrium, the cohesive energy data obtained from molecular statics simulations were used in Eq. 5.16 to calculate the Gibbs free energy of mixing as a function of mole fraction for the OC structure. The results are plotted in Fig. 5.7 at two temperatures, 300 K and 500 K. Fig. 5.7 clearly shows that the system splits into two co-existing phases. At the low temperature of 300 K, the equilibrium mole fractions for the $\alpha$ and $\beta$ phases are $X = 0$ and 0.45 respectively, corresponding to $x$ of about 0 and 0.82. While the potential correctly predicts the phase separation, the equilibrium composition for the $\alpha$ and $\beta$ phases are somewhat underestimated and overestimated, respectively. As mentioned earlier, it has been

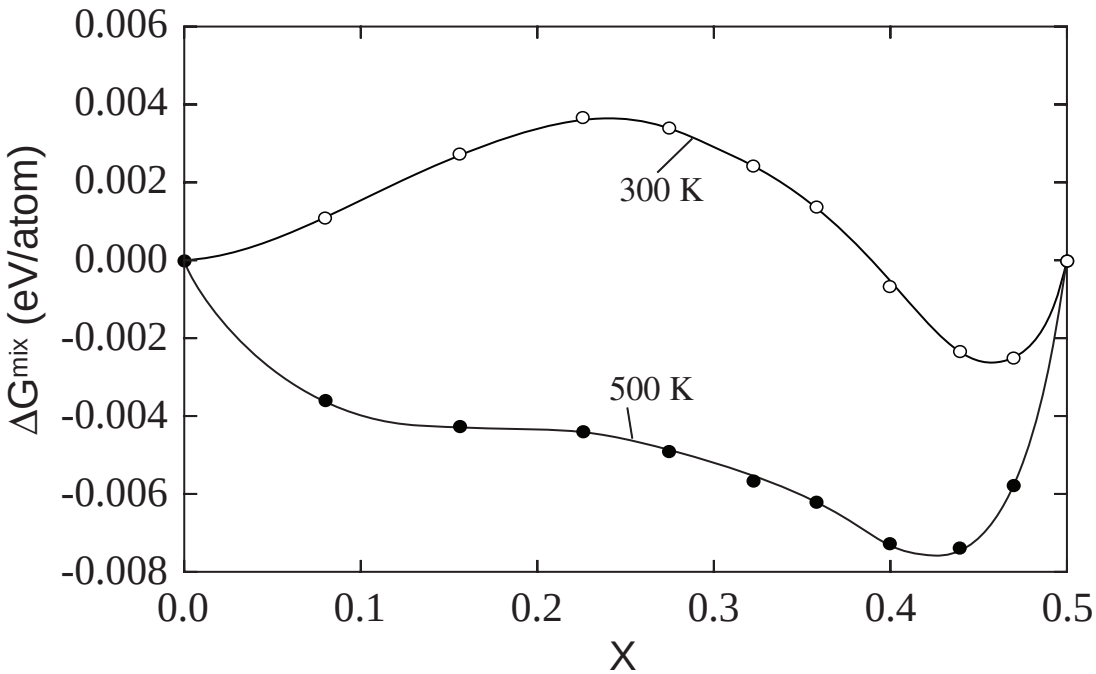



**Figure 5.7.** Gibbs free energy of mixing as a function of mole fraction for the OC structure.

experimentally determined that phase boundaries at room temperature are $x = 0.03$ and 0.6[15]. It should be noted that such a discrepancy was also characteristic of the "lost" potential by Wolf et al[16]. They noted $\alpha$ and $\beta$ compositions of $x = 0.1385$ and 0.8362. Our potential is an improvement in this regard. At the high temperature of 500 K, the mole fractions for the $\alpha$ and $\beta$ phases become around $X = 0.1$ and 0.4 respectively, corresponding to $x$ of about 0.11 and 0.67.

It is interesting to note that the free energy curves shown in Fig. 5.7 do not match the theoretical curves expected from the potential parameter fitting program FitEAM (see Appendix A). In the FitEAM program, a random distribution of hydrogen is assumed for a given concentration $x$ or mole fraction $X$. However, our atomistic system contains a distribution which may vary somewhat from this theoretical one. More importantly, using MD to calculate the heat of mixing curve permits the atoms to relax from their starting positions, thereby achieving a lower energy configuration. This relaxation cannot be done in the fitting program. Hence, the MD-generated curves shown in Fig. 5.7 are accurate with regard to the miscibility behavior of the simulated material.

### 5.6.6 Elastic Constants

The minimized energy OC configurations were used to calculate the elastic constants B, C′, and $C_{44}$ (B, C′, and $C_{44}$ can be converted from $C_{11}$, $C_{12}$, and $C_{44}$, Table 5.5). The results are shown in Fig. 5.8 as functions of composition.

It can be seen that the general trend of how elastic constants vary with composition matches well with the experiments by Schwarz et al.[27] Also, the quantitative agreement is very good, especially considering that these experimental values were only weakly used in the fitting process. Several specific features can be noticed: The value of the bulk modulus at high concentrations of hydrogen, i.e. beyond the $\beta$ phase boundary composition, is only slightly lower than the value at the dilute limit, and is essentially flat within the high concentration range $0.82 \leq x \leq 1$. C′, shown in Fig. 5.8(b), exhibits a peak value at the equilibrium composition of the $\beta$ phase. Our computationally-determined curve even has a shape similar to the one theorized by Schwarz et al. and shown in Fig. 9 of their paper[27]. The value of $C_{44}$, while differing somewhat from the bilinear form supposed by Schwarz et al., does show a decreasing trend over the high hydrogen concentration range of $0.82 \leq x \leq 1$. Also, it does appear that a line connecting the $\beta$ concentration point with the $x = 0$ (pure Pd) point would have a lower slope than a line interpolated through the high concentration points.

### 5.6.7 Mechanical Strength

We also performed molecular statics calculations to explore the mechanical behavior of the $PdH_x$ OC structure as a function of composition. We first examined the tensile strength using systems containing 6×6×6 cubic cells in the three coordinate directions (a total of 864 to 1,728 atoms depending on the hydrogen concentration). Periodic boundary conditions were used in all directions, and the system was sequentially stretched in the $y$ direction by a strain increment of 0.00005. For each stretch increment, the energy of the system was minimized and a constant zero pressure condition was applied in the $x$- and $z$- directions to simulate uniaxial stress loading. The virial stress of the relaxed system as a function of stretch was calculated. Two examples, $PdH_{0.1}$ and $PdH_{0.8}$ respectively, are shown in Fig. 5.9. In Fig. 5.9, the applied tensile strain is defined as $\epsilon = (1.00005)^n - 1$ where $n$ is the number of stretch increments applied. If strength is defined as the maximum stress in the stress vs. applied tensile strain curve, the results in Fig. 5.9 indicate that the strength of the material decreases with increasing hydrogen composition. This is indeed shown to be the case in Fig. 5.10, where strength is plotted as a function of composition. Our results indicate that tensile strength decreases monotonically as hydrogen composition is increased.

We next examined shear strength, using systems containing 10×10×10 cubic cells for a total of 4,000 to 8,000 atoms. Periodic boundary conditions are applied in the $x$-

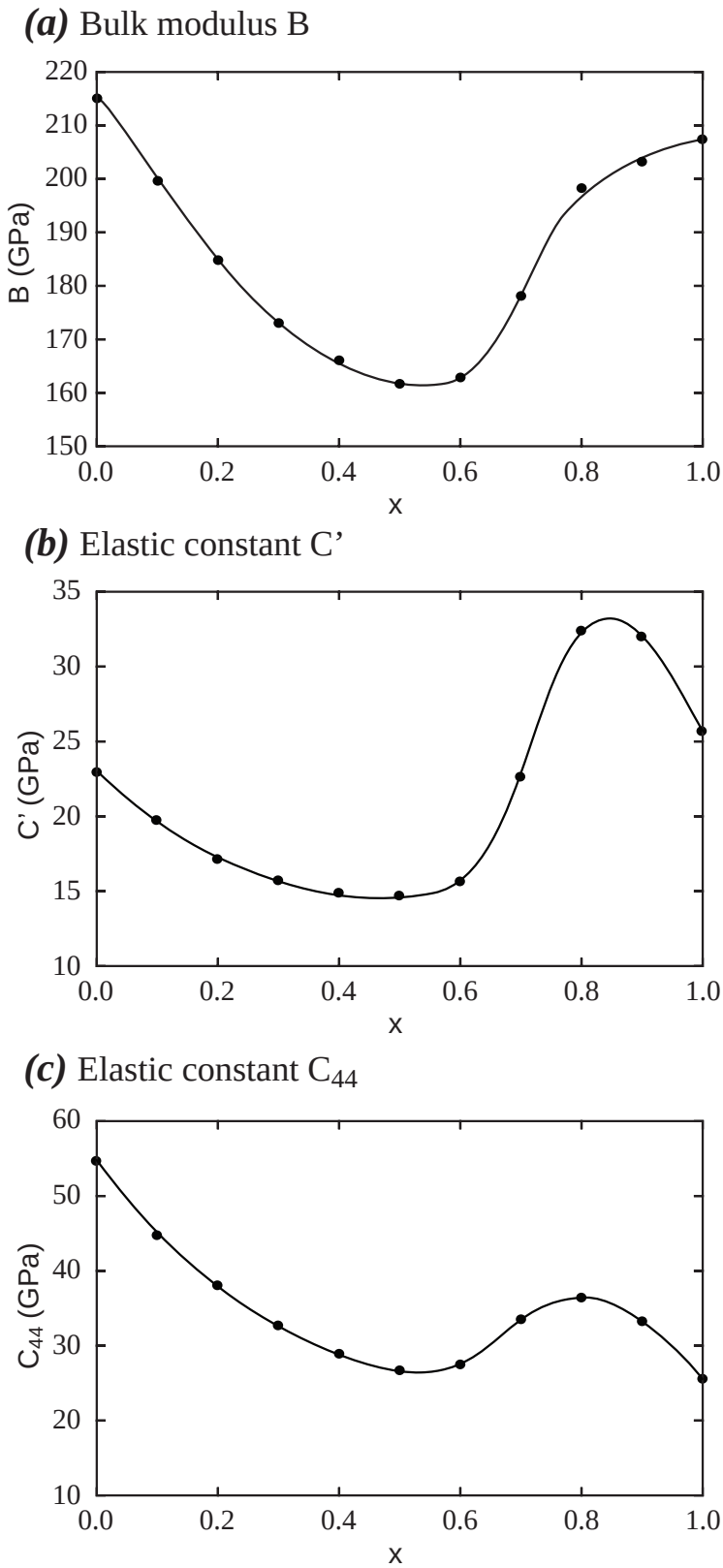



**Figure 5.8.** Elastic constants as a function of composition for OC structure. (a) B; (b) C′; and (c) $C_{44}$. Curved, solid lines are shown as guides for connecting the simulated compositions.

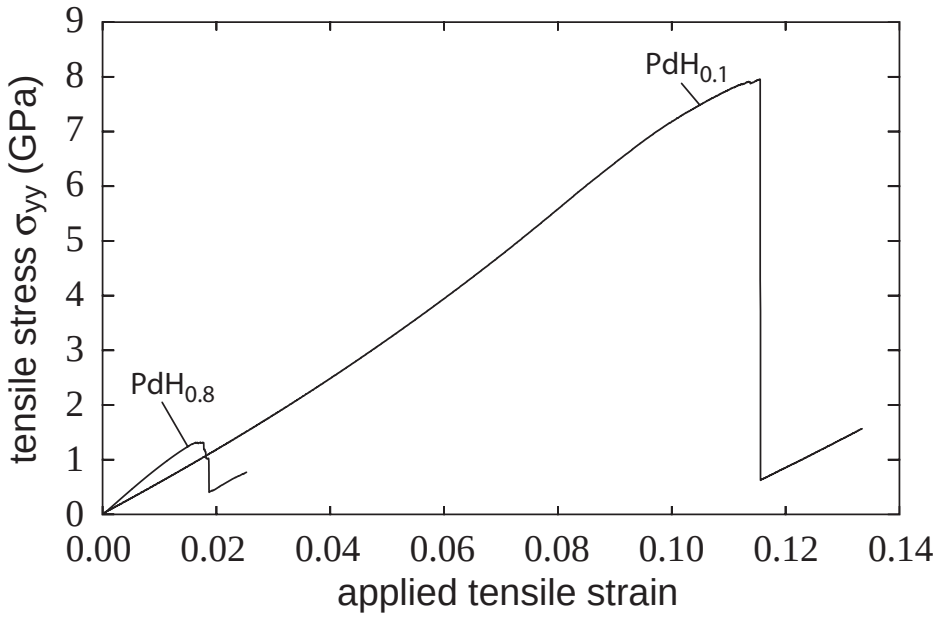



**Figure 5.9.** Tensile stress as a function of stretch.

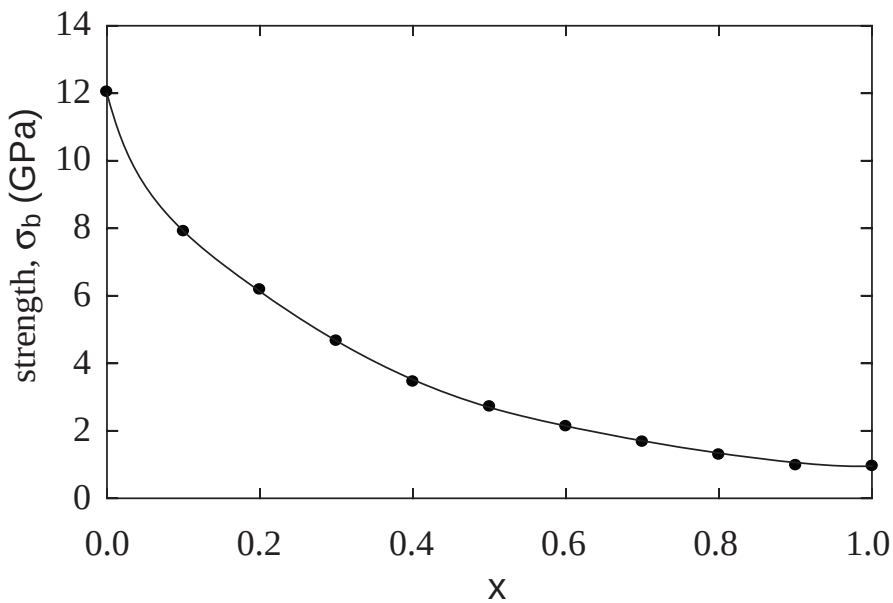



**Figure 5.10.** Tensile strength as a function of composition for OC structures.

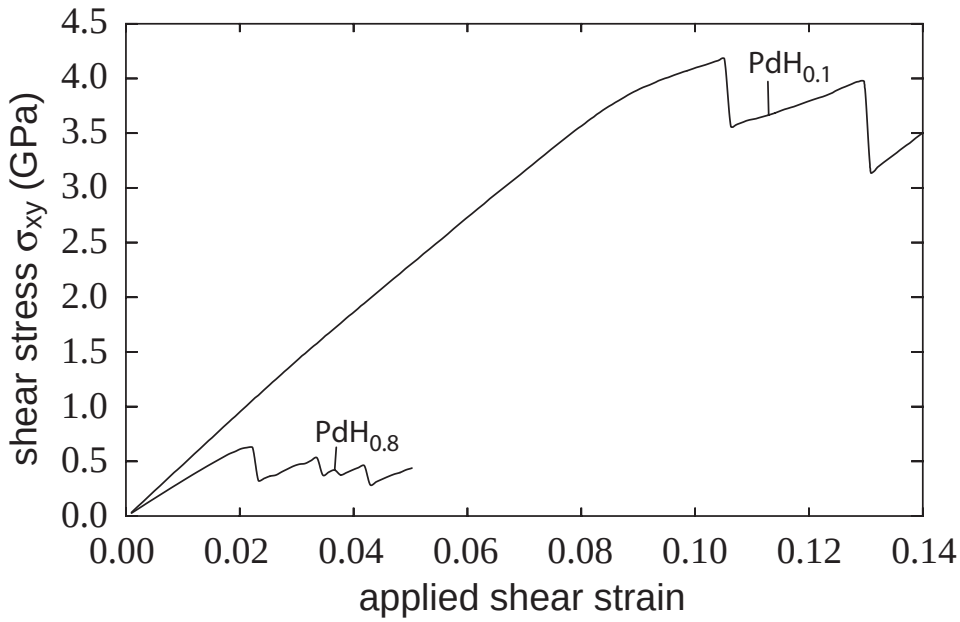



**Figure 5.11.** Shear stress as a function of applied strain.

and $z$- directions, while the top and bottom layers (referred to the $y$- direction) are displaced in the $x$- direction to simulate a simple shear mode of deformation. As the shear is applied, the system is allowed to relax in the $y$- direction, but the $x$- and $z$- dimensions are kept constant. A strain increment of 0.0001 is used. Due to the non-periodicity of the boundaries in the $y$-direction, the stress is not calculated using all the atoms within the system. Rather, a sub-region in the middle of the system with a dimension of $10\times5\times10$ cubic cells is used to estimate the stresses, thereby minimizing the effect of boundary conditions on this material property. Fig. 5.11 shows how this estimate of shear stress varies with the applied shear strain, and Fig. 5.12 shows how shear strength, the maximum value for these shear stress curves, varies as a function of composition. It is observed that for our potential, shear strength also decreases with increasing hydrogen concentration.

## 5.7 Molecular Dynamics Simulations

The simulations discussed above were all carried out under static conditions and relaxed atomic spacing. During MD simulations, atoms are disturbed from their ideal lattice sites. This allows the system to sample many of the configurations that have not been sampled under the static conditions. As a result, statically stable structures may become unstable during MD simulations. We have therefore carried out a series of MD simulations to verify the fidelity of our potential.

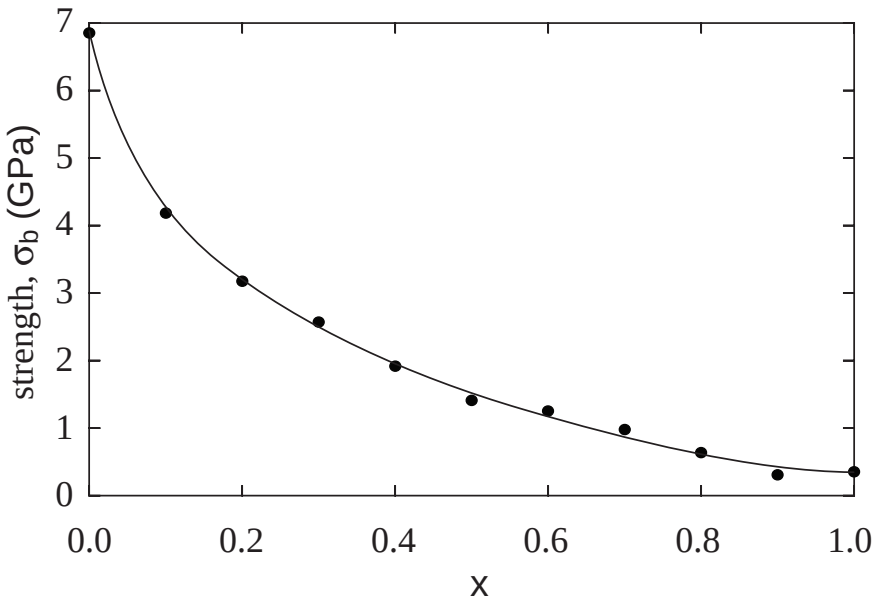



**Figure 5.12.** Shear strength as a function of composition for OC structures.

In the first type of simulations, OC and TE structures with various compositions were annealed for a nanosecond at various temperatures ranging from 200 K to 500 K using the MD simulation approach. All OC structures were found stable, and all TE structures were found to transform to OC structures. As an example, an initial $PdH_{0.392}H_{0.392}$ TE structure containing 6×6×6 cubic cells in the three coordinate directions is shown in Fig. 5.13(a). Molecular dynamics was first used to anneal the structure at 300 K for a nanosecond. Molecular statics was subsequently used to minimize the energy of the system. The atomic configuration after MD+MS simulations is shown in Fig. 5.13(b). It can be seen that many of the hydrogen atoms switched from the tetrahedral sites to the octahedral sites during the MD+MS energy minimization process. The potential hence ensures both phase stability and the correct hydrogen interstitial sites.

The second type of MD tests used the same crystal geometry to examine the diffusion of a single hydrogen atom. We found that when the initial system temperature was initialized according to a Boltzmann energy distribution for all atoms in the system, a classical MD simulation always predicted a hydrogen atom jump rate that was almost the same as its vibrational rate. This occurred in both constant energy and constant temperature simulations that were carried out using time steps ranging from 0.00005 ps to 0.001 ps (small time steps were needed because the hydrogen atom moved very fast as its mass is about 100 times smaller than that of a palladium atom). However, if the initial temperature was only initialized to the palladium atoms the initial velocity of the hydrogen atom was set to zero, or the mass of hydrogen atom was artifically

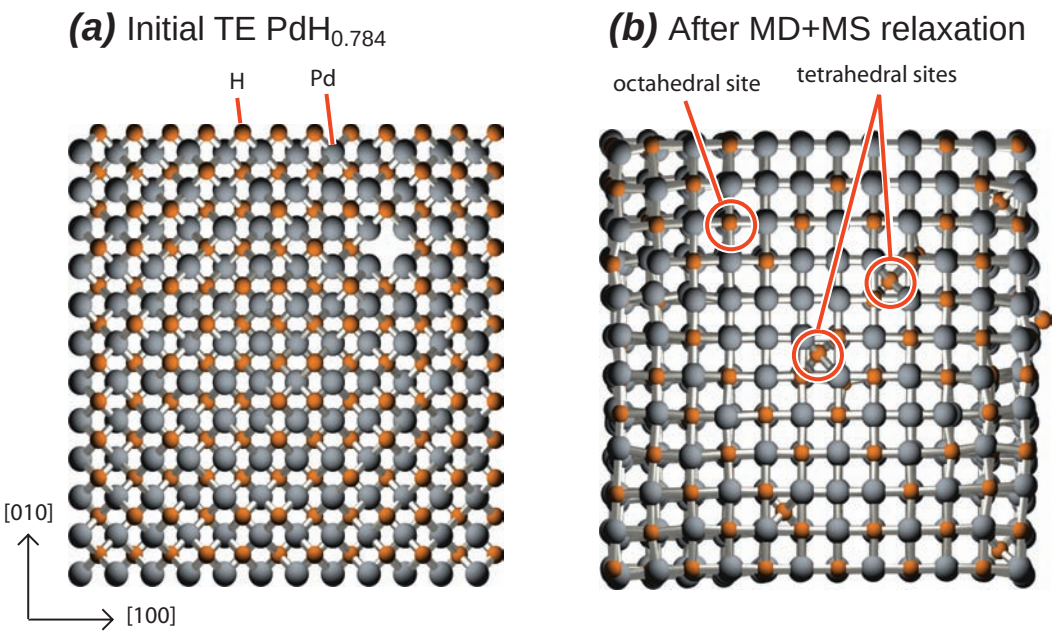



**Figure 5.13.** Change of atomic configurations of a $PdH_{0.784}$ phase due to MD annealing at 300 K for one nanosecond followed by MS energy minimization. (a) The TE structure before MD+MS relaxation; and (b) After MD+MS energy relaxation.

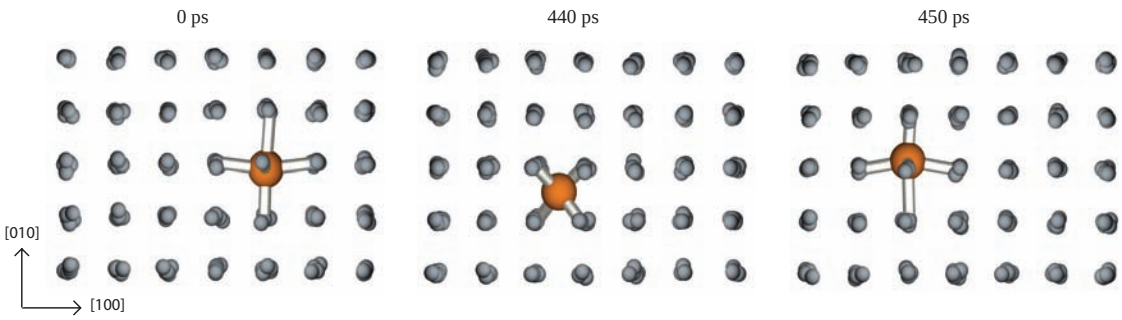



**Figure 5.14.** Snapshots of single hydrogen atom diffusion in a palladium matrix. (a) time 0.000 ps, (b) time 0.043 ps; and (c) time 0.063 ps. The size of the hydrogen atom is magnified for clarity.

increased to a value comparable to that of a palladium atom, then realistic diffusion jump rates were obtained. We noted that because the hydrogen mass was small, little energy transfer occurred during each vibration of the hydrogen atom in the palladium lattice. If the hydrogen acquired enough energy (either due to velocity initialization or the dragging force used for constant temperature simulation) that could overcome the energy barrier for jumps, it maintained its jump capability over a long time, resulting in fast jump rates. It is not clear if the temperature initialization or the temperature control routines used in classical MD simulations are accurate representations of the equilibrium atomistic systems involving particles with vastly different masses. We believe that a realistic simulation requires a constant energy MD simulation over a long time using a very small time step with a zero initial velocity for the hydrogen atom. Since our purpose was to test the potential, a more efficient computational approach was used. We artificially increased the mass of the hydrogen atom to that of a palladium atom. This increase will change the vibrational frequency (for harmonic systems, vibration frequency is inversely proportional to the square root of the mass), but not the activation energy barrier for the jumps. The time evolution of the position of the hydrogen atom obtained from a MD simulation at a temperature of 300 K is shown in 5.14. A hydrogen atom was initially put at an octahedral site Fig. 5.14(a). After 440 ps, the hydrogen was seen to make a first jump to a tetrahedral site, Fig. 5.14(b). After 450 ps, the hydrogen atom was again seen to jump to an octahedral site, Fig. 5.14(c). As a result, jumps occur on the order of 400 ps, via an octahedral → tetrahedral → octahedral path. The fast diffusion is in good agreement with experiments.[82]

## 5.8 Conclusions

Analytical EAM potentials have been developed and integrated with a published EAM potential of palladium to form a Pd-H alloy EAM potential. This potential is suitable for any composition of the Pd-H system. It well predicts the lattice constant, cohesive energy, bulk modulus, and other elastic constants for a variety of structures. It can also predict the phase miscibility gap for the Pd-H system. Direct MD simulations indicate that this potential ensures the phase stability during dynamic simulations and can correctly capture the diffusion mechanism of hydrogen in the palladium lattice. The following approaches were found to be critical to the development: (a) hydrogen and other non-equilibrium phases must be included in the potential parameterization; (b) both the embedding energy and electron density functions must be normalized for palladium and hydrogen in order to create a well-behaved alloy potential; (c) a solid solution model significantly improves the efficiency of the potential parameterization while still possesses sufficient accuracy needed for the potential parameterization; (d) more general embedding and pair energy functions can be used to improve the fitting; and (e) the parameterization must be carried out iteratively with respect to cutoff distance, adjustment of target values, the weights used to define the objective function, the bounds for parameters or other constraints, and fitting options such as "$>$" or "$<$" instead of "$=$".

This page intentionally left blank.

# Chapter 6

# Conclusions and Future Work

## 6.1 Conclusions

This report describes the different paths our group has traversed in an attempt to develop inter-atomic potentials for the Pd-H and Pd-T systems. In Chapter 2, we discovered that while a number of potentials have been published in the scientific literature, they are either unable to capture the complex physics of the miscibility gap characteristic of the Pd-H system, or the published works do not contain the necessary details for implementation. As such we then preceded to use the simple addition of pair functions (Chapter 3) and advanced numerical optimization techniques (Chapter 4) to construct the needed potentials. These efforts showed limited success as the potentials produced did not correctly predict the site occupancy of H atoms and were not able to maintain a stable FCC lattice structure for a $PdH_x$ alloy. Our most recent effort (Chapter 5) to combine analytical expressions that codify these physical attributes with numerical optimization has proven to yield acceptable results. This method produces results comparable to, if not an improvement upon, the "lost" potential developed by Wolf *et al.* [16].

That stated, it has become apparent that our fitting method and resulting potential (as with any empirical or semi-empirical inter-atomic potential) may be limited with regard to how accurately it models all the physical and mechanical properties desired, *e.g.* miscibility gap concentration boundaries, elastic constants, and cohesive energies. This limitation may be due in part to the use of simple analytic functions used for embedding energy, electron charge density and pair energy that keep the same functional form over their entire respective ranges. This practice may be physically unreasonable, and may need to be abandoned to construct any improvement on our efforts. Another potential limitation is the form of the EAM potential itself, and more advanced potentials may need to be considered. This option is warranted considering the amount of modification to both potential functions and target (*i.e.* fitting) values that was done in order to achieve a miscibility gap. Further modification did not appear to improve the alpha and beta phase concentrations; thus, more complex types of potentials may be necessary to achieve such an improvement. Finally, as pointed out in Chapter 3, the ability of both inter-atomic potentials and *ab inito* methods, *i.e.* DFT, may be inadequate for capturing the complicated physics of hydrogen. The

enhancement of these approaches is no doubt needed for our and other important physics problems.

## 6.2 Future work

We are continuing to improve the fitting of the EAM potential discussed in Chapter 5; in particular, we are focusing on modifications to the parameter fitting process to enable us to represent accurate concentration boundaries to the miscibility gap. Using the existing form of our potential, we will characterize it with regard to physical and mechanical properties (*e.g.* elastic constants, tensile and shear strengths) and perform He bubble growth simulations of multiple, interacting bubbles in bulk material, near free surfaces, and near grain boundaries. These simulations will provide information on fracture and dislocation mechanisms active during He bubble growth, which will be applied to our continuum-scale models thereby producing an accurate capability for predicting accelerated release of He from palladium-tritide and hence, its useful lifetime.

# References


[1] S. Thiébaut, V. Paul-Boncour, A. Percheron Guégan, B. Limacher, O. Blaschko, C. Maier, C. Tailland, and D. Leroy. Structural changes in Pd (Rh, Pt) solid solutions due to $^3$He formation during tritium storage. Physical Review B, 57(17):10379–10387, 1998.

[2] S. Thiébaut, B. Décamps, J.M. Pénisson, B. Limacher, and A. Percheron Guégan. TEM study of the aging of palladium-based alloys during tritium storage. Journal of Nuclear Materials, 277:217–225, 2000.

[3] R. Causey. Interaction of Helium with dislocations in metals: Mini review. 2001.

[4] W.G. Wolfer and W.J. Drugan. Elastic interaction energy between a prismatic dislocation loop and a spherical cavity. Philosophical Magazine A, 57(6):923–927, 1988.

[5] W.G. Wolfer. The pressure for dislocation loop punching by a single bubble. Philosophical Magazine A, 58(2):285–297, 1988.

[6] W.G. Wolfer. Dislocation loop punching in bubble arrays. Philosophical Magazine A, 59(1):87–103, 1989.

[7] D.C. Chrzan and W.G. Wolfer. Helium bubble growth by the dislocation pipe diffusion mechanism. Technical Report SAND91-8671, Sandia National Laboratories, 1991.

[8] R. Rajaraman, B. Viswanathan, M.C. Valsakumar, and K.P. Gopinathan. Anomalous helium-bubble growth in palladium. Physical Review B, 50(1):597–600, 1994.

[9] J.B. Adams, W.G. Wolfer, S.M. Foiles, C.M. Rohlfing, and C.D. Van Siclen. Theoretical Studies of Helium in Metals. In S.E. Donnelly and J.H. Evans, editors, Fundamental Aspects of Inert Gases in Solids, Proceedings of a NATO Advanced Research Workshop, pages 3–16. NATO, 1991.

[10] S.M. Foiles and J.J. Hoyt. Computer Simulation of Bubble Growth in Metals Due to He. Technical Report SAND2001-0661, Sandia National Laboratories, 2001.

[11] J.A. Zimmerman. Computer simulation of boundary effects on bubble growth in metals due to He. Technical Report SAND2003-8137, Sandia National Laboratories, 2003.

[12] D. Zeigler, J. Zimmerman, and Cowgill D. Using a dynamic point-source percolation model to simulate bubble growth. Technical Report SAND2004-1485, Sandia National Laboratories, 2004.

[13] D.F. Cowgill. Helium nano-bubble evolution in aging metal tritides. Technical Report SAND2004-1739, Sandia National Laboratories, 2004.

[14] R. Feenstra, G.J. de Bruin-Hordijk, H.L.M. Bakker, R. Griessen, and D.G. de Groot. Critical point lowering in thin $PdH_x$ films. Journal of Physics F: Metallurgical Physics, 13:L13–L18, 1983.

[15] S.H. Goods and S.E. Guthrie. Mechanical Properties of Palladium and Palladium Hydride. Scripta Metallurgica et Materialia, 26:561–565, 1992.

[16] R.J. Wolf, M.W. Lee, R.C. Davis, P.J. Fay, and J.R. Ray. Pressure-composition isotherms for palladium-hydride. Physical Review B, 48(17):12415–12418, 1993.

[17] M.S. Daw and M.I. Baskes. Embedded-atom-method: Derivation and application to impurities, surfaces, and other defects in metals. Physical Review B, 29(12):6443–6453, 1984.

[18] 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. Physical Review B, 33(12):7983–7991, 1986.

[19] http://www.wolfram.com/, 2007.

[20] M.S. Daw and M.I. Baskes. Semiempirical, quantum mechanical calculation of hydrogen embrittlement in metals. Physical Review Letters, 50(17):1285–1288, 1983.

[21] E. Clementi and C. Roetti. Roothaan-hartree-fock atomic wavefunctions. Atomic Data and Nuclear Data Tables, 14:177–478, 1974.

[22] M.J. Puska and R.M. Nieminen. Atoms embedded in an electron gas: Phase shifts and cross sections. Physical Review B, 27(10):6121–6128, 1983.

[23] M.J. Puska and R.M. Nieminen. Theory of hydrogen and helium impurities in metals. Physical Review B, 29(10):5382–5397, 1984.

[24] M.J. Puska and R.M. Nieminen. Erratum: Theory of hydrogen and helium impurities in metals. Physical Review B, 32(2):1353, 1985.

[25] W. Zhong, Y.S. Li, and D. Tománek. Effect of adsorbates on surface phonon modes: H on Pd(001) and Pd(110). Physical Review B, 44(23):13053–13062, 1991.

[26] W. Zhong, Y. Cai, and D. Tománek. Mechanical stability of Pd-H systems: A molecular-dynamics study. Physical Review B, 46(13):8099–8108, 1992.

[27] R.B. Schwarz, H.T. Bach, U. Harms, and D. Tuggle. Elastic properties of Pd-hydrogen, Pd-deuterium and Pd-tritium single crystals. Acta Materialia, 53:569–580, 2005.

[28] R.J. Wolf, K.A. Mansour, M.W. Lee, and J.R. Ray. Temperature dependence of elastic constants of embedded-atom models of palladium. Physical Review B, 46(13):8027–8035, 1992.

[29] R.J. Wolf, M.W. Lee, and J.R. Ray. Pressure-composition isotherms for nanocrystalline palladium hydride. Physical Review Letters, 73(4):557–569, 1994.

[30] M.W. Lee, R.J. Wolf, and J.R. Ray. Atomistic calculations of hydrogen loading in palladium. Journal of Alloys and Compounds, 231:343–346, 1995.

[31] J.H. Rose, J.R. Smith, F. Guinea, and J. Ferrante. Universal features of the equation of state of metals. Physical Review B, 29:2963–2969, 1984.

[32] K.A. Mansour and R.J. Wolf. Initial report on molecular modeling of metal hydrides (U). Technical Report WSRC-TR-90-156, Westinghouse Savannah River Company, 1990.

[33] R.J. Wolf and K.A. Mansour. Molecular modeling of metal hydrides: II. calculation of lattice defect structures and energies utilizing the embedded atom method (U). Technical Report WSRC-TR-90-275, Westinghouse Savannah River Company, 1990.

[34] T.V. Graham. Mechanical Properties of Palladium with Defects and Impurities: A Molecular Dynamics Study. PhD thesis, Clemson University, 1994.

[35] R.G. Parr and W. Yang. Density-Functional Theory of Atoms and Molecules. Oxford University Press, New York, 1989.

[36] http://cms.mpi.univie.ac.at/vasp/, 1999.

[37] http://www.cs.sandia.gov/∼sjplimp/, 2007.

[38] W.G. Wolfer. The elastic properties of aged palladium tritides. Technical Report SAND90-8253, Sandia National Laboratories, 1990.

[39] R. Caputo and A. Alavi. Where do Hatoms reside in $PdH_x$ systems? Molecular Physics, 101(11):1781–1787, 2003.

[40] http://www.cs.sandia.gov/dakota/software.html, 2007.

[41] http://towhee.sourceforge.net/, 2007.

[42] M.S. Eldred, S.L. Brown, B.M. Adams, D.M. Dunlavy, D.M. Gay, L.P. Swiler, A.A. Giunta, W.E. Hart, J.-P. Watson, J.P. Eddy, J.D. Griffin, P.D. Hough, T.G. Kolda, M.L. Martinez-Canales, and P.J. Williams. DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 4.0 Reference Manual. Technical Report SAND2006-4055, Sandia National Laboratories, 2006.

[43] M.S. Eldred, S.L. Brown, B.M. Adams, D.M. Dunlavy, D.M. Gay, L.P. Swiler, A.A. Giunta, W.E. Hart, J.-P. Watson, J.P. Eddy, J.D. Griffin, P.D. Hough, T.G. Kolda, M.L. Martinez-Canales, and P.J. Williams. DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 4.0 Developers Manual. Technical Report SAND2006-4056, Sandia National Laboratories, 2006.

[44] M.S. Eldred, S.L. Brown, B.M. Adams, D.M. Dunlavy, D.M. Gay, L.P. Swiler, A.A. Giunta, W.E. Hart, J.-P. Watson, J.P. Eddy, J.D. Griffin, P.D. Hough, T.G. Kolda, M.L. Martinez-Canales, and P.J. Williams. DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 4.0 Users Manual. Technical Report SAND2006-6337, Sandia National Laboratories, 2006.

[45] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. SIAM Rev., 47(1):99–131, 2005.

[46] P. Gilmore and C. T. Kelley. An implicit filtering algorithm for optimization of functions with many local minima. SIAM Journal on Optimization, 5(2):269–285, 1995.

[47] C. T. Kelley. Iterative Methods for Optimization. SIAM, Philadelphia, PA, 1999.

[48] http://software.sandia.gov/appspack/version5.0/index.html, 2006.

[49] R. Lasser. Tritium and Helium-3 in Metals, volume 9. Springer-Verlag, Berlin, 1989.

[50] G. Alefeld and J. V́olkl, editors. Hydrogen in Metals, volume 1 of 28-29. Springer-Verlag, Berlin, 1978.

[51] G. Alefeld and J. V́olkl, editors. Hydrogen in Metals, volume 2 of 28-29. Spinger-Verlag, Berlin, 1978.

[52] E. L. Muetterties, editor. Transition Metal Hydrides, volume 1. M. Dekker, New York, 1971.

[53] W. M. Mueller, J. P. Blackledge, and G. G. Libowitz. Metal Hydrides. Academic Press, New York, 1968.

[54] Y. Fukai. The Metal-Hydrogen System, volume 21 of 21. Springer-Verlag, Berlin, 1993.

[55] R. Povel, K. Feucht, W. Gelse, and G. Withalm. Hydrogen fuel for motorcars. Interdiscip. Sci. Rev., 14(4):365–373, 1989.

[56] M. S. Ortman, L. K. Heung, A. Nobile, and R. L. Rabun. Tritium processing at the savannah river site - present and future. J. Vac. Sci. Technol. A, 8(3):2881–2889, 1990.

[57] L. K. Heung. Heat transfer and kinetics of a metal hydride reactor. Zeitschrift fur Physikalische Chem. Neue Folge, 164:1415–1420, 1989.

[58] M. S. Ortman, T. J. Warren, and D. J. Smith. Use of metal hydrides for handling tritium. Fusion Technol., 8(2):2330–2336, 1985.

[59] V. I. Anisimkin, I. M. Kotelyanskii, P. Verardi, and E. Verona. Elastic properties of thin film palladium for surface-accoustic-wave (saw) sensors. Sensors Actuators B, 23(2-3):203–208, 1995.

[60] L. A. Nygren and R. G. Leisure. Elastic constants of $\alpha$'-phase $pdh_x$ over the temperature range 4-300 k. Phys. Rev. B, 37(11):6482–6485, 1988.

[61] D. K. Hsu and R. G. Leisure. Elastic constants of palladium and $\beta$-phase palladium hydride between 4 and 300 k. Phys. Rev. B, 20(4):1339–1344, 1979.

[62] A. A. Lucas. Helium in metals. Physica B-C, 127(1-3):225–239, 1984.

[63] C. Cawthorne and E. J. Fulton. Voids in irradiated stainless steel. Nature, 216(5115):575–576, 1967.

[64] G. C. Abell and A. Attalla. Nmr studies of aging effects in palladium tritide. Fusion Technol., 14(2):643–648, 1988.

[65] R. T. Walters and M. W. Lee. 2 plateaus for palladium hydride and the effect of helium from tritium decay on the desorption plateau pressure for palladium tritide. Mater. Char., 27(3):157–166, 1991.

[66] G. Andreasen, A. Visintin, R. C. Salvarezza, W. E. Triaca, and A. J. Arvia. Hydrogen induced deformation of metals followed by in situ scanning tunneling microscopy, palladium electrolytic hydrogen charging and discharging in alkaline solution. Langmuir, 15(1):1–5, 1999.

[67] W. Zhong, Y. S. Li, and D. Tomanek. Effect of adsorbates on surface phonon modes - h on pd(001) and pd(110). Phys. Rev. B, 44(23):13053–13062, 1991.

[68] R. J. Wolf, P. J. Fay, and J. R. Ray. Private communication, 2002.

[69] X. W. Zhou, H. N. G. Wadley, R. A. Johnson, D. J. Larson, N. Tabat, A. Cerezo, A. K. Petford-Long, G. D. W. Smith, P. H. Clifton, R. L. Martens, , and T. F. Kelly. Atomic scale structure of sputtered metal multilayers. Acta Mater., 49(19):4005–4015, 2001.

[70] W. Zou, H. N. G. Wadley, X. W. Zhou, R. A. Johnson, and D. Brownell. Surfactant mediated growth of giant magnetoresistance multilayers. Phys. Rev. B, 64(17):174418.1–10, 2001.

[71] M. Ruda, D. Farkas, and J. Abriata. Embedded atom interatomic potentials for hydrogen in metals and intermetallic alloys. Phys. Rev. B, 54(14):9765–9774, 1996.

[72] X. W. Zhou, R. A. Johnson, and H. N. G. Wadley. Misfit energy increasing dislocations in vapor deposited cofe/nife multilayers. Phys. Rev. B, 69(14):144113.1–144113.10, 2004.

[73] R. A. Johnson. Alloy models with the embedded atom method. Phys. Rev. B, 39(17):12554–12559, 1989.

[74] S. M. Foiles. Application of the embedded atom method to liquid transition metals. Phys. Rev. B, 32(6):3409–3415, 1985.

[75] M. W. Finnis and J. E. Sinclair. A simple empirical n body potential for transition metals. Phil. Mag. A, 50(1):45–55, 1984.

[76] A. Prince. Alloy Phase Equilibria. Elsevier Publishing Company, Amsterdam, 1966.

[77] Richard A. Swalin. Thermodynamics of Solids. John Wiley & Sons, New York, 1972.

[78] 2007. These calculations were performed for the authors by Thomas K. Mattsson, a technical staff member of the HEDP Theory department at Sandia National Laboratories, Albuquerque, NM.

[79] E.W. Washburn and C.J. West, editors. International Critical Tables of Numerical Data, Physics, Chemistry and Technology, volume 7 of National Research Council. McGraw-Hill Book Company, Inc., New York, 1st ed. edition, 1930.

[80] Chemical Rubber Company. CRC Handbook of Chemistry and Physics. CRC Press, Cleveland, 1977.

[81] 2007. http://www.wolfram.com/products/mathematica/index.html, © Wolfram Research, Inc.

[82] Y. Fukai and H. Sugimoto. Diffusion of hydrogen in metals. Advances in Physics, 34(2):263–326, 1985.

# Appendix A

# FitEAM: A Mathematica Notebook for Analytic Fitting of EAM Potential

## EAM Fitting Codes Developed by Xiaowang Zhou

```
Quit[];
```

## ■ Include Files and Utility Functions

```
$HistoryLength = 0;
Off[General::"spell1"];
Off[General::"spell"];
Off[General::"unfl"];
Needs["Graphics`MultipleListPlot`"];
Needs["Graphics`Legend`"];
Needs["NumericalMath`NLimit`"];
Needs["Statistics`DescriptiveStatistics`"];
Needs["Utilities`MemoryConserve`"];
Off[MemoryConserve::"start"];
Off[MemoryConserve::"end"];
SetDirectory["C:\\xzhou\\ing2"];

Exist = 0;
boltz = 0.0000861734315;

polynomial[x_, npow_] :=
 Block[{n}, Return[c0 + Sum[cn * (x - a0)^n, {n, 1, npow}]]]

(*The occupancy probability of the first element is the same
 as its composition. The occupacy probabilities of other
 elements are then converted from their compositions based
 on the relative number of the correcsponding lattice
 sites with respect to that of the first element.*)
cvtC[raw_, instr_] := Block[{lp11, factor, comp, ith},
  If[Length[raw] > ntypes,
   Print["length of composition exceeds ntypes"]; Quit[]];
  If[Max[raw] > 1 || Min[raw] < 0,
   Print["composition values invalid"]; Quit[]];
  For[lp11 = 1, lp11 ≤ nlat,
   If[Tlat[[lp11]] == instr, ith = lp11]; lp11++];
  factor = Table[natoms[[ith, 1]] / natoms[[ith, lp11]],
    {lp11, 1, ntypes}];
  comp = Table[1, {lp11, 1, ntypes}];
  For[lp11 = ntypes, lp11 ≥ ntypes - Length[raw] + 1, comp[[lp11]] =
    raw[[lp11 + Length[raw] - ntypes]] * factor[[lp11]]; lp11--];
  Return[comp];
 ]

(*ρatmp and ϕtmp are functions of rtmp and vrtmp. If vrtmp→
 rtmp, then UnitStep functions are involved.*)
quickFunc[inparam_] := Block[{lp11, lp12},
  ρatmp = Table[PowerExpand[ρa[lp11, rtmp, vrtmp] /. inparam],
    {lp11, 1, ntypes}];
  ϕtmp = Table[Table[PowerExpand[ϕ[lp11, lp12, rtmp, vrtmp] /.
       inparam], {lp12, 1, ntypes}], {lp11, 1, ntypes}];
  Fatmp = Table[PowerExpand[Fa[lp11, ρtmp] /. inparam],
```

```
    {lp11, 1, ntypes}];
 ]

findN[alat_, pos_, ncen_, nnei_, ij_, nx_, ny_, nz_] :=
 Block[{lp11, lp12, lp13, lp14, ip1, image, separation, rlist},
  rlist = {};
  If[ncen * ij == 0, Return[rlist];];
  For[lp11 = 1, lp11 ≤ Length[nnei],
   ip1 = nnei[[lp11]]; For[lp12 = -nx, lp12 ≤ nx,
    For[lp13 = -ny, lp13 ≤ ny, For[lp14 = -nz, lp14 ≤ nz,
      image = {lp12 * alat, lp13 * alat, lp14 * alat};
      separation = Sqrt[Tr[(pos[[ip1]] - pos[[ncen]] + image)^2]];
      If[separation < rcut[[ij]] && separation > 0.1,
       rlist = Join[rlist, {separation}]];
      lp14++]; lp13++]; lp12++]; lp11++];
  Return[rlist];
 ]

findrlist[ilat_, icen_, jnei_] :=
 Block[{str, alat, type, pos, ncen, nnei, ij, nbox},
  str = Tlat[[ilat]];
  alat = ToExpression["A" <> str] /. r0guessedmin;
  type = ToExpression["T" <> str];
  pos = ToExpression["P" <> str] /. r0guessedmin;
  ncen = Flatten[Position[type, icen]];
  If[Length[ncen] == 0, ncen = 0, ncen = First[ncen];];
  nnei = Flatten[Position[type, jnei]];
  If[icen == jnei, ij = icen,
   If[icen < jnei, ij = icen * ntypes - (icen * (icen + 1))/2 + jnei,
    ij = jnei * ntypes - (jnei * (jnei + 1))/2 + icen];];
  If[rcut[[jnei]] > rcut[[ij]], ij = jnei];
  If[ncen * Length[nnei] == 0, ij = 0;
   nbox = 2, nbox = IntegerPart[rcut[[ij]] / alat] + 1];
  Return[findN[alat, pos, ncen, nnei, ij, nbox, nbox, nbox]];
 ]

(*This Ecoh requires existing Fatmp, ρatmp, and ϕtmp,
and returns E and ρsite as functions of r1 and vr1. If vr1→r1,
then UnitStep functions are involved.*)
Ecoh[ilat_, comp_, r1_, vr1_] :=
 Block[{Etot, ρsite, lp11, lp12, lp13, rij, vrij, ij},
```

```
  Etot = 0;
  ρsite = Table[0, {lp11, 1, ntypes}];
  For[lp11 = 1, lp11 ≤ ntypes,
   For[lp12 = 1, lp12 ≤ ntypes, If[lp11 == lp12, ij = lp11,
     If[lp11 < lp12, ij = lp11 * ntypes - (lp11 * (lp11 + 1))/2 + lp12,
      ij = lp12 * ntypes - (lp12 * (lp12 + 1))/2 + lp11]];]
    For[lp13 = 1, lp13 ≤ Length[zlat[[ilat, lp11, lp12]]],
     rij = r1 * ratiolat[[ilat, lp11, lp12, lp13]];
     vrij = vr1 * ratiolat[[ilat, lp11, lp12, lp13]];
     If[vrij < rcut[[lp12]], ρsite[[lp11]] =
       ρsite[[lp11]] + comp[[lp12]] * zlat[[ilat, lp11, lp12, lp13]] *
         (ρatmp[[lp12]] /. {rtmp → rij, vrtmp → vrij})];
     If[vrij < rcut[[ij]], Etot = Etot + 1/2 * natoms[[ilat, lp11]] *
         comp[[lp11]] * comp[[lp12]] * zlat[[ilat, lp11, lp12, lp13]] *
         (ϕtmp[[lp11, lp12]] /. {rtmp → rij, vrtmp → vrij})];
     lp13++]; lp12++]; lp11++];
  For[lp11 = 1, lp11 ≤ ntypes, Etot = Etot + natoms[[ilat, lp11]] *
      comp[[lp11]] * (Fatmp[[lp11]] /. ρtmp → ρsite[[lp11]]); lp11++];
  Return[PowerExpand[{Etot / Tr[natoms[[ilat]] * comp], ρsite}]];
 ]

pEcoh[i_, comporig_, comp_, rminRd_] :=
 Block[{rmin, rmax, rmax1, lp11, lp12, ij, ratiomin},
  rmax = -1;
  For[rmin = rminRd,
   First[Ecoh[i, comp, rmin, rmin]] > 4, rmin = rmin + 0.01];
  For[lp11 = 1, lp11 ≤ ntypes, For[lp12 = lp11,
    lp12 ≤ ntypes, If[lp11 == lp12, ij = lp11,
     ij = lp11 * ntypes - (lp11 * (lp11 + 1))/2 + lp12];
    rmax1 = Max[rcut[[ij]], rcut[[lp11]], rcut[[lp12]]];
    ratiomin = Min[ratiolat[[i, lp11, lp12]]];
    If[rmax < rmax1/ratiomin &&
      natoms[[i, lp11]] * natoms[[i, lp12]] * comp[[lp11]] * comp[[lp12]] ≠ 0,
     rmax = rmax1/ratiomin]; lp12++]; lp11++];
  Plot[First[Ecoh[i, comp, r, r]], {r, rmin, rmax},
   PlotRange → All, Frame → True, FrameLabel →
    {"r1 (Å)", "E (eV/atom)"}, PlotLabel → "E of lattice " <>
```

```
      ToString[i] <> " with comp. " <> ToString[N[comporig]]];

pEcohall[inparam_, mode_] := Block[{lp11, lp12, compout, ilat},
  quickFunc[inparam];
  If[mode == 1,
   For[lp11 = 1, lp11 ≤ nlat, For[lp12 = 1, lp12 ≤ Length[IniRawC],
     compout = cvtC[IniRawC⟦lp12⟧, Tlat⟦lp11⟧];
     pEcoh[lp11, IniRawC⟦lp12⟧, compout, r0 /. r0guessed1⟦lp11⟧];
     lp12++]; lp11++],
   For[lp11 = 1, lp11 ≤ Length[StructI], compout =
      cvtC[CompI⟦lp11⟧, StructI⟦lp11⟧]; For[lp12 = 1, lp12 ≤ nlat,
      If[StructI⟦lp11⟧ == Tlat⟦lp12⟧, ilat = lp12]; lp12++]; pEcoh[
      ilat, CompI⟦lp11⟧, compout, (r0 /. r0I⟦lp11⟧) - Δ]; lp11++];
  ];
 ]

normalize[inparam_] :=
 Block[{lp11, lp12, ilat, comp, vr1, tmp, n},
  For[n = 1, n ≤ ntypes, shift_n = 0; n++];
  quickFunc[inparam];
  For[lp11 = 1, lp11 ≤ ntypes,
   For[lp12 = 1, lp12 ≤ nlat,
    If[latcn⟦lp11, 1⟧ == Tlat⟦lp12⟧, ilat = lp12]; lp12++];
   comp = latcn⟦lp11, 2⟧;
   For[lp12 = 1, lp12 ≤ ntypes,
    If[comp⟦lp12⟧ > 0.1, n = lp12]; lp12++];
   vr1 = latcn⟦lp11, 3⟧;
   tmp = Ecoh[ilat, comp, vr1, vr1];
   tmp = tmp⟦2⟧;
   tmp = tmp⟦n⟧;
   shift_n = D[Fatmp⟦n⟧, {ρtmp, 1}] /. ρtmp → tmp;
   lp11++];
  quickFunc[inparam];
 ]

Unprotect[DiracDelta];
DiracDelta[x_] := 0

smoothStep[conRate_, xdiv_, x_] := Block[{},
  Return[Exp[conRate * (x - xdiv)] / (Exp[conRate * (x - xdiv)] + 1)];
 ]

smoothStep1[conRate_, xdiv_, x_] := Block[{},
  If[conRate == 0, Return[1]];
  If[conRate > 0, Return[UnitStep[x - xdiv]]];
  If[conRate < 0, Return[UnitStep[xdiv - x]]];
 ]
```

```
strain[vec_] := Block[{stn, vecs, lp11},
  stn = (ϵ11 + 1   1/2 ϵ12   1/2 ϵ13
         1/2 ϵ12   ϵ22 + 1   1/2 ϵ23
         1/2 ϵ13   1/2 ϵ23   ϵ33 + 1);
  vecs = 0 * vec;
  For[lp11 = 1, lp11 ≤ Length[vec],
   vecs[[lp11]] = stn.vec[[lp11]]; lp11++];
  Return[PowerExpand[vecs]];
 ]

findNvec[alat_, pos_, type_, nx_, ny_, nz_] := Block[
  {vec, Tvec, lp11, lp12, lp13, lp14,
   lp15, ip1, ip2, ij, image, vectmp, separation},
  vec = Table[{}, {lp11, 1, Length[pos]}];
  Tvec = Table[{}, {lp11, 1, Length[pos]}];
  For[lp11 = 1, lp11 ≤ Length[pos],
   ip1 = type[[lp11]];
   For[lp12 = 1, lp12 ≤ Length[pos],
    ip2 = type[[lp12]];
    If[ip1 == ip2, ij = ip1,
     If[ip1 < ip2, ij = ip1 * ntypes - (ip1 * (ip1 + 1))/2 + ip2,
       ij = ip2 * ntypes - (ip2 * (ip2 + 1))/2 + ip1];
    ];
    For[lp13 = -nx, lp13 ≤ nx,
     For[lp14 = -ny, lp14 ≤ ny,
      For[lp15 = -nz, lp15 ≤ nz,
       image = {lp13 * alat, lp14 * alat, lp15 * alat};
       vectmp = pos[[lp12]] + image - pos[[lp11]];
       separation = Sqrt[Tr[vectmp^2]];
       If[separation <
          Max[rcut[[ij]], rcut[[ip2]]] && separation > 0.1,
        vec[[lp11]] = Join[vec[[lp11]], {vectmp}];
        Tvec[[lp11]] = Join[Tvec[[lp11]], {ip2}];
       ];
       lp15++];
      lp14++];
     lp13++];
    lp12++];
```

```
    lp11++];
  Return[{vec, Tvec}];
]

GEcAna[alat_, pos_, type_, comp_] := Block[
  {nbox, tmp, atomNei, atomNei1, atomNeiT, ρsite,
   lp11, lp12, ip1, ip2, separation, vseparation, ij},
  nbox = IntegerPart[Max[rcut]/alat] + 1;
  tmp = findNvec[alat, pos, type, nbox, nbox, nbox];
  atomNei = tmp[[1]];
  atomNeiT = tmp[[2]];
  ρsite = Table[0, {lp11, 1, Length[pos]}];
  tmp = 0;
  For[lp11 = 1, lp11 ≤ Length[pos],
   ip1 = type[[lp11]];
   atomNei1 = strain[atomNei[[lp11]]];
   For[lp12 = 1, lp12 ≤ Length[atomNei[[lp11]]],
    ip2 = atomNeiT[[lp11, lp12]];
    If[ip1 == ip2, ij = ip1,
     If[ip1 < ip2, ij = ip1 * ntypes - (ip1 * (ip1 + 1))/2 + ip2,
       ij = ip2 * ntypes - (ip2 * (ip2 + 1))/2 + ip1];
    ];
    separation = Sqrt[Tr[(atomNei1[[lp12]])^2]];
    vseparation = separation /.
      {ϵ11 → 0, ϵ12 → 0, ϵ13 → 0, ϵ22 → 0, ϵ23 → 0, ϵ33 → 0};
    If[vseparation < rcut[[ip2]], ρsite[[lp11]] =
      ρsite[[lp11]] + ρa[ip2, separation, vseparation] * comp[[ip2]]];
    If[vseparation < rcut[[ij]], tmp = tmp + 1/2 * ϕ[ip1, ip2,
        separation, vseparation] * comp[[ip1]] * comp[[ip2]]];
    lp12++];
   lp11++];
  For[lp11 = 1, lp11 ≤ Length[pos],
   ip1 = type[[lp11]];
   tmp = tmp + Fa[ip1, ρsite[[lp11]]] * comp[[ip1]];
   lp11++];
  tmp = PowerExpand[tmp];
  Return[tmp];
]
```

```
Celas[nth_, ilat_] := Block[
  {alat, ecp, factor, tmp},
  alat = ToExpression["A" <> StructI⟦ilat⟧] /. r0I⟦ilat⟧;
  If[nth == "prime",
   ecp =
    {ϵ11 → -ϵ / 2, ϵ12 → 0, ϵ13 → 0, ϵ22 → ϵ, ϵ23 → 0, ϵ33 → -ϵ / 2};
   factor = 1/(3 alat^3);];
  If[nth == "44",
   ecp = {ϵ11 → 0, ϵ12 → 0, ϵ13 → 0, ϵ22 → 0, ϵ23 → ϵ, ϵ33 → 0};
   factor = 1/alat^3;];
  tmp = GEc⟦ilat⟧ /. ecp;
  tmp = D[tmp, {ϵ, 2}];
  tmp = factor * tmp /. ϵ → 0;
  tmp = PowerExpand[tmp];
  Return[tmp];
 ]

CPPString[man_, exp_, myWidth_, myPrecision_] := Block[{},
  ToString[PaddedForm[man, {myWidth, myPrecision}]] <> "e" <>
   ToString[PaddedForm[exp, 2, NumberSigns → {"-", "+"},
     SignPadding → True, NumberPadding → {"0", "0"}]]
 ]
CPPForm[num_, myWidth_, myPrecision_] := Block[{man, exp},
  If[num == 0 || num == 0.,
   exp = 0;
   man = 0,
   exp = Floor[Log[10, Abs[num]]];
   man = N[num / (10^exp)];
   If[Abs[man] ≥ 10., man = man / 10; exp = exp + 1];
  ];
  CPPString[man, exp, myWidth, myPrecision]
 ]

WriteEAM[infilen_, inparam_, nρo_, nro_, ρmaxo_] :=
 Block[{str, dρo, dro, rcuto, lp11, lp12, lp13, ij, tmp, vtmp},
  OpenWrite[infilen];
  quickFunc[inparam];
  str = "Pd-H EAM potential, Xiaowang -
     version1\n*******************\n*******************\n";
  str = str <> ToString[PaddedForm[ntypes, 4]] <> "\n";
  dρo = ρmaxo/(nρo - 1);
```

```
rcuto = Max[rcut];
dro = rcuto/(nro - 1);
str = str <> ToString[PaddedForm[nρo, 4]] <>
  CPPForm[dρo, 18, 16] <> ToString[PaddedForm[nro, 4]] <>
  CPPForm[dro, 18, 16] <> CPPForm[rcuto, 18, 16] <> "\n";
For[lp11 = 1, lp11 ≤ ntypes,
 str = str <> ToString[PaddedForm[ielement[[lp11]], 4]] <>
   ToString[PaddedForm[amass[[lp11]], {9, 2}]] <> ToString[
    PaddedForm[latcn[[lp11, 3]] * Sqrt[2], {13, 4}]] <> "     fcc\n";
 For[lp12 = 1, lp12 ≤ nρo, tmp = (lp12 - 1) * dρo;
  vtmp = Fatmp[[lp11]] /. ρtmp → tmp; If[Abs[vtmp] < 10^-10, vtmp = 0];
  str = str <> CPPForm[vtmp, 18, 16];
  If[5 * IntegerPart[(lp12 / 5)] == lp12,
   str = str <> "\n"]; lp12++];
 For[lp12 = 1, lp12 ≤ nro, tmp = (lp12 - 1) * dro;
  If[tmp ≥ rcut[[lp11]], vtmp = 0,
   vtmp = ρatmp[[lp11]] /. {rtmp → tmp, vrtmp → tmp}]; If[
   Abs[vtmp] < 10^-10, vtmp = 0]; str = str <> CPPForm[vtmp, 18, 16];
  If[5 * IntegerPart[(lp12 / 5)] == lp12, str = str <> "\n"];
  lp12++];
 lp11++];
For[lp11 = 1, lp11 ≤ ntypes,
 For[lp12 = 1, lp12 ≤ lp11,
  If[lp11 == lp12, ij = lp11,
   ij = lp12 * ntypes - (lp12 * (lp12 + 1))/2 + lp11];
  For[lp13 = 1, lp13 ≤ nro,
   tmp = (lp13 - 1) * dro; If[tmp ≥ rcut[[ij]], vtmp = 0,
    vtmp = tmp * ϕtmp[[lp11, lp12]] /. {rtmp → tmp, vrtmp → tmp}];
   If[Abs[vtmp] < 10^-10, vtmp = 0];
   str = str <> CPPForm[vtmp, 18, 16];
   If[5 * IntegerPart[(lp13 / 5)] == lp13, str = str <> "\n"];
   lp13++];
  lp12++];
 lp11++];
WriteString[infilen, str];
Close[infilen];
```

```
]

mychk[inparam_, relax_] :=
 Block[{lp11, StrCompI, StrEcI, StrEccal, StrEwght, StrBI,
   StrBcal, StrBwght, StrPressI, StrPresscal, StrPwght,
   StrCprimeI, StrCprimecal, StrCprimewght, StrC44I,
   StrC44cal, StrC44wght, StrCRelax, StrRelaxr, StrRelaxEc},
  StrCompI = Table[ToString[PaddedForm[CompI[[lp11]], {3, 2}]],
    {lp11, 1, Length[StructI]}];
  StrEcI = Table[ToString[PaddedForm[EcI[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrEccal = Table[ToString[PaddedForm[Eccal[[lp11]] /. inparam,
      {8, 4}]], {lp11, 1, Length[StructI]}];
  StrEwght = Table[ToString[PaddedForm[Ewght[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrBI = Table[ToString[PaddedForm[BI[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrBcal = Table[ToString[PaddedForm[Bcal[[lp11]] /. inparam,
      {8, 4}]], {lp11, 1, Length[StructI]}];
  StrBwght = Table[ToString[PaddedForm[Bwght[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrPressI = Table[ToString[PaddedForm[PressI[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrPresscal = Table[ToString[PaddedForm[Presscal[[lp11]] /.
       inparam, {8, 4}]], {lp11, 1, Length[StructI]}];
  StrPwght = Table[ToString[PaddedForm[Pwght[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrCprimeI = Table[ToString[PaddedForm[CprimeI[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrCprimecal = Table[ToString[PaddedForm[Cprimecal[[lp11]] /.
       inparam, {8, 4}]], {lp11, 1, Length[StructI]}];
  StrCprimewght = Table[ToString[PaddedForm[Cprimewght[[lp11]],
      {8, 4}]], {lp11, 1, Length[StructI]}];
  StrC44I = Table[ToString[PaddedForm[C44I[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  StrC44cal = Table[ToString[PaddedForm[C44cal[[lp11]] /. inparam,
      {8, 4}]], {lp11, 1, Length[StructI]}];
  StrC44wght = Table[ToString[PaddedForm[C44wght[[lp11]], {8, 4}]],
    {lp11, 1, Length[StructI]}];
  Print["check cohesive energy:"];
  Print[TableForm[
    Transpose[{StructI, StrCompI, StrEcI, StrEccal, StrEwght}],
    TableAlignments → Left]];
  Print["check bulk modulus:"];
  Print[TableForm[
    Transpose[{StructI, StrCompI, StrBI, StrBcal, StrBwght}],
    TableAlignments → Left]];
  Print["check pressure:"];
  Print[TableForm[Transpose[{StructI, StrCompI, StrPressI,
      StrPresscal, StrPwght}], TableAlignments → Left]];
  Print["check C':"];
```

```
  Print[TableForm[Transpose[{StructI, StrCompI, StrCprimeI,
      StrCprimecal, StrCprimewght}], TableAlignments → Left]];
  Print["check C44':"];
  Print[TableForm[Transpose[{StructI, StrCompI, StrC44I,
      StrC44cal, StrC44wght}], TableAlignments → Left]];
  If[Length[relax] ≠ 0,
   StrCRelax = Table[ToString[PaddedForm[relax⟦lp11, 1⟧, {3, 2}]],
     {lp11, 1, Length[StructI]}];
   StrRelaxr = Table[ToString[PaddedForm[relax⟦lp11, 2⟧, {8, 4}]],
     {lp11, 1, Length[StructI]}];
   StrRelaxEc = Table[ToString[PaddedForm[relax⟦lp11, 3⟧,
       {8, 4}]], {lp11, 1, Length[StructI]}];
   Print["check relaxed atomic spacing and cohesive energy:"];
   Print[TableForm[Transpose[{StructI, StrCRelax,
       StrRelaxr, StrRelaxEc}], TableAlignments → Left]];
  ];
 ]

continueFit[preVar_, myvar_] := Block[{small, tmp},
  small = 0.01;
  If[Length[preVar] == 0, Return[myvar],
   Return[Table[{myvar⟦lp11⟧, tmp = myvar⟦lp11⟧ /. preVar;
      tmp - small, tmp + small}, {lp11, 1, Length[myvar]}]]];
 ]

Genrange[n_, rmin_] :=
 Block[{lp11, lp12, ij, tmp1, tmp2, rmax, rmax01, rmax02, rmax0},
  rmax01 = 100;
  rmax02 = 100;
  For[lp11 = 1, lp11 ≤ ntypes,
   For[lp12 = lp11, lp12 ≤ ntypes,
    If[lp11 == lp12, ij = lp11,
     ij = lp11 * ntypes - (lp11 * (lp11 + 1))/2 + lp12];
    tmp1 = Select[rmin * ratiolat⟦n, lp11, lp12⟧,
      # < (rcut⟦ij⟧ - smooth⟦ij⟧) &];
    tmp2 = Select[rmin * ratiolat⟦n, lp11, lp12⟧,
      (rcut⟦ij⟧ - smooth⟦ij⟧) ≤ # < rcut⟦ij⟧ &];
    If[Length[tmp1] ≠ 0,
     rmax = Max[tmp1];
     tmp1 = (rcut⟦ij⟧ - smooth⟦ij⟧)/rmax * rmin;
     If[rmax01 > tmp1, rmax01 = tmp1];
    ];
    If[Length[tmp2] ≠ 0,
     rmax = Max[tmp2];
```

```
      tmp2 = rcut[[ij]]/rmax * rmin;
      If[rmax02 > tmp2, rmax02 = tmp2];
    ];
    tmp1 = Select[rmin * ratiolat[[n, lp11, lp12]],
      # < (rcut[[lp11]] - smooth[[lp11]]) &];
    tmp2 = Select[rmin * ratiolat[[n, lp11, lp12]],
      (rcut[[lp11]] - smooth[[lp11]]) ≤ # < rcut[[lp11]] &];
    If[Length[tmp1] ≠ 0,
      rmax = Max[tmp1];
      tmp1 = (rcut[[lp11]] - smooth[[lp11]])/rmax * rmin;
      If[rmax01 > tmp1, rmax01 = tmp1];
    ];
    If[Length[tmp2] ≠ 0,
      rmax = Max[tmp2];
      tmp2 = rcut[[lp11]]/rmax * rmin;
      If[rmax02 > tmp2, rmax02 = tmp2];
    ];
    tmp1 = Select[rmin * ratiolat[[n, lp11, lp12]],
      # < (rcut[[lp12]] - smooth[[lp12]]) &];
    tmp2 = Select[rmin * ratiolat[[n, lp11, lp12]],
      (rcut[[lp12]] - smooth[[lp12]]) ≤ # < rcut[[lp12]] &];
    If[Length[tmp1] ≠ 0,
      rmax = Max[tmp1];
      tmp1 = (rcut[[lp12]] - smooth[[lp12]])/rmax * rmin;
      If[rmax01 > tmp1, rmax01 = tmp1];
    ];
    If[Length[tmp2] ≠ 0,
      rmax = Max[tmp2];
      tmp2 = rcut[[lp12]]/rmax * rmin;
      If[rmax02 > tmp2, rmax02 = tmp2];
    ];
    lp12++];
  lp11++];
 rmax0 = If[rmax02 < rmax01, rmax02, rmax01];
 Return[{rmin, rmax0}];
]
```

```
Genranges[n_, rminRd_] := Block[{ranges, rmin, tmp},
  ranges = {};
  For[rmin = r0 /. r0 → rminRd, rmin ≠ 100, tmp = Genrange[n, rmin];
   rmin = tmp[[2]]; If[rmin ≠ 100, ranges = Join[ranges, {tmp}]]];
  Return[ranges];
 ]

(*This EvsCr determines the "splined" functions of cohesive
 energy as a function of composition and lattice spacing r1.*)
EvsCr[inparam_, curves_] :=
 Block[{lp11, lp12, rminRd, comp, r1, tmp, ilat},
  quickFunc[inparam];
  funcranges = Table[
    For[lp12 = 1, lp12 ≤ nlat,
     If[curves[[lp11, 1]] == Tlat[[lp12]], ilat = lp12]; lp12++];
    comp = curves[[lp11, 2]];
    rminRd = 0;
    For[lp12 = 1, lp12 ≤ Length[comp],
     If[comp[[lp12]] == 1  || comp[[lp12]] == 1.0,
      tmp = latcn[[lp12, 3]] - Δ;
      If[rminRd < tmp, rminRd = tmp];
     ];
     lp12++];
    tmp = Genranges[ilat, rminRd];
    tmp,
    {lp11, 1, Length[curves]}];
  GECr = Table[
    tmp = funcranges[[lp11]];
    comp = cvtC[curves[[lp11, 2]], curves[[lp11, 1]]];
    For[lp12 = 1, lp12 ≤ nlat,
     If[curves[[lp11, 1]] == Tlat[[lp12]], ilat = lp12]; lp12++];
    Table[PowerExpand[First[Ecoh[ilat, comp, r1,
        (tmp[[lp12, 1]] + tmp[[lp12, 2]])/2]]], {lp12, 1, Length[tmp]}],
    {lp11, 1, Length[curves]}];
 ]

findr1e[n_, vCs_] := Block[{lp11, nstart, na, nb, xtmp,
   ytmp, xtmp1, ytmp1, ytmp2, req, vreq, vEc, tmp1, tmp2},
  xtmp = funcranges[[n]];
  ytmp = GECr[[n]];
  vEc = 100 000;
  For[lp11 = 1, lp11 ≤ Length[xtmp],
   xtmp1 = xtmp[[lp11]];
   ytmp1 = ytmp[[lp11]] /. {Cs → vCs, r1 → xtmp1[[1]]};
   If[ytmp1 < vEc, vEc = ytmp1; nstart = lp11];
   lp11++];
```

```
  na = nstart - 1;
  If[na < 1, na = 1];
  nb = nstart + 0;
  vEc = 100;
  For[nstart = na, nstart ≤ nb,
   xtmp1 = xtmp[[nstart]];
   ytmp1 = PowerExpand[ytmp[[nstart]] /. Cs → vCs];
   ytmp2 = PowerExpand[D[ytmp1, {r1, 1}]];
   Off[FindRoot::"reged"];
   Off[FindRoot::"lstol"];
   req = FindRoot[ytmp2 == 0,
     {r1, (xtmp1[[1]] + xtmp1[[2]])/2, xtmp1[[1]], xtmp1[[2]]}];
   On[FindRoot::"reged"];
   On[FindRoot::"lstol"];
   tmp1 = r1 /. req;
   tmp2 = ytmp1 /. req;
   If[tmp2 < vEc, vreq = tmp1; vEc = tmp2;];
   nstart++];
  Return[{vreq, vEc}];
 ]

pErvsC[inparam_, curves_, ncomp_, nGibbs_] :=
 Block[{npts, lp11, lp12, datapts, datapts1, datapts2,
   vCs, vr1E, energyPts, pcomp, flag, ptmp, ilat, p1, p2},
  EvsCr[inparam, curves];
  npts = 30;
  pcomp = {};
  energyPts = 0;
  For[lp11 = 1, lp11 ≤ Length[curves],
   datapts =
    Table[{vCs = (lp12 - 1) * 1 / (npts - 1); vr1E = findr1e[lp11, vCs];
      vCs, vr1E[[1]], vr1E[[2]]}, {lp12, 1, npts}];
   datapts1 = Table[{datapts[[lp12, 1]], datapts[[lp12, 2]]},
     {lp12, 1, npts}];
   datapts2 = Table[{datapts[[lp12, 1]], datapts[[lp12, 3]]},
     {lp12, 1, npts}];
   p1 = ListPlot[datapts1, PlotJoined → False,
     PlotStyle → PointSize[0.02], PlotRange → All,
     DisplayFunction → Identity];
   p2 = ListPlot[datapts2, PlotJoined → False,
     PlotStyle → PointSize[0.02], PlotRange → All,
     DisplayFunction → Identity];
   If[lp11 == nGibbs, energyPts = datapts];
   flag = Flatten[Position[ncomp, lp11]];
   If[Length[flag] ≠ 0,
    If[flag[[1]] == 1,
     ptmp = ListPlot[datapts2, PlotJoined → False, PlotStyle →
```

```
        {Hue[0.5], Hue[0.6], Hue[0.7], PointSize[0.02]},
       PlotRange → All, DisplayFunction → Identity],
      ptmp = p2;
     ];
     pcomp = Join[pcomp, {ptmp}];
    ];
    ilat = 0;
    For[lp12 = 1, lp12 ≤ Length[DFTdata],
     If[DFTdata⟦lp12⟧ == "DFTdata" <> curves⟦lp11, 1⟧ &&
       DFTcomp⟦lp12⟧ == curves⟦lp11, 2⟧, ilat = lp12]; lp12++];
    If[ilat ≠ 0,
     If[NumberQ[pWolfa⟦ilat⟧] == False,
      Show[pDFTa⟦ilat⟧, p1, pWolfa⟦ilat⟧,
       DisplayFunction → $DisplayFunction, Frame → True,
       FrameLabel → {"composition", "r1 (Å)"},
       PlotLabel → "r1 of lattice " <> ToString[curves⟦lp11⟧]],
      Show[pDFTa⟦ilat⟧, p1, DisplayFunction → $DisplayFunction,
       Frame → True, FrameLabel → {"composition", "r1 (Å)"},
       PlotLabel → "r1 of lattice " <> ToString[curves⟦lp11⟧]]
     ];
     Show[pDFTe⟦ilat⟧, p2, DisplayFunction → $DisplayFunction,
      Frame → True, FrameLabel → {"composition", "Ecoh (eV/atom)"},
      PlotLabel → "Ecoh of lattice " <> ToString[curves⟦lp11⟧]],
     Show[p1, DisplayFunction → $DisplayFunction,
      Frame → True, FrameLabel → {"composition", "r1 (Å)"},
      PlotLabel → "r1 of lattice " <> ToString[curves⟦lp11⟧]];
     Show[p2, DisplayFunction → $DisplayFunction, Frame → True,
      FrameLabel → {"composition", "Ecoh (eV/atom)"},
      PlotLabel → "Ecoh of lattice " <> ToString[curves⟦lp11⟧]];
    ];
    lp11++];
   If[Length[pcomp] ≠ 0,
    Show[pcomp, DisplayFunction → $DisplayFunction, Frame → True,
      FrameLabel → {"composition", "Ecoh (eV/atom)"},
      PlotLabel → "Ecoh of curves " <> ToString[ncomp] <>
        ", curve " <> ToString[ncomp⟦1⟧] <> " (blue)"];
   ];
   Return[energyPts];
  ]

Gibbs[energyPts_, T_, nGibbs_] :=
 Block[{lp11, data, datafitpow, condition1, condition2,
   condition, datafitcoe, p1, p2, tmpa, tmpb, tmp, stmpa,
   stmpb, stmp, ilat, comp1, comp2, numa, sum, ΔH, ΔS, ΔG},
  data = Table[{energyPts⟦lp11, 1⟧, energyPts⟦lp11, 3⟧},
    {lp11, 1, Length[energyPts]}];
  datafitpow = 3;
```

```
  condition1 = {{a0, 0.4}, {c0, -3.8}};
  condition2 = Table[cn, {n, 1, datafitpow}];
  condition = Join[condition1, condition2];
  datafitcoe =
   FindFit[data, polynomial[xa, datafitpow], condition, xa];
  For[lp11 = 1, lp11 ≤ nlat,
   If[curves⟦nGibbs, 1⟧ == Tlat⟦lp11⟧, ilat = lp11]; lp11++];
  comp1 = curves⟦nGibbs, 2⟧ /. Cs → x;
  comp2 = cvtC[comp1, Tlat⟦ilat⟧];
  nformula = Tr[comp1] /. x → 0;
  numa = natoms⟦ilat⟧;
  tmp = polynomial[x, datafitpow] /. datafitcoe;
  p1 = ListPlot[data, PlotJoined → False,
    PlotStyle → PointSize[0.02], DisplayFunction → Identity];
  p2 = Plot[tmp, {x, 0, 1}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "Ecoh (eV/atom)"},
   PlotLabel → "Ecoh of curve " <> ToString[nGibbs]];
  tmpa = tmp /. x → 0;
  tmpb = tmp /. x → 1;
  ΔH = 1/(nformula + x) ((nformula + x) * tmp -
     (nformula + 1) * x * tmpb - nformula * (1 - x) * tmpa);
  Plot[ΔH, {x, 0, 1}, Frame → True,
   FrameLabel → {"composition", "ΔH (eV/atom)"},
   PlotLabel → "ΔH of curve " <> ToString[nGibbs]];
  sum = 0;
  For[lp11 = 1, lp11 ≤ Length[comp2],
   If[(comp2⟦lp11⟧ ≠ 0 && comp2⟦lp11⟧ ≠ 1) ||
     NumberQ[comp2⟦lp11⟧] == False,
    sum = sum + numa⟦lp11⟧ * (comp2⟦lp11⟧ * Log[comp2⟦lp11⟧] +
          (1 - comp2⟦lp11⟧) * Log[1 - comp2⟦lp11⟧])];
   lp11++];
  stmp = FullSimplify[sum/(Tr[numa * comp2])];
  stmpa = Limit[stmp, x → 0];
  stmpb = Limit[stmp, x → 1];
  ΔS = -boltz * 1/(nformula + x) ((nformula + x) * stmp -
     (nformula + 1) * x * stmpb - nformula * (1 - x) * stmpa);
  ΔG = ΔH - T * ΔS;
  Return[ΔG];
 ]

pGibbs[GibbsExpr_, vT_] := Block[{tmp},
  tmp = GibbsExpr /. {T → vT, x → (xa * nformula)/(1 - xa)};
```

```
  Plot[tmp, {xa, 0, 1/(1 + nformula)}, Frame → True, FrameLabel →
    {"composition (atomic fraction)", "ΔG (eV/atom)"},
   PlotLabel → "ΔG of curve " <> ToString[nGibbs] <>
     " at temperature " <> ToString[vT] <> " K"];
 ]

relaxedEr[curves_] := Block[
  {data, lp11, lp12, lp13, ilat, jlat, match, nth, vCs, vr1E1},
  data = Table[
    {
     For[lp12 = 1, lp12 ≤ Length[curves],
      If[curves[[lp12, 1]] == StructI[[lp11]],
       match = 0;
       For[lp13 = 1, lp13 ≤ Length[CompI[[lp11]]],
        If[curves[[lp12, 2, lp13]] === Cs,
         nth = lp13;
         match++,
         If[curves[[lp12, 2, lp13]] == CompI[[lp11, lp13]], match++];
        ];
        lp13++];
       If[match == Length[CompI[[lp11]]], ilat = lp12; jlat = nth];
      ];
      lp12++];
     vCs = CompI[[lp11, jlat]];
     vr1E1 = findr1e[ilat, vCs];
     CompI[[lp11]], vr1E1[[1]], vr1E1[[2]]
    },
    {lp11, 1, Length[StructI]}];
  Return[data];
 ]

eval[inparam_] :=
 Block[{energyPts, GibbsExpr, curves, ncomp, nGibbs},
  Print[rcut];
  Print[latcn];
  Print["EABfit = ", EABfit];
  Print["Ewght = ", Ewght];
  Print["Bwght = ", Bwght];
  Print["Pwght = ", Pwght];
  Print["Cprimewght = ", Cprimewght];
  Print["C44wght = ", C44wght];
  Print["constraint1 = ", constraint1];
  Print["constraint2 = ", constraint2];
  outputEAM[inparam];
  pEcohall[inparam, 2];
  curves = ( "OC"  {1, Cs}
             "TE"  {1, Cs}
             "OC"  {Cs, 1} );
```

```
  ncomp = {1, 2};
  nGibbs = 1;
  energyPts = pErvsC[inparam, curves, ncomp, nGibbs];
  mychk[inparam, relaxedEr[curves]];
  GibbsExpr = Gibbs[energyPts, T, nGibbs];
  For[vT = 300, vT ≤ 1500, pGibbs[GibbsExpr, vT]; vT = vT + 200];
]
```

## ■ Pre-analysis

```
(*shifted based on experimental cohesive
 energies of Pd and PdH0.5, see DFT.nb*)
```

$$\text{DFTdataOC} = \begin{pmatrix} 0.000 & \begin{pmatrix} 3.66 & -3.31179 \\ 3.76 & -3.67153 \\ 3.86 & -3.85487 \\ 3.96 & -3.90865 \\ 4.06 & -3.86992 \\ 4.16 & -3.76631 \end{pmatrix} \\ 0.250 & \begin{pmatrix} 3.90 & -3.58662 \\ 3.95 & -3.62741 \\ 4.00 & -3.64521 \\ 4.05 & -3.64346 \\ 4.10 & -3.62502 \\ 4.15 & -3.59257 \end{pmatrix} \\ 0.500 & \begin{pmatrix} 4.00 & -3.43762 \\ 4.05 & -3.48445 \\ 4.10 & -3.48169 \\ 4.15 & -3.46485 \\ 4.20 & -3.43610 \end{pmatrix} \\ 0.750 & \begin{pmatrix} 4.00 & -3.32552 \\ 4.05 & -3.35224 \\ 4.10 & -3.36235 \\ 4.15 & -3.35816 \\ 4.20 & -3.34175 \\ 4.25 & -3.31481 \end{pmatrix} \\ 1.000 & \begin{pmatrix} 4.00 & -3.19372 \\ 4.05 & -3.23095 \\ 4.10 & -3.25152 \\ 4.15 & -3.25760 \\ 4.20 & -3.25125 \\ 4.25 & -3.20755 \end{pmatrix} \end{pmatrix};$$

```
DFTdata = {"DFTdataOC"};
DFTcomp = {{1, Cs}};
DFTcaeb = Table[, {lp, 1, Length[DFTdata]}];
pDFTa = Table[, {lp, 1, Length[DFTdata]}];
pDFTe = Table[, {lp, 1, Length[DFTdata]}];
pDFTb = Table[, {lp, 1, Length[DFTdata]}];
```

```
Datareduce1[DFTfitpow_, nth_] :=
 Block[{condition1, condition2, condition,
   this, data0, p1, p2, tmp, tmp1, tmp2, xaeq, vxa},
  condition1 = {{a0, 4}, {c0, -3.5}};
  condition2 = Table[cn, {n, 1, DFTfitpow}];
  condition = Join[condition1, condition2];
  this = ToExpression[DFTdata[[nth]]];
  For[i = 1, i ≤ Length[this],
   data0 = this[[i, 2]];
   DFTcoei =
    FindFit[data0, polynomial[xa, DFTfitpow], condition, xa];
   p1 = Plot[polynomial[xa, DFTfitpow] /. DFTcoei,
     {xa, data0[[1, 1]], data0[[Length[data0], 1]]},
     DisplayFunction → Identity];
   p2 = ListPlot[data0, PlotJoined → False, PlotStyle →
      {PointSize[0.02]}, DisplayFunction → Identity];
   Show[p1, p2, DisplayFunction → $DisplayFunction, Frame → True,
    FrameLabel → {"a (Å)", "E (eV/atom)"}, PlotLabel →
     "E vs a at composition " <> ToString[this[[i, 1]]]];
   i++];
  DFTcaeb[[nth]] =
   Table[tmp = polynomial[xa, DFTfitpow] /. DFTcoei; tmp1 =
     D[tmp, {xa, 1}]; tmp2 = 4 * (1 + this[[i, 1]]) * D[tmp, {xa, 2}];
    data0 = this[[i, 2]]; xaeq = FindRoot[tmp1 == 0,
      {xa, (data0[[1, 1]] + data0[[Length[data0], 1]])/2, data0[[1, 1]],
       data0[[Length[data0], 1]]}]; vxa = xa /. xaeq;
    {this[[i, 1]], vxa, tmp /. xaeq, (tmp2 /. xaeq)/(9 * vxa)},
    {i, 1, Length[this]}];
  Print[PaddedForm[TableForm[DFTcaeb[[nth]]], {8, 4}]];
  For[i = 1, i ≤ Length[DFTcaeb[[nth]]],
   DFTcaeb[[nth, i, 2]] = DFTcaeb[[nth, i, 2]]/Sqrt[2]; i++];
  Print["Raw data to be copied to DFTcaeb"];
  Print[TableForm[DFTcaeb[[nth]]]];
```

$$\text{DFTcaeb}[[1]] = \begin{pmatrix} 0.` & 2.7460491847379576` & -3.91` & 1.077 \\ 0.25` & 2.787485642115489` & -3.5668` & 1.026 \\ 0.5` & 2.828922099493021` & -3.4677` & 2.114 \\ 0.75` & 2.8703585568705527` & -3.4126` & 1.060 \\ 1.` & 2.911795014248084` & -3.3807` & 1.769 \end{pmatrix}$$

```
  Print["Redesign for heat of mixing"];
  Print[TableForm[DFTcaeb[[nth]]]];
  tmp = Table[{DFTcaeb[[nth, i, 1]], DFTcaeb[[nth, i, 2]]},
```

```
    {i, 1, Length[DFTcaeb[[nth]]]};
  pDFTa[[nth]] = ListPlot[tmp, PlotJoined → False,
    PlotStyle → {PointSize[0.04]},
    PlotRange → All, DisplayFunction → Identity];
  tmp = Table[{DFTcaeb[[nth, i, 1]], DFTcaeb[[nth, i, 3]]},
    {i, 1, Length[DFTcaeb[[nth]]]}];
  pDFTe[[nth]] = ListPlot[tmp, PlotJoined → False,
    PlotStyle → {PointSize[0.04]},
    PlotRange → All, DisplayFunction → Identity];
  tmp = Table[{DFTcaeb[[nth, i, 1]], DFTcaeb[[nth, i, 4]]},
    {i, 1, Length[DFTcaeb[[nth]]]}];
  pDFTb[[nth]] = ListPlot[tmp, PlotJoined → False,
    PlotStyle → {PointSize[0.04]},
    PlotRange → All, DisplayFunction → Identity];
  Show[pDFTa[[nth]], DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "r1 (Å)"},
   PlotLabel → "DFT data"];
  Show[pDFTe[[nth]], DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "Ecoh (eV/atom)"},
   PlotLabel → "DFT data"];
  Show[pDFTb[[nth]], DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "B (eV/Å^3)"},
   PlotLabel → "DFT data"];
 ]
DFTfitpow = 3;
For[lp = 1, lp ≤ Length[DFTdata],
  Datareduce1[DFTfitpow, lp]; lp++];
```

$$\text{CpSchwarz} = \begin{pmatrix} 0.58667 & 26.18140 \\ 0.59203 & 26.30698 \\ 0.59619 & 26.48372 \\ 0.60453 & 26.84651 \\ 0.61147 & 27.02326 \\ 0.61900 & 27.16744 \\ 0.64415 & 27.20000 \\ 0.65682 & 27.09767 \\ 0.67266 & 27.01860 \\ 0.68513 & 26.92093 \\ 0.68988 & 26.82326 \\ 0.69324 & 26.86047 \\ 0.69660 & 26.71628 \\ 0.71422 & 26.59070 \\ 0.71818 & 26.58140 \\ 0.72629 & 26.44186 \\ 0.73995 & 26.29302 \\ 0.75499 & 26.13488 \\ 0.76825 & 26.07442 \end{pmatrix};$$

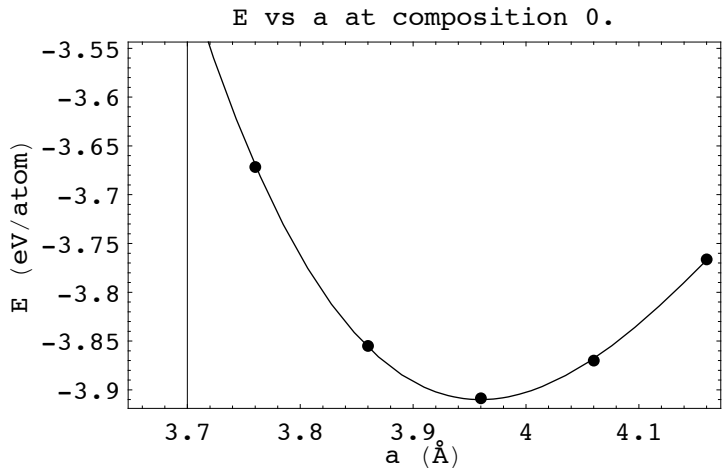
```
C44Schwarz = (0.58679  65.14296
              0.59235  65.10730
              0.59606  64.97181
              0.60451  64.87911
              0.61131  64.36567
              0.61894  64.26583
              0.64408  63.62403
              0.65583  63.53132
              0.67273  62.76829
              0.68468  62.99649
              0.68942  62.70411
              0.69334  62.26911
              0.69664  62.46165
              0.71395  62.20493
              0.71807  61.57026
              0.72611  61.81272
              0.73930  61.43477
              0.75475  60.85002
              0.76794  60.95698);
```

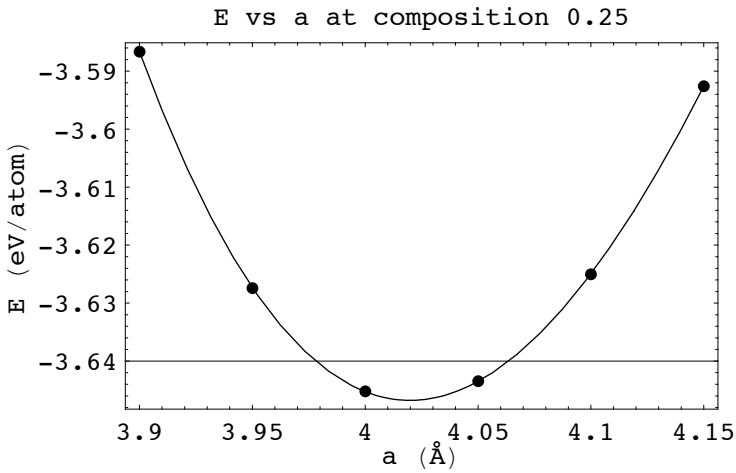
```
vExprSchwarz[coes_, npow_, xamin_, xamax_, xa_] :=
 Block[{tmp, xvar, xpt, ypt, dypt},
  tmp = polynomial[xvar, npow] /. coes;
  If[xamin ≤ xa && xa ≤ xamax, Return[tmp /. xvar → xa];];
  If[xa < xamin,
   xpt = xamin;
   ypt = tmp /. xvar → xamin;
   dypt = D[tmp, {xvar, 1}] /. xvar → xamin;
  ];
  If[xa > xamax,
   xpt = xamax;
   ypt = tmp /. xvar → xamax;
   dypt = D[tmp, {xvar, 1}] /. xvar → xamax;
  ];
  Return[ypt + dypt * (xa - xpt)];
 ]
```

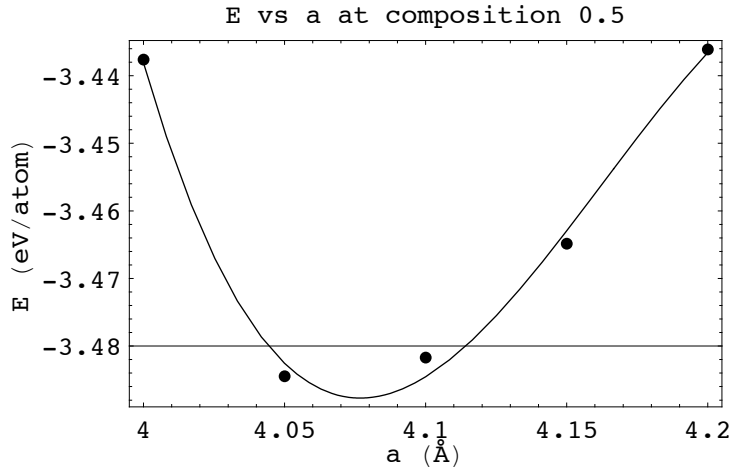
```
Datareduce2[CpSchwarzfitpow_, C44Schwarzfitpow_] :=
 Block[{condition1, condition2, condition, p1, p2},
  condition1 = {{a0, 0.67}, {c0, 45 / 160}};
  For[i = 1, i ≤ Length[CpSchwarz],
   CpSchwarz[[i, 2]] = CpSchwarz[[i, 2]] / 160; i++];
  For[i = 1, i ≤ Length[C44Schwarz],
   C44Schwarz[[i, 2]] = C44Schwarz[[i, 2]] / 160; i++];
  condition2 = Table[cn, {n, 1, CpSchwarzfitpow}];
  condition = Join[condition1, condition2];
```

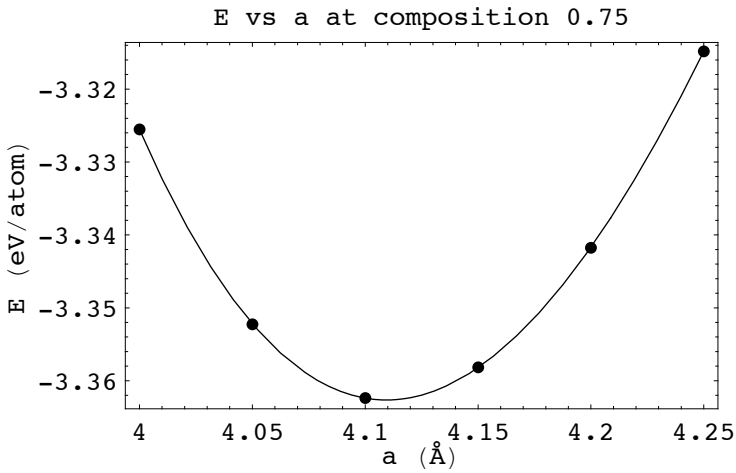
```
  CpSchwarzcoe = FindFit[CpSchwarz,
    polynomial[xa, CpSchwarzfitpow], condition, xa];
  condition2 = Table[c_n, {n, 1, C44Schwarzfitpow}];
  condition = Join[condition1, condition2];
  C44Schwarzcoe = FindFit[C44Schwarz,
    polynomial[xa, C44Schwarzfitpow], condition, xa];
  p1 = Plot[vExprSchwarz[CpSchwarzcoe, CpSchwarzfitpow,
     CpSchwarz[[1, 1]], CpSchwarz[[Length[CpSchwarz], 1]], xa],
    {xa, 0.5, 0.8}, DisplayFunction → Identity];
  p2 = ListPlot[CpSchwarz, PlotJoined → False,
    PlotStyle → {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "C' (eV/Å^3)"},
   PlotLabel → "Schwarz data"];
  p1 = Plot[vExprSchwarz[C44Schwarzcoe, C44Schwarzfitpow,
     C44Schwarz[[1, 1]], C44Schwarz[[Length[C44Schwarz], 1]], xa],
    {xa, 0.5, 0.8}, DisplayFunction → Identity];
  p2 = ListPlot[C44Schwarz, PlotJoined → False,
    PlotStyle → {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"composition", "C44 (eV/Å^3)"},
   PlotLabel → "Schwarz data"];
  Print["C' "];
  Print[vExprSchwarz[CpSchwarzcoe, CpSchwarzfitpow,
    CpSchwarz[[1, 1]], CpSchwarz[[Length[CpSchwarz], 1]], 0.25]];
  Print[vExprSchwarz[CpSchwarzcoe, CpSchwarzfitpow,
    CpSchwarz[[1, 1]], CpSchwarz[[Length[CpSchwarz], 1]], 0.5]];
  Print[vExprSchwarz[CpSchwarzcoe, CpSchwarzfitpow,
    CpSchwarz[[1, 1]], CpSchwarz[[Length[CpSchwarz], 1]], 0.75]];
  Print[vExprSchwarz[CpSchwarzcoe, CpSchwarzfitpow,
    CpSchwarz[[1, 1]], CpSchwarz[[Length[CpSchwarz], 1]], 1.0]];
  Print["C44 "];
  Print[vExprSchwarz[C44Schwarzcoe,
    C44Schwarzfitpow, C44Schwarz[[1, 1]],
    C44Schwarz[[Length[C44Schwarz], 1]], 0.25]];
  Print[vExprSchwarz[C44Schwarzcoe, C44Schwarzfitpow,
    C44Schwarz[[1, 1]], C44Schwarz[[Length[C44Schwarz], 1]], 0.5]];
  Print[vExprSchwarz[C44Schwarzcoe,
    C44Schwarzfitpow, C44Schwarz[[1, 1]],
    C44Schwarz[[Length[C44Schwarz], 1]], 0.75]];
  Print[vExprSchwarz[C44Schwarzcoe, C44Schwarzfitpow,
    C44Schwarz[[1, 1]], C44Schwarz[[Length[C44Schwarz], 1]], 1.0]];
 ]
CpSchwarzfitpow = 4;
C44Schwarzfitpow = 1;
Datareduce2[CpSchwarzfitpow, C44Schwarzfitpow];

pWolfa = Table[0, {lp, 1, Length[DFTdata]}];
```

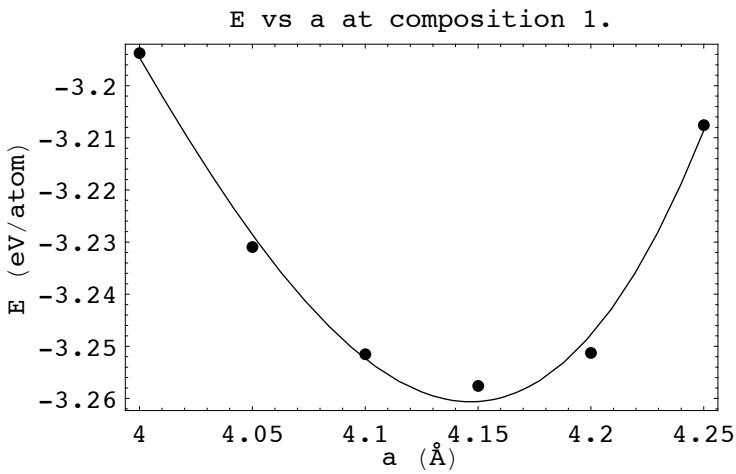
```
pWolfa[[1]] = Plot[(3.8835 + 0.2344 * x)/Sqrt[2],
   {x, 0, 0.8}, DisplayFunction → Identity];
```





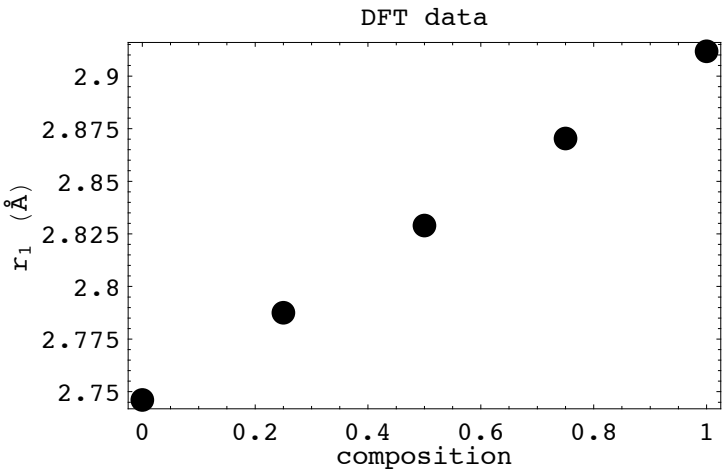

| | | | |
|---|---|---|---|
| 0.0000 | 3.9596 | -3.9100 | 1.0780 |
| 0.2500 | 4.0198 | -3.6468 | 1.0262 |
| 0.5000 | 4.0769 | -3.4877 | 2.1143 |
| 0.7500 | 4.1094 | -3.3626 | 1.0602 |
| 1.0000 | 4.1467 | -3.2606 | 1.7697 |

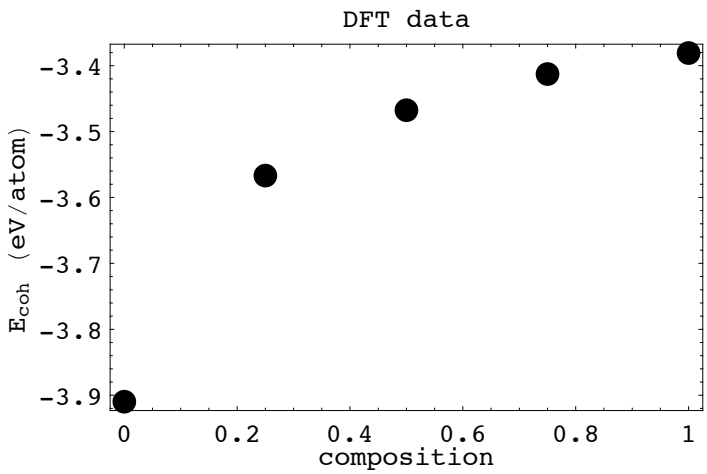
Raw data to be copied to DFTcaeb

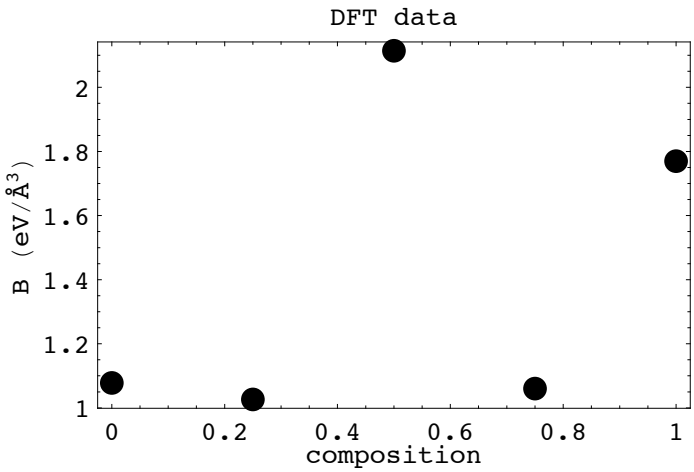
| | | | |
|---|---|---|---|
| 0. | 2.79986 | -3.91001 | 1.07798 |
| 0.25 | 2.84245 | -3.64677 | 1.02621 |
| 0.5 | 2.88278 | -3.4877 | 2.11427 |
| 0.75 | 2.90581 | -3.36264 | 1.06024 |
| 1. | 2.93214 | -3.26065 | 1.76974 |

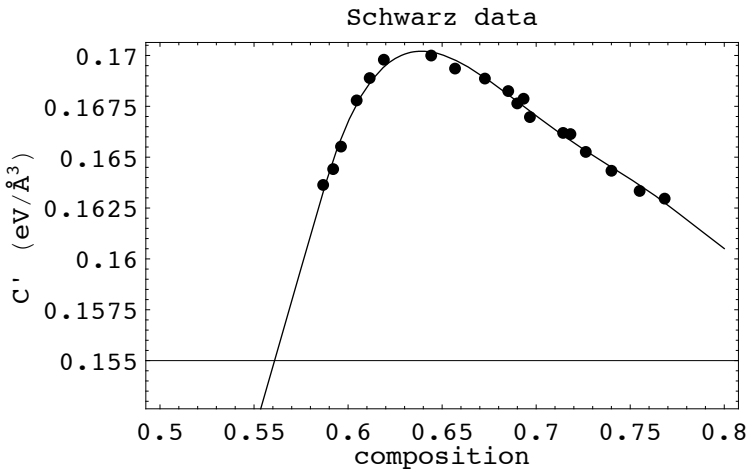
Redesign for heat of mixing

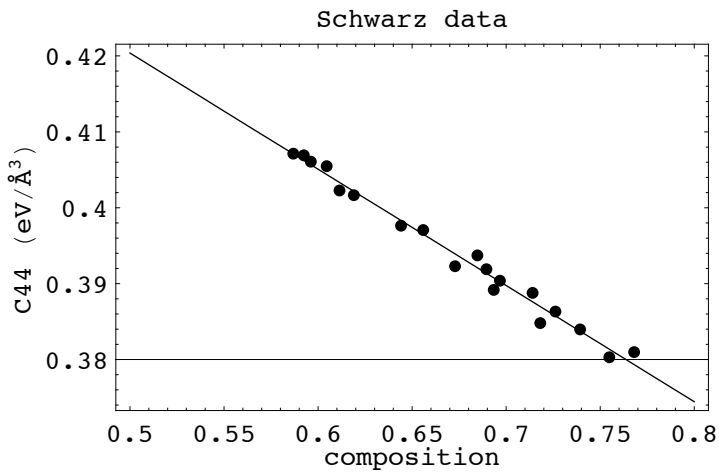
| | | | |
|---|---|---|---|
| 0. | 2.74605 | -3.91 | 1.07798 |
| 0.25 | 2.78749 | -3.5668 | 1.02621 |
| 0.5 | 2.82892 | -3.4677 | 2.11427 |
| 0.75 | 2.87036 | -3.4126 | 1.06024 |
| 1. | 2.9118 | -3.3807 | 1.76974 |


DFT data
2.9
2.875
2.85
2.825
2.8
2.775
2.75
r1 (Å)
0
0.2
0.4
0.6
0.8
1
composition


DFT data
-3.4
-3.5
-3.6
-3.7
-3.8
-3.9
Ecoh (eV/atom)
0
0.2
0.4
0.6
0.8
1
composition


DFT data
2
1.8
1.6
1.4
1.2
1
B (eV/Å3)
0
0.2
0.4
0.6
0.8
1
composition




C'

0.0552133

0.135452

0.163923

0.146278

C44

0.458684

0.420389

0.382094

0.343799

Null[19]

## ■ Lattices

```
ntypes = 2;
Pd = 1;
H = 2;
ielement = {46, 1};
amass = {106.42, 1.00794};
(*All compositions must be defined in latcn. Each set of the
 compositions define a pure element. The structures used in
 latcn must define lattice occupancy probabilities that equal
 compositions because normalize function cannot call cvtC.*)
```

$$\text{latcn} = \begin{pmatrix} \text{"OC"} & \{1, 0\} & \frac{3.88486}{\sqrt{2}} \\ \text{"OC"} & \{0, 1\} & \frac{2.28400}{\sqrt{2}} \end{pmatrix};$$

$$\text{latcn} = \begin{pmatrix} \text{"OC"} & \{1, 0\} & \frac{3.88486}{\sqrt{2}} \\ \text{"OC"} & \{0, 1\} & \frac{4.00000}{\sqrt{2}} \end{pmatrix};$$

$$\text{latcn} = \begin{pmatrix} \text{"OC"} & \{1, 0\} & \frac{3.88486}{\sqrt{2}} \\ \text{"OC"} & \{0, 1\} & \frac{3.50000}{\sqrt{2}} \end{pmatrix};$$

$$\text{latcn} = \begin{pmatrix} \text{"OC"} & \{1, 0\} & \frac{3.88486}{\sqrt{2}} \\ \text{"OC"} & \{0, 1\} & \frac{3.40000}{\sqrt{2}} \end{pmatrix};$$

$$\text{latcn} = \begin{pmatrix} \text{"OC"} & \{1, 0\} & \frac{3.88486}{\sqrt{2}} \\ \text{"OC"} & \{0, 1\} & \frac{3.38000}{\sqrt{2}} \end{pmatrix};$$

$$\text{npair} = \frac{\text{ntypes}\ (\text{ntypes} + 1)}{2};$$

```
rcut = {5.35, 5.35, 4.95};
smooth = {0.3, 0.3, 0.3};
Tlat = {"OC", "TE"};
nlat = Length[Tlat];
```

Pd-H with H at octahedron sites

$$\text{AOC} = \sqrt{2}\ \text{r0};$$

$$\text{TOC} = \{\text{Pd}, \text{Pd}, \text{Pd}, \text{Pd}, \text{H}, \text{H}, \text{H}, \text{H}\};$$

$$\text{POC} = \text{AOC}\left\{\{0, 0, 0\}, \left\{\frac{1}{2}, \frac{1}{2}, 0\right\}, \left\{\frac{1}{2}, 0, \frac{1}{2}\right\}, \left\{0, \frac{1}{2}, \frac{1}{2}\right\}, \left\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right\}, \left\{\frac{1}{2}, 0, 0\right\}, \left\{0, \frac{1}{2}, 0\right\}, \left\{0, 0, \frac{1}{2}\right\}\right\};$$

Pd-H with H at tetrahedron sites

```
ATE = √2 r0;
TTE = {Pd, Pd, Pd, Pd, H, H, H, H, H, H, H, H};
PTE = ATE {{0, 0, 0}, {1/2, 1/2, 0}, {1/2, 0, 1/2}, {0, 1/2, 1/2},
    {1/4, 1/4, 1/4}, {3/4, 1/4, 1/4}, {1/4, 1/4, 3/4}, {3/4, 1/4, 3/4},
    {1/4, 3/4, 1/4}, {3/4, 3/4, 1/4}, {1/4, 3/4, 3/4}, {3/4, 3/4, 3/4}};
```

Compositions

```
IniRawC = {{0}, {1/4}, {1/2}, {3/4}};
Δ = 0.35;
```

The following r0guessed1 and r0guessed2 inputs are guessed nearest neighbor distances, where r0guessed1 is for Tlat at the intial compositions IniRawc (Length[r0guessed1]=Length[Tlat]), and r0guessed2 is for StructI (Length[r0guessed2]=Length[StructI]).

$$r0guessed1 = \begin{pmatrix} r0 \to \frac{4.0198}{\sqrt{2}} \\ r0 \to \frac{4.0769}{\sqrt{2}} \end{pmatrix};$$

$$r0guessed2 = \begin{pmatrix} r0 \to \frac{4.0198}{\sqrt{2}} \\ r0 \to \frac{4.0769}{\sqrt{2}} \\ r0 \to \frac{4.1094}{\sqrt{2}} \\ r0 \to \frac{4.1467}{\sqrt{2}} \\ r0 \to \frac{4.0769}{\sqrt{2}} \\ r0 \to \frac{4.1094}{\sqrt{2}} \\ r0 \to \frac{2.2840}{\sqrt{2}} \end{pmatrix};$$

```
r0guessedmin = 100;
Block[{tmp},
 For[lp1 = 1, lp1 ≤ Length[r0guessed1],
  tmp = (r0 /. r0guessed1⟦lp1⟧) - Δ;
  If[r0guessedmin > tmp, r0guessedmin = tmp];
  r0guessed1⟦lp1⟧ = {r0 → tmp};
  lp1++];
 For[lp1 = 1, lp1 ≤ Length[r0guessed2],
  tmp = (r0 /. r0guessed2⟦lp1⟧) - Δ;
  If[r0guessedmin > tmp, r0guessedmin = tmp];
  r0guessed2⟦lp1⟧ = {r0 → tmp};
  lp1++];
]
r0guessedmin = {r0 → r0guessedmin};

Block[{str, type},
 natoms = Table[, {lp1, 1, nlat}];
 For[lp1 = 1, lp1 ≤ nlat, str = Tlat⟦lp1⟧;
  type = ToExpression["T" <> str]; natoms⟦lp1⟧ =
   Table[, {lp2, 1, ntypes}]; For[lp2 = 1, lp2 ≤ ntypes,
   natoms⟦lp1, lp2⟧ = Count[type, lp2]; lp2++]; lp1++];
]
Print["natoms ", natoms];
```

```
natoms {{4, 4}, {4, 8}}
```

## ■ Lattice Analysis

```
Block[{rlist, rlist1},
 zlat = Table[, {lp1, 1, nlat}];
 ratiolat = Table[, {lp1, 1, nlat}];
 For[lp1 = 1, lp1 ≤ nlat, zlat[[lp1]] = Table[, {lp2, 1, ntypes}];
  ratiolat[[lp1]] = Table[, {lp2, 1, ntypes}]; For[lp2 = 1,
   lp2 ≤ ntypes, zlat[[lp1, lp2]] = Table[, {lp3, 1, ntypes}];
   ratiolat[[lp1, lp2]] = Table[, {lp3, 1, ntypes}];
   lp2++]; lp1++];
 For[lp1 = 1, lp1 ≤ nlat, For[lp2 = 1, lp2 ≤ ntypes,
   For[lp3 = 1, lp3 ≤ ntypes, zlat[[lp1, lp2, lp3]] = {};
    rlist = findrlist[lp1, lp2, lp3];
    rlist = Round[rlist * 1 000 000] / 1 000 000;
    rlist1 = Sort[Union[rlist]];
    For[lp4 = 1, lp4 ≤ Length[rlist1], zlat[[lp1, lp2, lp3]] =
      Join[zlat[[lp1, lp2, lp3]], {Count[rlist, rlist1[[lp4]]]}];
     lp4++]; ratiolat[[lp1, lp2, lp3]] = rlist1 / (r0 /. r0guessedmin);
    Print["lattice ", lp1, " species ", lp2,
     " neighbors ", lp3]; Print[zlat[[lp1, lp2, lp3]]];
    Print[ratiolat[[lp1, lp2, lp3]]]; lp3++]; lp2++]; lp1++];
]
```

```
lattice 1 species 1 neighbors 1

{12, 6, 24, 12, 24, 8, 48, 6, 36, 24, 24, 24, 72, 48, 12, 48}

{1., 1.41421, 1.73205, 2., 2.23607, 2.44949, 2.64575, 2.82843,
 3., 3.16228, 3.31662, 3.4641, 3.60555, 3.87298, 4., 4.12311}

lattice 1 species 1 neighbors 2

{6, 8, 24, 30, 24, 24, 48, 24, 48, 30, 32, 72, 48, 48}

{0.707107, 1.22474, 1.58114, 2.12132, 2.34521, 2.54951, 2.91548,
 3.08221, 3.24037, 3.53553, 3.67423, 3.80789, 4.06202, 4.1833}

lattice 1 species 2 neighbors 1

{6, 8, 24, 30, 24, 24, 48, 24, 48, 30, 32, 72, 48, 48}

{0.707107, 1.22474, 1.58114, 2.12132, 2.34521, 2.54951, 2.91548,
 3.08221, 3.24037, 3.53553, 3.67423, 3.80789, 4.06202, 4.1833}

lattice 1 species 2 neighbors 2
```

```
{12, 6, 24, 12, 24, 8, 48, 6, 36, 24, 24, 24, 72, 48, 12, 48}

{1., 1.41421, 1.73205, 2., 2.23607, 2.44949, 2.64575, 2.82843,
 3., 3.16228, 3.31662, 3.4641, 3.60555, 3.87298, 4., 4.12311}

lattice 2 species 1 neighbors 1

{12, 6, 24, 12, 24, 8, 48, 6, 36, 24, 24, 24, 72, 48, 12, 48}

{1., 1.41421, 1.73205, 2., 2.23607, 2.44949, 2.64575, 2.82843,
 3., 3.16228, 3.31662, 3.4641, 3.60555, 3.87298, 4., 4.12311}

lattice 2 species 1 neighbors 2

{8, 24, 24, 32, 48, 24, 48, 72, 24, 56, 72, 48, 72, 72, 48, 48, 120, 72}

{0.612373, 1.1726, 1.5411, 1.83712, 2.09165, 2.3184,
 2.52488, 2.7157, 2.89396, 3.06186, 3.22102, 3.37268,
 3.51781, 3.65718, 3.79144, 3.9211, 4.0466, 4.16833}

lattice 2 species 2 neighbors 1

{4, 12, 12, 16, 24, 12, 24, 36, 12, 28, 36, 24, 36, 36, 24, 24, 60, 36}

{0.612373, 1.1726, 1.5411, 1.83712, 2.09165, 2.3184,
 2.52488, 2.7157, 2.89396, 3.06186, 3.22102, 3.37268,
 3.51781, 3.65718, 3.79144, 3.9211, 4.0466, 4.16833}

lattice 2 species 2 neighbors 2

{6, 12, 8, 6, 24, 24, 12, 30, 24, 24, 8, 24, 48, 6, 48,
 36, 24, 24, 48, 24, 24, 30, 72, 32, 72, 48, 12, 48, 48, 48}

{0.707107, 1., 1.22474, 1.41421, 1.58114, 1.73205, 2., 2.12132,
 2.23607, 2.34521, 2.44949, 2.54951, 2.64575, 2.82843, 2.91548,
 3., 3.08221, 3.16228, 3.24037, 3.31662, 3.4641, 3.53553,
 3.60555, 3.67423, 3.80789, 3.87298, 4., 4.06202, 4.12311, 4.1833}
```

# EAM Functions

## Initial parameters and initial EAM tables

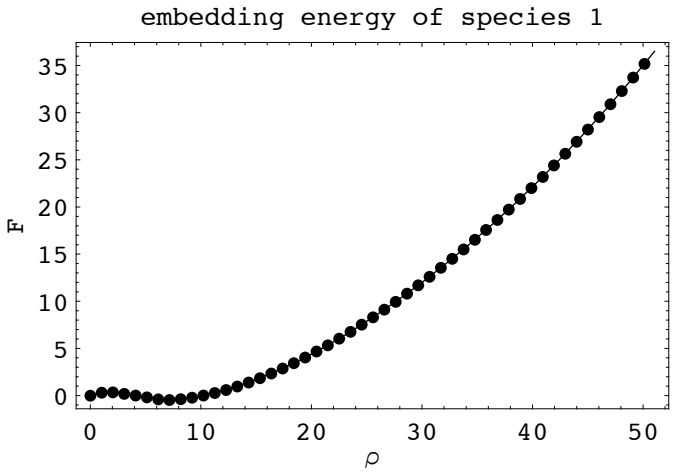
```
KtoeV = 0.0000861734315;
(*The following initialPara is from Josh.*)
initialPara =
  {D12 → 0.017074144766331022`, G12 → 2.125`, α12 → 5.5123948488`,
   β12 → 2.7561974244`, D22 → 0.9240795943015095`,
   G22 → 4.1797799285`, α22 → 1.57558370972`,
   β22 → 0.78779185486`, C2 → 1000.`, δ2 → 50.`,
   A2 → 0.024458327565237414`, B2 → -0.08529954759375455`};

initialPara = {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60,
   C2 → 11.003912875394953`, D12 → 0.234580672977548`,
   D22 → 0.07287155024003068`, G12 → 1.5125423949321948`,
   G22 → 2.4983485612790086`, α12 → 4.842354956626778`,
   α22 → 3.6749949240472164`, β12 → 2.1374260366912874`,
   β22 → 1.471049785271651`, δ2 → 1.323659492018399`};

Aϕm = Table[0, {lp, 1, npair}];
βϕm = Table[0, {lp, 1, npair}];
rϕm = Table[0, {lp, 1, npair}];
For[i = 1, i ≤ ntypes, shift_i = 0; i++];
nρ = Table[, {lp, 1, ntypes}];
dρ = Table[, {lp, 1, ntypes}];
nr = Table[, {lp, 1, ntypes}];
dr = Table[, {lp, 1, ntypes}];
fρ = Table[, {lp, 1, ntypes}];
ρlist = Table[, {lp, 1, ntypes}];
fρ1 = Table[, {lp, 1, ntypes}];
fρ2 = Table[, {lp, 1, ntypes}];
fρ3 = Table[, {lp, 1, ntypes}];
ρr = Table[, {lp, 1, ntypes}];
rlist = Table[, {lp, 1, ntypes}];
ρr1 = Table[, {lp, 1, ntypes}];
ρr2 = Table[, {lp, 1, ntypes}];
ρr3 = Table[, {lp, 1, ntypes}];
ϕr = Table[, {lp, 1, ntypes}];
ϕr1 = Table[, {lp, 1, ntypes}];
ϕr2 = Table[, {lp, 1, ntypes}];
ϕr3 = Table[, {lp, 1, ntypes}];

ReadRestart[i_, filename_String] :=
 Block[{dog, strpos, datapos, r},
  dog = Import[filename, "List"];
  strpos = Flatten[Position[dog, _String]];
  For[lp = 1,
   lp ≤ Length[strpos] && lp == strpos[[lp]], datapos = lp; lp++];
```

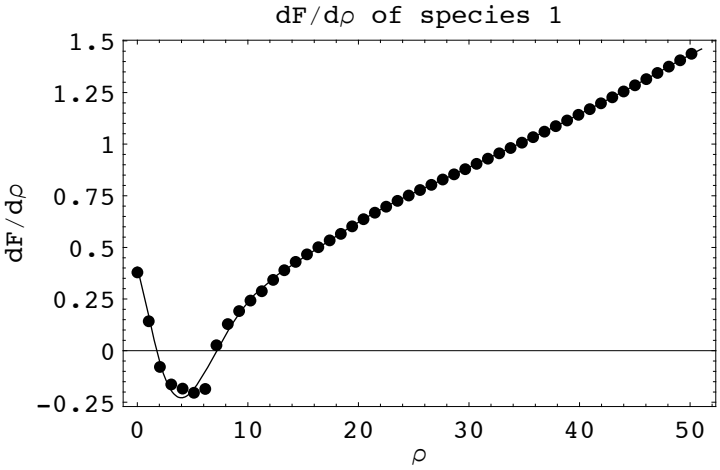
```
nρ[[i]] = Part[dog, datapos + 2];
dρ[[i]] = Part[dog, datapos + 3];
nr[[i]] = Part[dog, datapos + 4];
dr[[i]] = Part[dog, datapos + 5];
rcut[[i]] = Part[dog, datapos + 6];
datapos = datapos + 10;
fρ[[i]] = Table[Part[dog, datapos + lp], {lp, 1, nρ[[i]]}];
ρlist[[i]] = Table[(lp - 1) * dρ[[i]], {lp, 1, nρ[[i]]}];
fρ1[[i]] = Table[, {lp, 1, nρ[[i]]}];
fρ2[[i]] = Table[, {lp, 1, nρ[[i]]}];
fρ3[[i]] = Table[, {lp, 1, nρ[[i]]}];
datapos = datapos + nρ[[i]];
ρr[[i]] = Table[Part[dog, datapos + lp], {lp, 1, nr[[i]]}];
rlist[[i]] = Table[(lp - 1) * dr[[i]], {lp, 1, nr[[i]]}];
ρr1[[i]] = Table[, {lp, 1, nr[[i]]}];
ρr2[[i]] = Table[, {lp, 1, nr[[i]]}];
ρr3[[i]] = Table[, {lp, 1, nr[[i]]}];
datapos = datapos + nr[[i]];
ϕr[[i]] = Table[If[lp ≤ 11, r = 10 dr[[i]], r = (lp - 1) dr[[i]];
  Part[dog, datapos + lp] / r, {lp, 1, nr[[i]]}];
ϕr1[[i]] = Table[, {lp, 1, nr[[i]]}];
ϕr2[[i]] = Table[, {lp, 1, nr[[i]]}];
ϕr3[[i]] = Table[, {lp, 1, nr[[i]]}];
fρ1[[i, 1]] = fρ[[i, 2]] - fρ[[i, 1]];
fρ1[[i, 2]] = (fρ[[i, 3]] - fρ[[i, 1]])/2;
fρ1[[i, nρ[[i]] - 1]] = (fρ[[i, nρ[[i]]]] - fρ[[i, nρ[[i]] - 2]])/2;
fρ1[[i, nρ[[i]]]] = fρ[[i, nρ[[i]]]] - fρ[[i, nρ[[i]] - 1]];
ρr1[[i, 1]] = ρr[[i, 2]] - ρr[[i, 1]];
ρr1[[i, 2]] = (ρr[[i, 3]] - ρr[[i, 1]])/2;
ρr1[[i, nr[[i]] - 1]] = (ρr[[i, nr[[i]]]] - ρr[[i, nr[[i]] - 2]])/2;
ρr1[[i, nr[[i]]]] = ρr[[i, nr[[i]]]] - ρr[[i, nr[[i]] - 1]];
ϕr1[[i, 1]] = ϕr[[i, 2]] - ϕr[[i, 1]];
ϕr1[[i, 2]] = (ϕr[[i, 3]] - ϕr[[i, 1]])/2;
ϕr1[[i, nr[[i]] - 1]] = (ϕr[[i, nr[[i]]]] - ϕr[[i, nr[[i]] - 2]])/2;
ϕr1[[i, nr[[i]]]] = ϕr[[i, nr[[i]]]] - ϕr[[i, nr[[i]] - 1]];

For[lp = 3, lp ≤ nρ[[i]] - 2,
 fρ1[[i, lp]] = 1/12 (fρ[[i, lp - 2]] - fρ[[i, lp + 2]] +
     8 (fρ[[i, lp + 1]] - fρ[[i, lp - 1]])); lp++];
For[lp = 3, lp ≤ nr[[i]] - 2, ρr1[[i, lp]] = 1/12
```

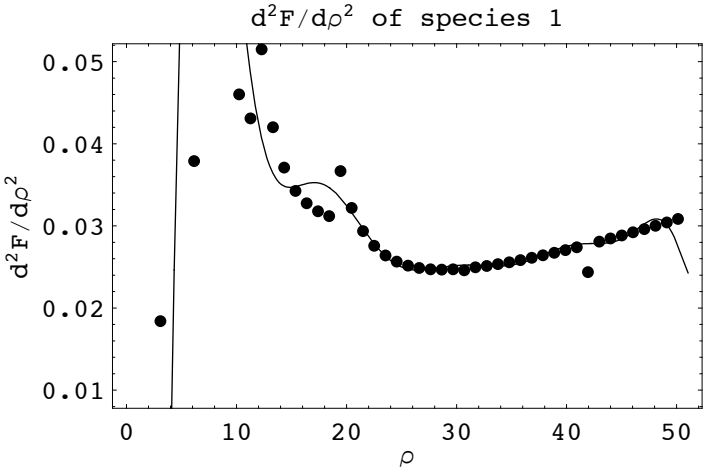
```
      (ρr⟦i, lp - 2⟧ - ρr⟦i, lp + 2⟧ + 8 (ρr⟦i, lp + 1⟧ - ρr⟦i, lp - 1⟧));
    ϕr1⟦i, lp⟧ = 1/12 (ϕr⟦i, lp - 2⟧ - ϕr⟦i, lp + 2⟧ +
        8 (ϕr⟦i, lp + 1⟧ - ϕr⟦i, lp - 1⟧)); lp++];
  For[lp = 1, lp ≤ nρ⟦i⟧ - 1, fρ2⟦i, lp⟧ =
    3 (fρ⟦i, lp + 1⟧ - fρ⟦i, lp⟧) - 2 fρ1⟦i, lp⟧ - fρ1⟦i, lp + 1⟧;
   fρ3⟦i, lp⟧ = fρ1⟦i, lp⟧ + fρ1⟦i, lp + 1⟧ -
     2 (fρ⟦i, lp + 1⟧ - fρ⟦i, lp⟧); lp++];
  For[lp = 1, lp ≤ nr⟦i⟧ - 1, ρr2⟦i, lp⟧ =
    3 (ρr⟦i, lp + 1⟧ - ρr⟦i, lp⟧) - 2 ρr1⟦i, lp⟧ - ρr1⟦i, lp + 1⟧;
   ρr3⟦i, lp⟧ = ρr1⟦i, lp⟧ + ρr1⟦i, lp + 1⟧ -
     2 (ρr⟦i, lp + 1⟧ - ρr⟦i, lp⟧); ϕr2⟦i, lp⟧ =
    3 (ϕr⟦i, lp + 1⟧ - ϕr⟦i, lp⟧) - 2 ϕr1⟦i, lp⟧ - ϕr1⟦i, lp + 1⟧;
   ϕr3⟦i, lp⟧ = ϕr1⟦i, lp⟧ + ϕr1⟦i, lp + 1⟧ -
     2 (ϕr⟦i, lp + 1⟧ - ϕr⟦i, lp⟧); lp++];
  fρ2⟦i, nρ⟦i⟧⟧ = 0;
  ρr2⟦i, nr⟦i⟧⟧ = 0;
  ϕr2⟦i, nr⟦i⟧⟧ = 0;
  fρ3⟦i, nρ⟦i⟧⟧ = 0;
  ρr3⟦i, nr⟦i⟧⟧ = 0;
  ϕr3⟦i, nr⟦i⟧⟧ = 0;
  ϕr2⟦i, 1⟧ = ϕr2⟦i, 2⟧;
 ]
ReadRestart[1, "files\pd1.set"];

FaT[i_, ρ_, vρ_] := Block[{ps, ks, tmp},
  ps = vρ / dρ⟦i⟧ + 1;
  ks = IntegerPart[ps];
  If[ks ≤ nρ⟦i⟧ - 1, ps = ρ / dρ⟦i⟧ + 1 - ks, ks = nρ⟦i⟧ - 1; ps = 1];
  tmp = PowerExpand[
    ((fρ3⟦i, ks⟧ ps + fρ2⟦i, ks⟧) ps + fρ1⟦i, ks⟧) ps + fρ⟦i, ks⟧];
  Return[tmp]]

ρaT[i_, r_, vr_] := Block[{ps, ks, tmp},
  ps = vr / dr⟦i⟧ + 1;
  ks = IntegerPart[ps];
  If[ks ≤ nr⟦i⟧ - 1, ps = r / dr⟦i⟧ + 1 - ks, ks = nr⟦i⟧ - 1; ps = 1];
  tmp = PowerExpand[
    ((ρr3⟦i, ks⟧ ps + ρr2⟦i, ks⟧) ps + ρr1⟦i, ks⟧) ps + ρr⟦i, ks⟧];
  Return[tmp]]

ϕT[i_, r_, vr_] := Block[{ps, ks, tmp},
  ps = vr / dr⟦i⟧ + 1;
  ks = IntegerPart[ps];
  If[ks ≤ nr⟦i⟧ - 1, ps = r / dr⟦i⟧ + 1 - ks, ks = nr⟦i⟧ - 1; ps = 1];
  tmp = PowerExpand[
    ((ϕr3⟦i, ks⟧ ps + ϕr2⟦i, ks⟧) ps + ϕr1⟦i, ks⟧) ps + ϕr⟦i, ks⟧];
  Return[tmp]]
```

## Fitting embedding energy table

```
FitEmb[i_, NpEmb_] := Block[{condition1,
   condition2, data, data1, Δn, p1, p2, deri1, deri2},
  condition1 = {a0 → 0, c0 → 0};
  condition2 = Table[cn, {n, 1, NpEmb}];
  data = Transpose[{ρlist[[i]], fρ[[i]]}];
  embcoei = FindFit[data,
    polynomial[ρ, NpEmb] /. condition1, condition2, ρ];
  embcoei = Join[condition1, embcoei];
  Δn = IntegerPart[nρ[[i]] / 50];
  data1 = Table[data[[lp]], {lp, 1, nρ[[i]], Δn}];
  p1 = Plot[polynomial[ρ, NpEmb] /. embcoei,
    {ρ, 0, (nρ[[i]] - 1) * dρ[[i]]}, DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"ρ", "F"},
   PlotLabel → "embedding energy of species " <> ToString[i]];
  deri1 = D[polynomial[ρ, NpEmb] /. embcoei, {ρ, 1}];
  For[lp = 1, lp ≤ Length[data1],
   data1[[lp, 2]] = D[FaT[i, ρ, data1[[lp, 1]]], {ρ, 1}] /.
     ρ → data1[[lp, 1]]; lp++];
  p1 = Plot[deri1, {ρ, 0, (nρ[[i]] - 1) * dρ[[i]]},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"ρ", "dF/dρ"},
   PlotLabel → "dF/dρ of species " <> ToString[i]];
  deri2 = D[polynomial[ρ, NpEmb] /. embcoei, {ρ, 2}];
  For[lp = 1, lp ≤ Length[data1],
   data1[[lp, 2]] = D[FaT[i, ρ, data1[[lp, 1]]], {ρ, 2}] /.
     ρ → data1[[lp, 1]]; lp++];
  p1 = Plot[deri2, {ρ, 0, (nρ[[i]] - 1) * dρ[[i]]},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"ρ", "d^2F/dρ^2"},
   PlotLabel → "d^2F/dρ^2 of species " <> ToString[i]];
 ]
NpEmb = 14;
Block[{j}, For[j = 1, j ≤ ntypes,
   If[Length[fρ[[j]]] > 10, FitEmb[j, NpEmb]]; j++];]
```


embedding energy of species 1
F
ρ

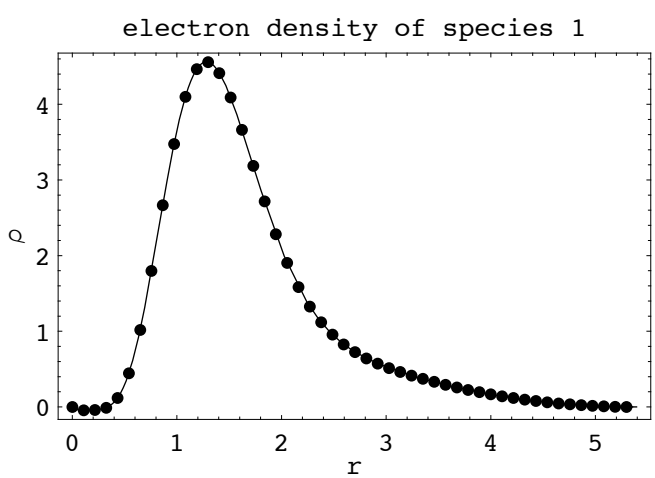

dF/dρ of species 1
dF/dρ
ρ

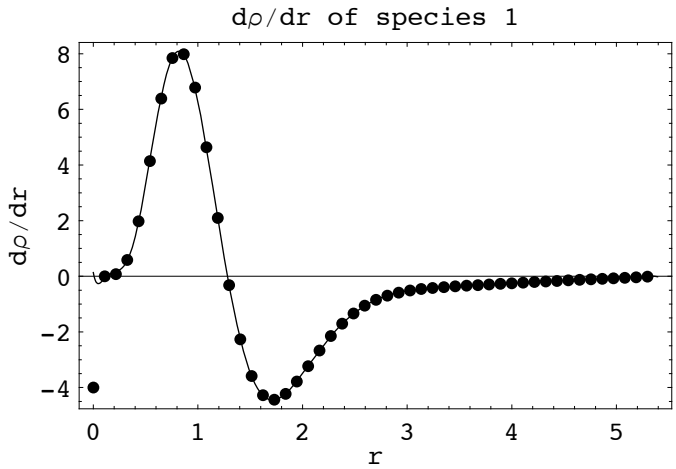

d²F/dρ² of species 1
d²F/dρ²
ρ

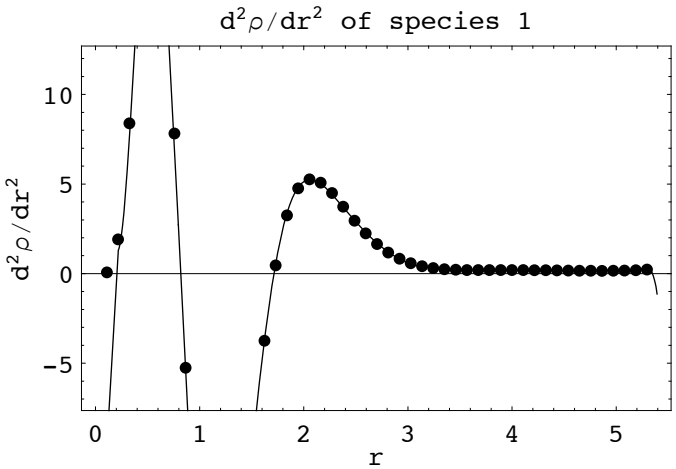
## Fitting electron density table

```
modρ[i_, datai_, rlb_] := Block[{ilat, tmp, tmp1, tmp2, datao},
  datao = datai;
  ilat = 0;
  For[lp = 1, lp ≤ Length[datai],
   If[datai[[lp, 1]] < rlb, ilat = lp]; lp++];
  If[ilat == 0, Return[datao]];
  tmp = ρaT[i, r, datai[[ilat, 1]]];
  tmp1 = D[tmp, {r, 1}];
  tmp2 = D[tmp, {r, 2}];
  tmp = tmp /. r → datai[[ilat, 1]];
  tmp1 = tmp1 /. r → datai[[ilat, 1]];
  tmp2 = tmp2 /. r → datai[[ilat, 1]];
  For[lp = 1, lp ≤ ilat,
   datao[[lp, 2]] = tmp + tmp1 * (datao[[lp, 1]] - datai[[ilat, 1]]) +
     tmp2 * (datao[[lp, 1]] - datai[[ilat, 1]])^2 / 2; lp++];
  Return[datao];
 ]

FitRho[i_, NpRho_] :=
 Block[{condition1, condition2, data, data1,
   data2, Δn, p1, p2, deri1, deri2, rtmp, tmp},
  condition1 = {a0 → rcut[[i]], c0 → 0};
  condition2 = Table[cn, {n, 1, NpRho}];
  data = Transpose[{rlist[[i]], ρr[[i]]}];
  Δn = IntegerPart[0 / dr[[i]] + 1;
  data1 = Table[data[[lp]], {lp, Δn, nr[[i]]}];
  data2 = modρ[i, data1, 0];
  rhocoei = FindFit[data2,
    polynomial[r, NpRho] /. condition1, condition2, r];
  rhocoei = Join[condition1, rhocoei];
  tmp = Table[rtmp = (lp - 1) * dr[[i]];
    polynomial[rtmp, NpRho] /. rhocoei, {lp, 1, nr[[i]]}];
  tmp = Min[tmp];
  If[1.1 tmp < ρshift, ρshift = 1.1 tmp];
  Δn = IntegerPart[nr[[i]] / 50];
  data1 = Table[data[[lp]], {lp, 1, nr[[i]], Δn}];
  p1 = Plot[polynomial[r, NpRho] /. rhocoei,
    {r, 0, (nr[[i]] - 1) * dr[[i]]}, DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "ρ"},
   PlotLabel → "electron density of species " <> ToString[i]];
  deri1 = D[polynomial[r, NpRho] /. rhocoei, {r, 1}];
```

```
  For[lp = 1, lp ≤ Length[data1],
   data1⟦lp, 2⟧ = D[ρaT[i, r, data1⟦lp, 1⟧], {r, 1}] /.
     r → data1⟦lp, 1⟧; lp++];
  p1 = Plot[deri1, {r, 0, (nr⟦i⟧ - 1) * dr⟦i⟧},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "dρ/dr"},
   PlotLabel → "dρ/dr of species " <> ToString[i]];
  deri2 = D[polynomial[r, NpRho] /. rhocoe_i, {r, 2}];
  For[lp = 1, lp ≤ Length[data1],
   data1⟦lp, 2⟧ = D[ρaT[i, r, data1⟦lp, 1⟧], {r, 2}] /.
     r → data1⟦lp, 1⟧; lp++];
  p1 = Plot[deri2, {r, 0, (nr⟦i⟧ - 1) * dr⟦i⟧},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "d^2ρ/dr^2"},
   PlotLabel → "d^2ρ/dr^2 of species " <> ToString[i]];
 ]
NpRho = 18;
ρshift = -10^-6;
Block[{j}, For[j = 1, j ≤ ntypes,
   If[Length[ρr⟦j⟧] > 10, FitRho[j, NpRho]]; j++];]
```

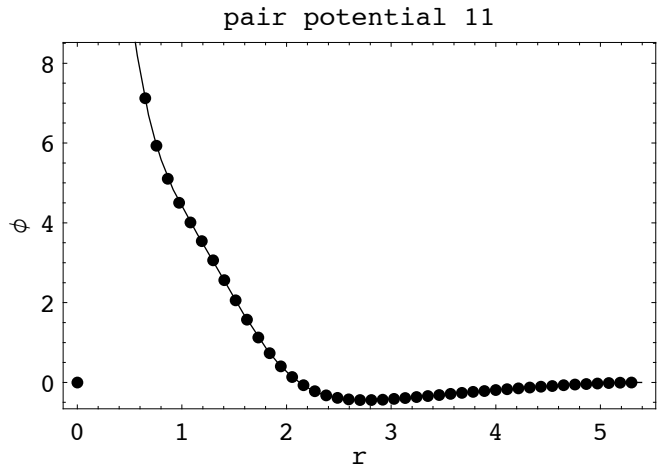


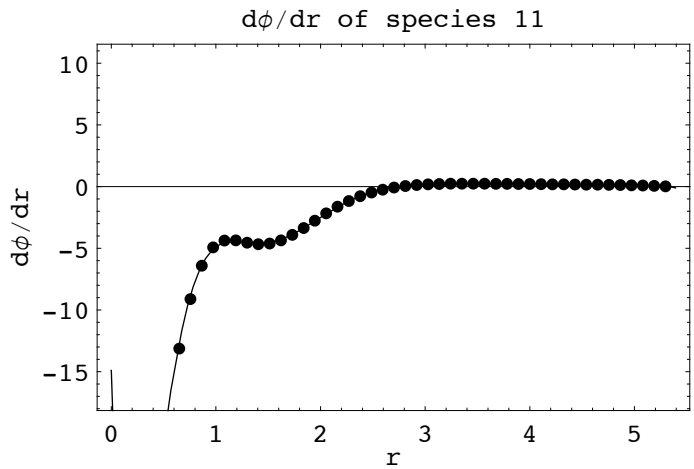

dρ/dr of species 1
dρ/dr
r
8
6
4
2
0
−2
−4
0
1
2
3
4
5

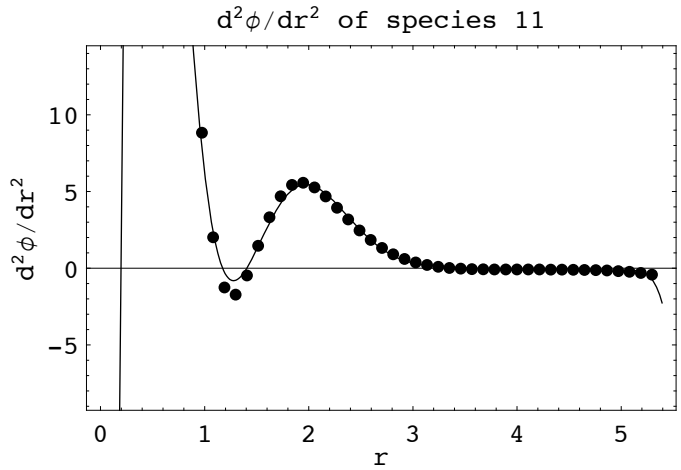

d²ρ/dr² of species 1
d²ρ/dr²
r
10
5
0
−5
0
1
2
3
4
5

## Fitting pair energy table

```
FitPhi[i_, NpPhi_] := Block[{condition1,
   condition2, data, data1, Δn, p1, p2, deri1, deri2},
  condition1 = {a0 → rcut⟦i⟧, c0 → 0};
  condition2 = Table[cn, {n, 1, NpPhi}];
  data = Transpose[{rlist⟦i⟧, ϕr⟦i⟧}];
  Δn = IntegerPart[1/(10 dr⟦i⟧)] + 1;
  data1 = Table[data⟦lp⟧, {lp, Δn, nr⟦i⟧}];
  phicoei = FindFit[data1,
    polynomial[r, NpPhi] /. condition1, condition2, r];
  phicoei = Join[condition1, phicoei];
  Δn = IntegerPart[nr⟦i⟧ / 50];
  data1 = Table[data⟦lp⟧, {lp, 1, nr⟦i⟧, Δn}];
  p1 = Plot[polynomial[r, NpPhi] /. phicoei,
    {r, 0, (nr⟦i⟧ - 1) * dr⟦i⟧}, DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "ϕ"}, PlotLabel →
    "pair potential " <> ToString[i] <> ToString[i]];
  deri1 = D[polynomial[r, NpPhi] /. phicoei, {r, 1}];
  For[lp = 1, lp ≤ Length[data1], data1⟦lp, 2⟧ =
    D[ϕT[i, r, data1⟦lp, 1⟧], {r, 1}] /. r → data1⟦lp, 1⟧; lp++];
  p1 = Plot[deri1, {r, 0, (nr⟦i⟧ - 1) * dr⟦i⟧},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "dϕ/dr"}, PlotLabel →
    "dϕ/dr of species " <> ToString[i] <> ToString[i]];
  deri2 = D[polynomial[r, NpPhi] /. phicoei, {r, 2}];
  For[lp = 1, lp ≤ Length[data1], data1⟦lp, 2⟧ =
    D[ϕT[i, r, data1⟦lp, 1⟧], {r, 2}] /. r → data1⟦lp, 1⟧; lp++];
  p1 = Plot[deri2, {r, 0, (nr⟦i⟧ - 1) * dr⟦i⟧},
    DisplayFunction → Identity];
  p2 = ListPlot[data1, PlotJoined → False, PlotStyle →
     {PointSize[0.02]}, DisplayFunction → Identity];
  Show[p1, p2, DisplayFunction → $DisplayFunction,
   Frame → True, FrameLabel → {"r", "d²ϕ/dr²"}, PlotLabel →
    "d²ϕ/dr² of species " <> ToString[i] <> ToString[i]];
 ]
NpPhi = 12;
Block[{j}, For[j = 1, j ≤ ntypes,
   If[Length[ϕr⟦j⟧] > 10, FitPhi[j, NpPhi]]; j++];]
```


pair potential 11
ϕ
r


dϕ/dr of species 11
dϕ/dr
r


d²ϕ/dr² of species 11
d²ϕ/dr²
r

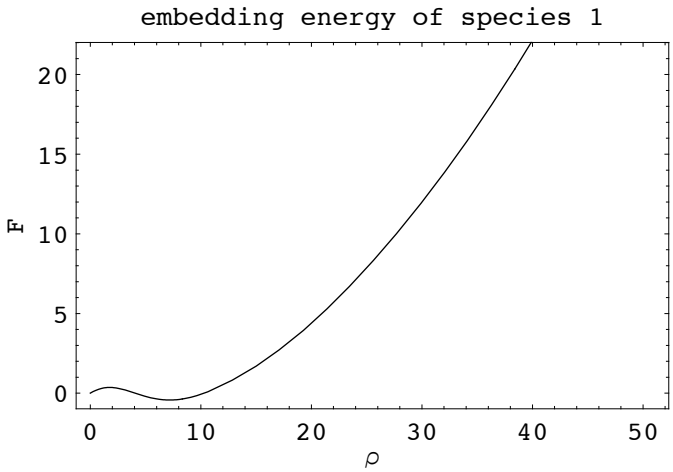
## Define EAM functions

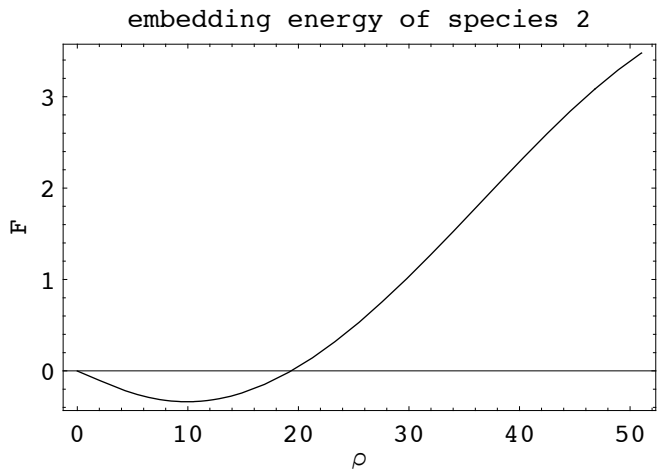
```
Fa[i_, ρ_] := Block[{str, Atmp, Btmp, ρxtmp, ρytmp, tmp},
  If[Length[fρ[[i]]] > 10,
    tmp =
     PowerExpand[(polynomial[ρ, NpEmb] /. embcoe_i) - shift_i * ρ];
    Return[tmp],
    str = ToString[i];
    Atmp = ToExpression["A" <> str];
    Btmp = ToExpression["B" <> str];
    ρxtmp = ToExpression["ρx" <> str];
    ρytmp = ToExpression["ρy" <> str];
    (*tmp=
      PowerExpand[ρ Atmp Log[ρ-ρshift]-shift_i*ρ+Btmp*ρ];*)
    tmp = PowerExpand[-Atmp * (1/(2 + Btmp) * (ρ - ρshift)^(2+Btmp) -
          (ρxtmp + ρytmp)/(1 + Btmp) * (ρ - ρshift)^(1+Btmp) +
          (ρxtmp * ρytmp)/Btmp * (ρ - ρshift)^Btmp) - shift_i * ρ];
    Return[tmp]];]

(*Will have UnitStep functions when vr is variable.*)
ρa[i_, r_, vr_] := Block[{str, Ctmp, δtmp, tmp, tmp1, vtmp1},
  If[Length[ρr[[i]]] > 10,
    tmp = PowerExpand[polynomial[r, NpRho] /. rhocoe_i];
    Return[tmp],
    str = ToString[i];
    Ctmp = ToExpression["C" <> str];
    δtmp = ToExpression["δ" <> str];
    tmp = PowerExpand[Ctmp Exp[-δtmp r]];
    tmp1 = r - rcut[[i]] + smooth[[i]];
    vtmp1 = vr - rcut[[i]] + smooth[[i]];
    tmp = UnitStep[vtmp1] * (1/2 + 1/2 * Cos[Pi * tmp1/smooth[[i]]]) * tmp +
      (1 - UnitStep[vtmp1]) * tmp;
    Return[tmp]];]

(*Will have nested UnitStep functions when vr is variable.*)
ϕ[i_, j_, r_, vr_] := Block[{str, Dtmp, Gtmp, αtmp,
   βtmp, tmp, tmp1, vtmp1, tmp2, vtmp2, tmp3, ij},
  If[i == j && Length[ϕr[[i]]] > 10,
    tmp = PowerExpand[(polynomial[r, NpPhi] /. phicoe_i) +
       2 * shift_i * (polynomial[r, NpRho] /. rhocoe_i)];
```

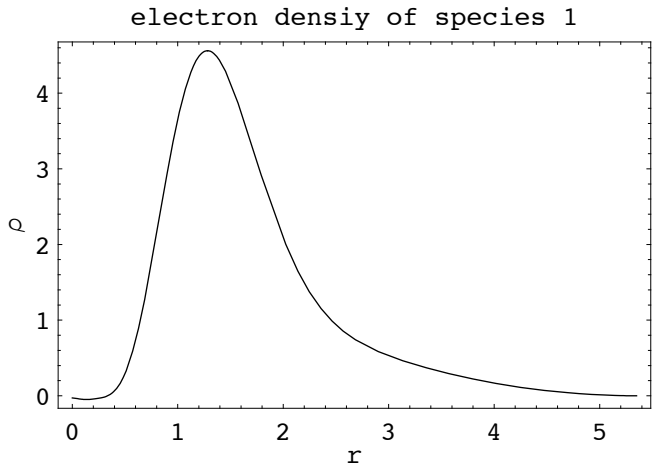
```
    Return[tmp],
    If[i ≤ j, str = ToString[i] <> ToString[j],
     str = ToString[j] <> ToString[i]];
    If[i == j, ij = i, If[i < j, ij = i * ntypes - (i * (i + 1))/2 + j,
       ij = j * ntypes - (j * (j + 1))/2 + i];];
    Dtmp = ToExpression["D" <> str];
    Gtmp = ToExpression["G" <> str];
    αtmp = ToExpression["α" <> str];
    βtmp = ToExpression["β" <> str];
    tmp = PowerExpand[βtmp Dtmp Exp[-αtmp (r - Gtmp)] -
       αtmp Dtmp Exp[-βtmp (r - Gtmp)]];
    tmp1 = r - rcut[[ij]] + smooth[[ij]];
    vtmp1 = vr - rcut[[ij]] + smooth[[ij]];
    tmp2 = r - rϕm[[ij]];
    vtmp2 = vr - rϕm[[ij]];
    tmp3 = Aϕm[[ij]] * Exp[-βϕm[[ij]] * r];
    tmp = (1 - UnitStep[vtmp2]) * tmp3 +
      UnitStep[vtmp2] * (1 - UnitStep[vtmp1]) * tmp +
      UnitStep[vtmp1] * (1/2 + 1/2 * Cos[(Pi * tmp1)/smooth[[ij]]]) * tmp;
    Return[tmp]];

(*This is an added function to modify the fitted pair
 function so that it is repulsive at short distances.*)
modϕ[i_, j_, rlb_, inparam_] := Block[{ij, drlb, tmp, tmp1},
  If[i == j, ij = i, If[i < j, ij = i * ntypes - (i * (i + 1))/2 + j,
     ij = j * ntypes - (j * (j + 1))/2 + i];];
  rϕm[[ij]] = 0;
  Aϕm[[ij]] = 0;
  βϕm[[ij]] = 0;
  If[rlb == 0 || rlb == 0., Return[]];
  drlb = 10^-8;
  tmp = ϕ[i, j, rlb + drlb, rlb + drlb] /. inparam;
  tmp1 = ϕ[i, j, rlb + 2 * drlb, 2 * drlb] /. inparam;
  tmp1 = (tmp1 - tmp)/drlb;
  rϕm[[ij]] = rlb;
  βϕm[[ij]] = -tmp1 / tmp;
  Aϕm[[ij]] = tmp * Exp[βϕm[[ij]] * rϕm[[ij]]];
 ]
```

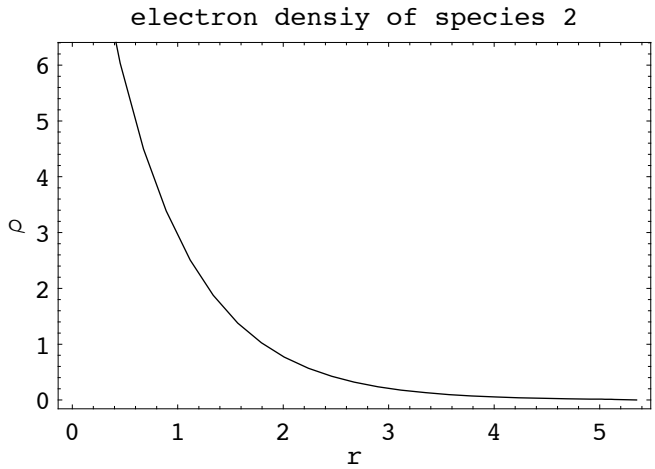
- **Plot EAM functions**

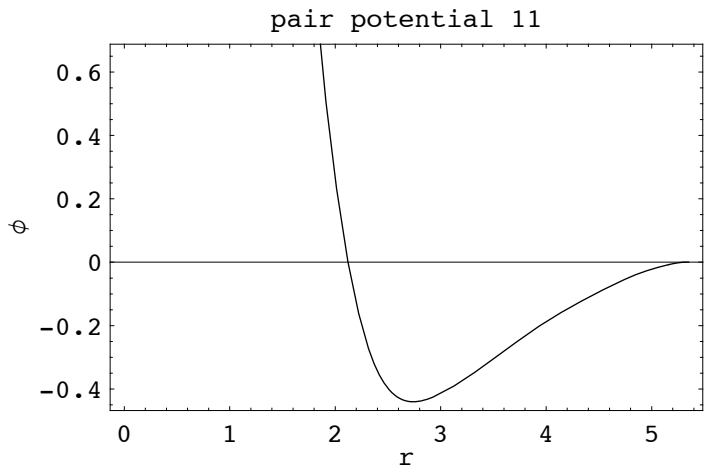
```
(*The plots use Fatmp, ρatmp, and ϕtmp.*)
pFa[i_] := Block[{tmp}, ρmax = 0; For[lp = 1, lp ≤ ntypes,
   If[Length[fρ⟦lp⟧] > 10, tmp = (nρ⟦lp⟧ - 1) dρ⟦lp⟧;
    If[ρmax < tmp, ρmax = tmp]]; lp++]; If[ρmax == 0, ρmax = 0.5];
  Plot[Fatmp⟦i⟧, {ρtmp, 0, ρmax}, Frame → True,
   FrameLabel → {"ρ", "F"},
   PlotLabel → "embedding energy of species " <> ToString[i]]]

pρa[i_] := Block[{rmax}, rmax = rcut⟦i⟧;
  Plot[ρatmp⟦i⟧ /. vrtmp → rtmp,
   {rtmp, 0, rmax}, Frame → True, FrameLabel → {"r", "ρ"},
   PlotLabel → "electron densiy of species " <> ToString[i]]]

pϕ[i_, j_] := Block[{ij, rmin, rmax, str},
  If[i == j, ij = i, If[i < j, ij = i * ntypes - (i * (i + 1))/2 + j,
     ij = j * ntypes - (j * (j + 1))/2 + i];]; rmax = rcut⟦ij⟧;
  For[rmin = 0, (ϕtmp⟦i, j⟧ /. {vrtmp → rmin, rtmp → rmin}) > 10,
   rmin = rmin + 0.01];
  If[i ≤ j, str = ToString[i] <> ToString[j],
   str = ToString[j] <> ToString[i]];
  Plot[ϕtmp⟦i, j⟧ /. vrtmp → rtmp, {rtmp, rmin, rmax},
   Frame → True, FrameLabel → {"r", "ϕ"},
   PlotLabel → "pair potential " <> str]]

outputEAM[initialPara_] := Block[{}, quickFunc[initialPara];
  For[lp1 = 1, lp1 ≤ ntypes, pFa[lp1]; lp1++];
  For[lp1 = 1, lp1 ≤ ntypes, pρa[lp1]; lp1++];
  For[lp1 = 1, lp1 ≤ ntypes,
   For[lp2 = lp1, lp2 ≤ ntypes, pϕ[lp1, lp2]; lp2++]; lp1++];]

For[n = 1, n ≤ ntypes, shift_n = 0; n++];
outputEAM[initialPara];
normalize[initialPara];
Print["Normalized EAM Functions **************"];
outputEAM[initialPara];
Print["*************************************"];
```

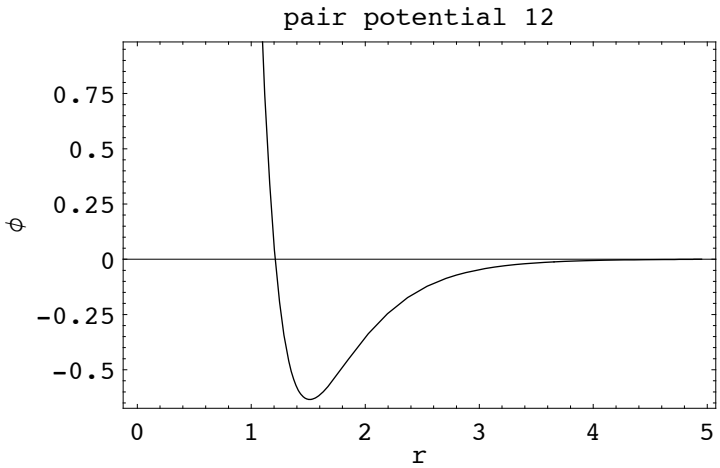

embedding energy of species 1
F
ρ
0
5
10
15
20
0
10
20
30
40
50

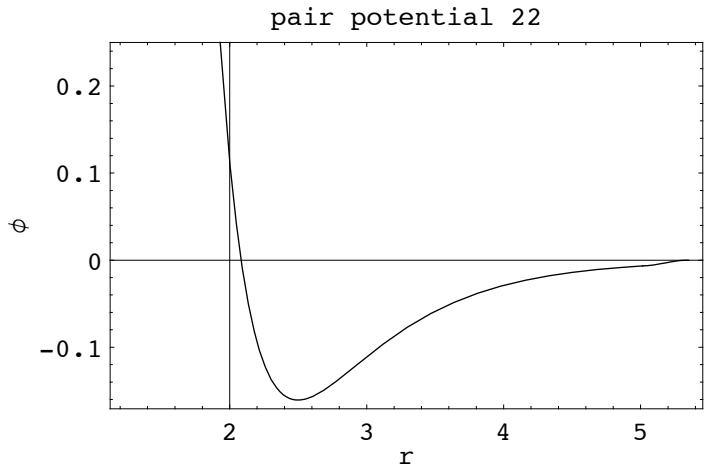

embedding energy of species 2
F
ρ
0
1
2
3
0
10
20
30
40
50


electron densiy of species 1
ρ
r
0
1
2
3
4
0
1
2
3
4
5

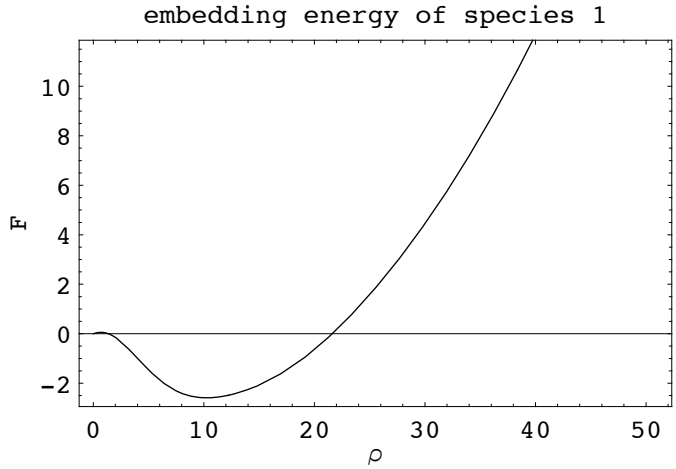

electron densiy of species 2
ρ
r

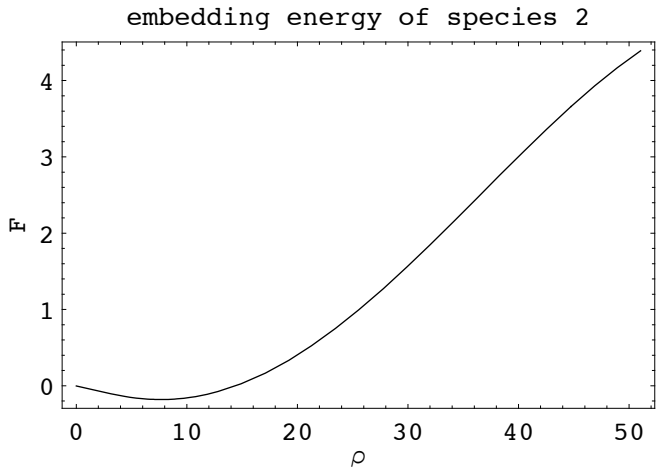

pair potential 11
ϕ
r

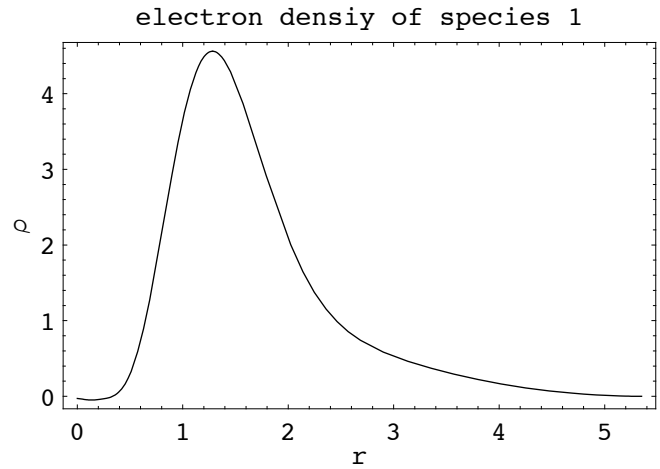

pair potential 12
ϕ
r

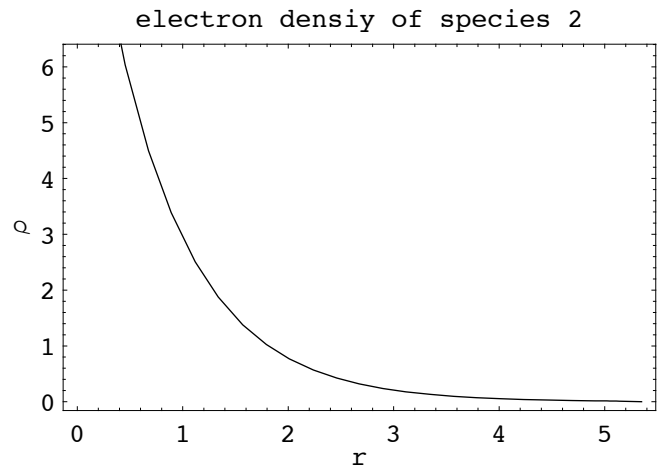


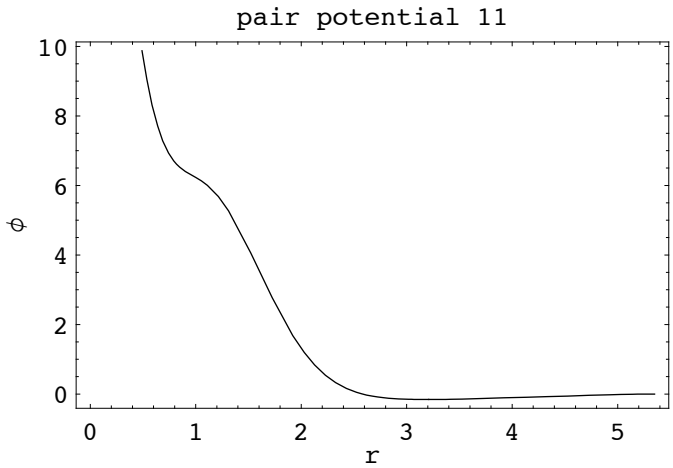

Normalized EAM Functions **********

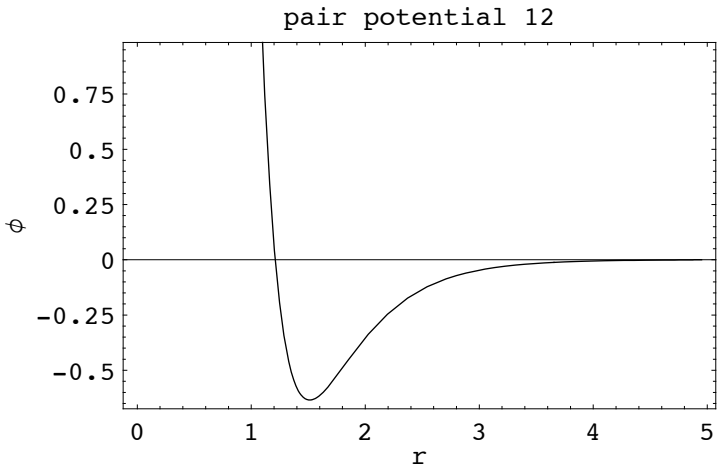


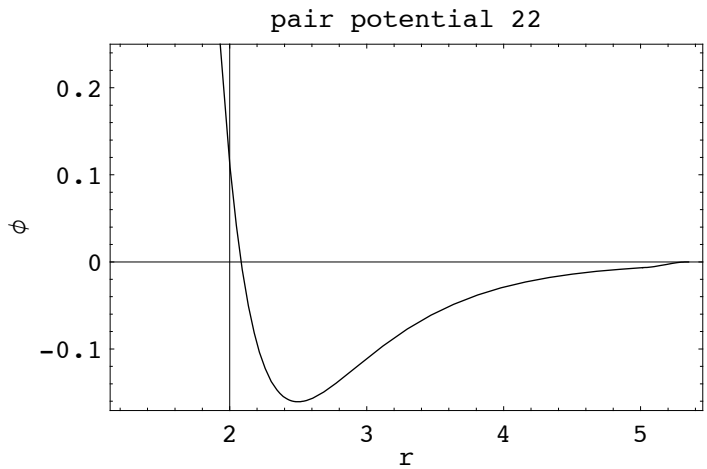




electron densiy of species 1
ρ
r
0
1
2
3
4
5


electron densiy of species 2
ρ
r
0
1
2
3
4
5
6


pair potential 11
ϕ
r
0
2
4
6
8
10

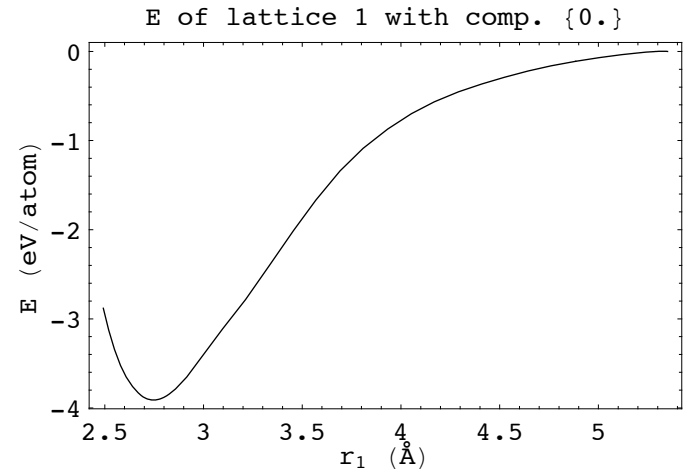


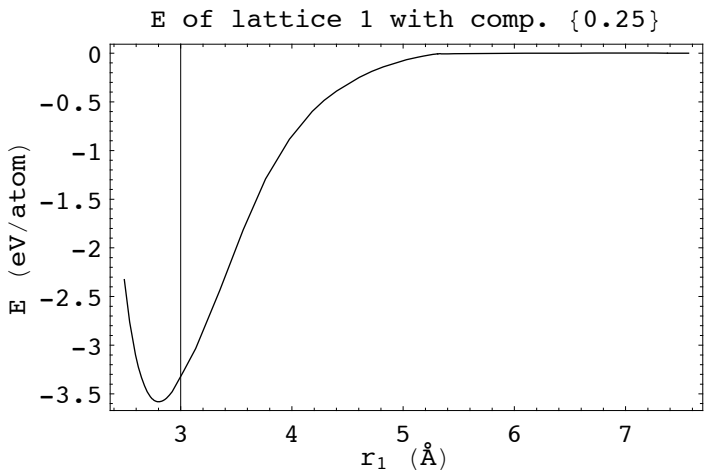



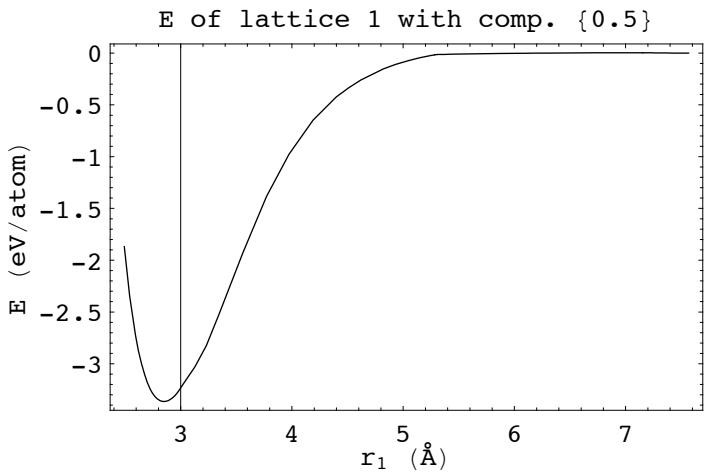

**********

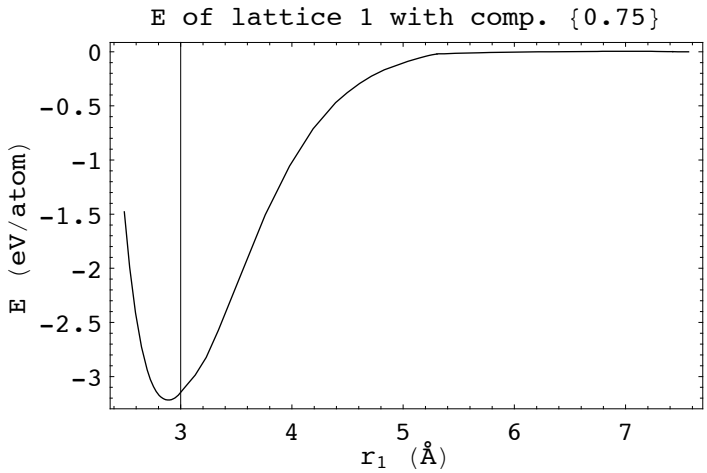
## ■ Print Initial Cohesive Energy

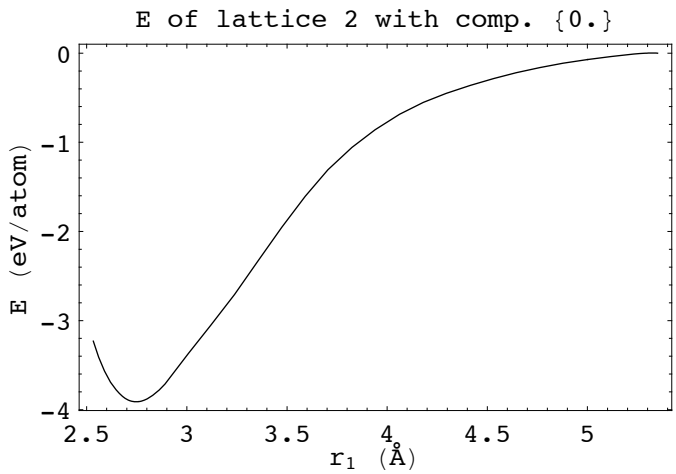
```
pEcohall[initialPara, 1];
```

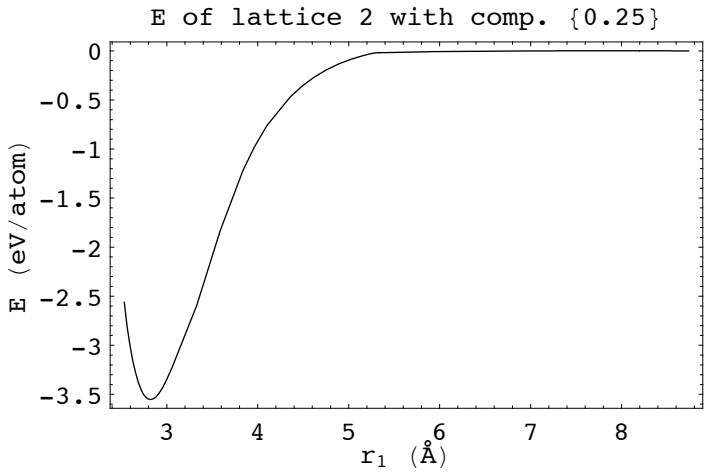

E of lattice 1 with comp. {0.}
E (eV/atom)
0
-1
-2
-3
-4
2.5
3
3.5
4
4.5
5
r1 (Å)

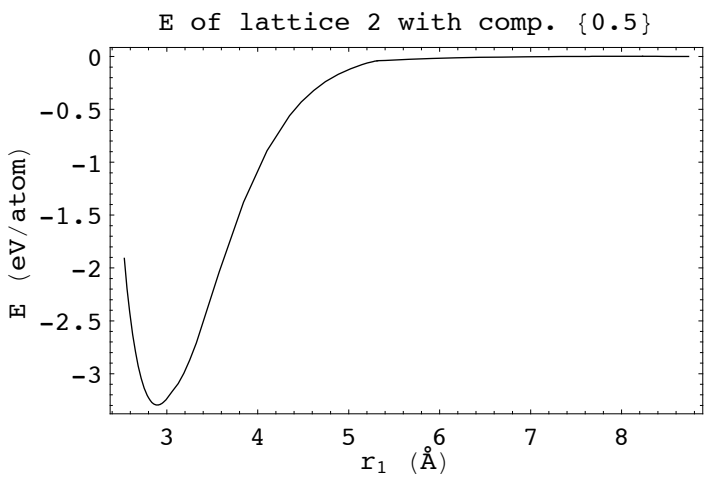

E of lattice 1 with comp. {0.25}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
3
4
5
6
7
r1 (Å)

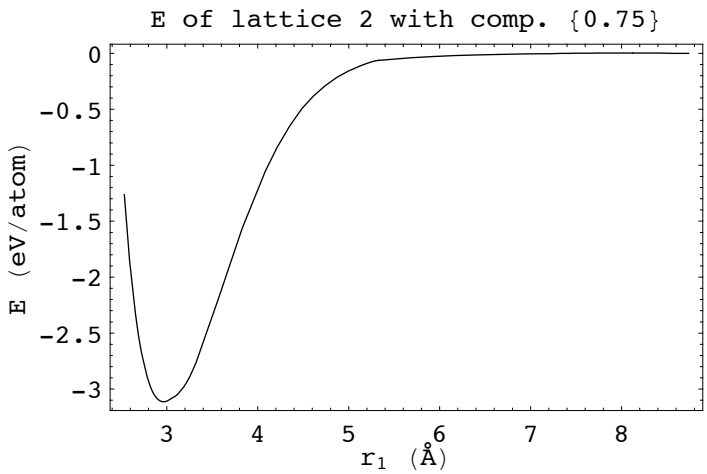

E of lattice 1 with comp. {0.5}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
r1 (Å)


E of lattice 1 with comp. {0.75}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
r1 (Å)


E of lattice 2 with comp. {0.}
E (eV/atom)
0
-1
-2
-3
-4
2.5
3
3.5
4
4.5
5
r1 (Å)


E of lattice 2 with comp. {0.25}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
3
4
5
6
7
8
r1 (Å)


E of lattice 2 with comp. {0.5}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
8
r1 (Å)


E of lattice 2 with comp. {0.75}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
8
r1 (Å)

# Fitting ...

## Set up varaibles

```
D12 =.;
G12 =.;
α12 =.;
β12 =.;
D22 =.;
G22 =.;
α22 =.;
β22 =.;
C2 =.;
δ2 =.;
A2 =.;
B2 =.;
ρx2 =.;
ρy2 =.;
myvar = Variables[{D12, G12, α12, β12,
    D22, G22, α22, β22, C2, δ2, A2, B2, ρx2, ρy2}];
myvar

normalize[all → all];
```

```
{A2, B2, C2, D12, D22, G12, G22, α12, α22, β12, β22, δ2, ρx2, ρy2}
```

## Input properties

```
(* "Structure Name" "Composition" "Cohesive energy"
 "Bulk modulus" "Bond Length" "C prime" "C44" *)
(*cohesive energy of H prior to adjustment
  using experimental calibration is -2.119 eV*)
EABfit =
```

$$\left(\begin{array}{lllll}
\text{"OC"} & \{1.0, 0.25\} & \{-3.64024, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{3.9421}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 0.50\} & \{-3.48770, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.0007}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 0.75\} & \{-3.37031, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.0593}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 1.00\} & \{-3.26966, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.1179}{\sqrt{2}}\right\} \{ \\
\text{"TE"} & \{1.0, 0.50\} & \{-3.50000, \text{">"}\} & \{1.050, \text{"="}\} & \left\{r0 \to \frac{4.0007}{\sqrt{2}}\right\} \{ \\
\text{"TE"} & \{1.0, 0.75\} & \{-3.50000, \text{">"}\} & \{1.050, \text{"="}\} & \left\{r0 \to \frac{4.0593}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{0.0, 1.00\} & \{-2.11900, \text{"="}\} & \{0.500, \text{">"}\} & \left\{r0 \to \frac{3.3800}{\sqrt{2}}\right\} \{
\end{array}\right)$$

```
;
```

```
EABfit =
```

$$\left(\begin{array}{lllll}
\text{"OC"} & \{1.0, 0.25\} & \{-3.56680, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{3.9421}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 0.50\} & \{-3.46770, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.0007}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 0.75\} & \{-3.41260, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.0593}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{1.0, 1.00\} & \{-3.38070, \text{"="}\} & \{1.050, \text{">"}\} & \left\{r0 \to \frac{4.1179}{\sqrt{2}}\right\} \{ \\
\text{"TE"} & \{1.0, 0.50\} & \{-3.50000, \text{">"}\} & \{1.050, \text{"="}\} & \left\{r0 \to \frac{4.0007}{\sqrt{2}}\right\} \{ \\
\text{"TE"} & \{1.0, 0.75\} & \{-3.50000, \text{">"}\} & \{1.050, \text{"="}\} & \left\{r0 \to \frac{4.0593}{\sqrt{2}}\right\} \{ \\
\text{"OC"} & \{0.0, 1.00\} & \{-2.11900, \text{"="}\} & \{0.500, \text{">"}\} & \left\{r0 \to \frac{3.3800}{\sqrt{2}}\right\} \{
\end{array}\right)$$

```
    )
    ;
    )

StructI = Column[EABfit, 1];
CompI = Column[EABfit, 2];
conRateEcI = Column[EABfit, 3];
conRateBI = Column[EABfit, 4];
r0I = Column[EABfit, 5];
conRateCprimeI = Column[EABfit, 6];
conRateC44I = Column[EABfit, 7];
PressI = Table[0, {lp, 1, Length[StructI]}];
EcI = Column[conRateEcI, 1];
conRateEcI = Column[conRateEcI, 2];
BI = Column[conRateBI, 1];
conRateBI = Column[conRateBI, 2];
CprimeI = Column[conRateCprimeI, 1];
conRateCprimeI = Column[conRateCprimeI, 2];
C44I = Column[conRateC44I, 1];
conRateC44I = Column[conRateC44I, 2];

For[lp = 1, lp ≤ Length[conRateEcI],
  If[conRateEcI[[lp]] == "=",
   conRateEcI[[lp]] = 0, If[conRateEcI[[lp]] == ">",
    conRateEcI[[lp]] = -20, conRateEcI[[lp]] = 20]];
  If[conRateBI[[lp]] == "=", conRateBI[[lp]] = 0,
   If[conRateBI[[lp]] == ">",
    conRateBI[[lp]] = -20, conRateBI[[lp]] = 20]];
  If[conRateCprimeI[[lp]] == "=", conRateCprimeI[[lp]] = 0,
   If[conRateCprimeI[[lp]] == ">",
    conRateCprimeI[[lp]] = -20, conRateCprimeI[[lp]] = 20]];
  If[conRateC44I[[lp]] == "=", conRateC44I[[lp]] = 0,
   If[conRateC44I[[lp]] == ">",
    conRateC44I[[lp]] = -20, conRateC44I[[lp]] = 20]];
  lp++];
```

```
Ewght = {{0}, {0}, {0}, {2}, {0}, {0}, {2}} // Flatten;

Bwght = {{1}, {1}, {1}, {1}, {0}, {0}, {1}} // Flatten;

Pwght = {{2}, {2}, {2}, {2}, {1}, {1}, {2}} // Flatten;

Cprimewght = {{1}, {1}, {1}, {1}, {0.5}, {0.5}, {1}} // Flatten;

C44wght = {{2}, {2}, {2}, {2}, {0.5}, {0.5}, {1}} // Flatten;

HOMfit = {{0.12, "="}, {0.02, "="}, {-0.08, "="}};

diffEfit = {{0.2, ">"}, {0.3, ">"}};

HOMI = Column[HOMfit, 1];
```

```
conRateHOMI = Column[HOMfit, 2];
diffEI = Column[diffEfit, 1];
conRatediffEI = Column[diffEfit, 2];

For[lp = 1, lp ≤ Length[conRateHOMI],
  If[conRateHOMI[[lp]] == "=",
   conRateHOMI[[lp]] = 0, If[conRateHOMI[[lp]] == ">",
    conRateHOMI[[lp]] = -20, conRateHOMI[[lp]] = 20]];
  lp++];
For[lp = 1, lp ≤ Length[conRatediffEI],
  If[conRatediffEI[[lp]] == "=",
   conRatediffEI[[lp]] = 0, If[conRatediffEI[[lp]] == ">",
    conRatediffEI[[lp]] = -20, conRatediffEI[[lp]] = 20]];
  lp++];
```

$$\text{HOMwght} = \begin{pmatrix} 16 \\ 16 \\ 16 \end{pmatrix} \text{ // Flatten;}$$

$$\text{diffEwght} = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \text{ // Flatten;}$$

Null[34]

## Analytical equations for EBA

```
Block[{ilat, tmp, comp, r1, vr1},
 If[Exist == 1,
   Eccal = Get["files\Eccal.m"];,
   Eccal = Table[
     For[lp2 = 1, lp2 ≤ nlat,
      If[StructI⟦lp1⟧ == Tlat⟦lp2⟧, ilat = lp2]; lp2++];
     comp = cvtC[CompI⟦lp1⟧, StructI⟦lp1⟧];
     vr1 = r0 /. r0I⟦lp1⟧;
     tmp = First[Ecoh[ilat, comp, r1, vr1]];
     tmp = tmp /. r1 → vr1;
     PowerExpand[tmp],
     {lp1, 1, Length[StructI]}];
   DeleteFile["files\Eccal.m"];
   Save["files\Eccal.m", Eccal];
  ];
]

Block[{},
 If[Exist == 1,
   HOMcal = Get["files\HOMcal.m"];,
   HOMcal = Table[
     PowerExpand[1/(1 + CompI⟦i, 2⟧) ((1 + CompI⟦i, 2⟧) * Eccal⟦i⟧ -
        2 * CompI⟦i, 2⟧ * Eccal⟦4⟧ - (1 - CompI⟦i, 2⟧) * (-3.91))],
     {lp1, 1, 3}];
   DeleteFile["files\HOMcal.m"];
   Save["files\HOMcal.m", HOMcal];
  ];
]

Block[{},
 If[Exist == 1,
   diffEcal = Get["files\diffEcal.m"];,
   diffEcal = {PowerExpand[Eccal⟦5⟧ - Eccal⟦2⟧],
     PowerExpand[Eccal⟦6⟧ - Eccal⟦3⟧]};
   DeleteFile["files\diffEcal.m"];
   Save["files\diffEcal.m", diffEcal];
  ];
]

Block[{ilat, tmp, comp, r1, vr1, vol},
 If[Exist == 1,
```

```
    Bcal = Get["files\Bcal.m"];,
    Bcal = Table[
      For[lp2 = 1, lp2 ≤ nlat,
       If[StructI[[lp1]] == Tlat[[lp2]], ilat = lp2]; lp2++];
      comp = cvtC[CompI[[lp1]], StructI[[lp1]]];
      vr1 = r0 /. r0I[[lp1]];
      tmp = First[Ecoh[ilat, comp, r1, vr1]];
      tmp = D[tmp, {r1, 2}];
      tmp = tmp /. r1 → vr1;
      vol = ToExpression["A" <> StructI[[lp1]] /. r0I[[lp1]];
      vol = vol^3 / (Tr[natoms[[ilat]] * comp]);
      PowerExpand[(vr1^2 * tmp) / (9 * vol)],
      {lp1, 1, Length[StructI]}];
    DeleteFile["files\Bcal.m"];
    Save["files\Bcal.m", Bcal];
   ];
]

Block[{ilat, tmp, comp, r1, vr1, vol},
 If[Exist == 1,
    Presscal = Get["files\Presscal.m"];,
    Presscal = Table[
      For[lp2 = 1, lp2 ≤ nlat,
       If[StructI[[lp1]] == Tlat[[lp2]], ilat = lp2]; lp2++];
      comp = cvtC[CompI[[lp1]], StructI[[lp1]]];
      vr1 = r0 /. r0I[[lp1]];
      tmp = First[Ecoh[ilat, comp, r1, vr1]];
      tmp = D[tmp, {r1, 1}];
      tmp = -tmp /. r1 → vr1;
      vol = ToExpression["A" <> StructI[[lp1]] /. r0I[[lp1]];
      vol = vol^3 / (Tr[natoms[[ilat]] * comp]);
      PowerExpand[(vr1 * tmp) / (3 * vol)],
      {lp1, 1, Length[StructI]}];
    DeleteFile["files\Presscal.m"];
    Save["files\Presscal.m", Presscal];
   ];
]
```

## Analytical equations for elastic constants

```
Block[{alat, type, pos, comp, lp},
 If[Exist == 0,
   GEc = Table[
      alat = ToExpression["A" <> StructI⟦lp⟧] /. r0I⟦lp⟧;
      type = ToExpression["T" <> StructI⟦lp⟧];
      pos = ToExpression["P" <> StructI⟦lp⟧] /. r0I⟦lp⟧;
      comp = cvtC[CompI⟦lp⟧, StructI⟦lp⟧];
      GEcAna[alat, pos, type, comp],
      {lp, 1, Length[StructI]}];
  ];
]

Block[{},
 If[Exist == 1,
   Cprimecal = Get["files\Cprimecal.m"];,
   Cprimecal =
    Table[Celas["prime", lp], {lp, 1, Length[StructI]}];
   DeleteFile["files\Cprimecal.m"];
   Save["files\Cprimecal.m", Cprimecal];
  ];
]

Block[{},
 If[Exist == 1,
   C44cal = Get["files\C44cal.m"];,
   C44cal = Table[Celas["44", lp], {lp, 1, Length[StructI]}];
   DeleteFile["files\C44cal.m"];
   Save["files\C44cal.m", C44cal];
  ];
]

Clear[GEc];
```

## Quick test

```
mychk[initialPara, {}];
```

```
check cohesive energy:

OC    { 1.00,  0.25}      -3.5668      -3.5791       0.0000
OC    { 1.00,  0.50}      -3.4677      -3.3615       0.0000
OC    { 1.00,  0.75}      -3.4126      -3.2148       0.0000
OC    { 1.00,  1.00}      -3.3807      -3.1141       2.0000
TE    { 1.00,  0.50}      -3.5000      -3.2644       0.0000
TE    { 1.00,  0.75}      -3.5000      -3.0608       0.0000
OC    { 0.00,  1.00}      -2.1190      -1.6540       2.0000

check bulk modulus:

OC    { 1.00,  0.25}       1.0500       1.2462       1.0000
OC    { 1.00,  0.50}       1.0500       1.1901       1.0000
OC    { 1.00,  0.75}       1.0500       1.1653       1.0000
OC    { 1.00,  1.00}       1.0500       1.1554       1.0000
TE    { 1.00,  0.50}       1.0500       1.3805       0.0000
TE    { 1.00,  0.75}       1.0500       1.5903       0.0000
OC    { 0.00,  1.00}       0.5000      -0.4220       1.0000

check pressure:

OC    { 1.00,  0.25}       0.0000       0.0190       2.0000
OC    { 1.00,  0.50}       0.0000       0.0247       2.0000
OC    { 1.00,  0.75}       0.0000       0.0214       2.0000
OC    { 1.00,  1.00}       0.0000       0.0118       2.0000
TE    { 1.00,  0.50}       0.0000       0.0874       1.0000
TE    { 1.00,  0.75}       0.0000       0.1215       1.0000
OC    { 0.00,  1.00}       0.0000      -0.0458       2.0000

check C':

OC    { 1.00,  0.25}       0.0550       0.1216       1.0000
OC    { 1.00,  0.50}       0.1350       0.1076       1.0000
OC    { 1.00,  0.75}       0.1640       0.0949       1.0000
OC    { 1.00,  1.00}       0.1460       0.0795       1.0000
TE    { 1.00,  0.50}       0.1630       0.2710       0.5000
TE    { 1.00,  0.75}       0.1630       0.5369       0.5000
OC    { 0.00,  1.00}       0.0500      -0.8701       1.0000

check C44':

OC    { 1.00,  0.25}       0.4590       0.3144       2.0000
OC    { 1.00,  0.50}       0.4200       0.2890       2.0000
OC    { 1.00,  0.75}       0.3820       0.2603       2.0000
OC    { 1.00,  1.00}       0.3440       0.2212       2.0000
TE    { 1.00,  0.50}       0.3810       0.3257       0.5000
TE    { 1.00,  0.75}       0.3810       0.3561       0.5000
OC    { 0.00,  1.00}       0.0500       0.1062       1.0000
```

## Fit setups

```
vcon = 0.05;
ρlow2 = 0.3;
ρhigh2 = 0.8;
tmp-fit =
  Length[EcI]
      ∑          Ewght[[i]] * smoothStep1[conRateEcI[[i]], EcI[[i]],
     i=1
    Eccal[[i]]] * (Eccal[[i]] - EcI[[i]])^2 +
   Length[BI]
      ∑          Bwght[[i]] * smoothStep1[conRateBI[[i]],
     i=1
    BI[[i]], Bcal[[i]]] * (Bcal[[i]] - BI[[i]])^2 +
   Length[Presscal]
      ∑          Pwght[[i]] * (Presscal[[i]])^2 +
     i=1
   Length[CprimeI]
      ∑          Cprimewght[[i]] *
     i=1
    smoothStep1[conRateCprimeI[[i]], CprimeI[[i]], Cprimecal[[i]]] *
    (Cprimecal[[i]] - CprimeI[[i]])^2 +
   Length[C44I]
      ∑          C44wght[[i]] * smoothStep1[conRateC44I[[i]],
     i=1
    C44I[[i]], C44cal[[i]]] * (C44cal[[i]] - C44I[[i]])^2 +
   Length[HOMI]
      ∑          HOMwght[[i]] * smoothStep1[conRateHOMI[[i]],
     i=1
    HOMI[[i]], HOMcal[[i]]] * (HOMcal[[i]] - HOMI[[i]])^2 +
   Length[diffEI]
      ∑          diffEwght[[i]] * smoothStep1[conRatediffEI[[
     i=1
    i]], diffEI[[i]], diffEcal[[i]]] * (diffEcal[[i]] - diffEI[[i]])^2;
```

```
constraint1 = {0 ≤ D12 ≤ 10, 1/2 ≤ G12 ≤ 7/2,
   0 < 5/4 * β12 ≤ α12 ≤ 5/2 * β12 < 10, 0 ≤ D22 ≤ 10, 1/2 ≤ G22 ≤ 7/2,
   0 < 5/4 * β22 ≤ α22 ≤ 5/2 * β22 < 10, 0 ≤ C2 ≤ 60, 0 ≤ δ2 ≤ 10,
   0 ≤ A2 ≤ 0.0005, 0.3 ≤ B2 ≤ 1.2, 3 ≤ ρx2 ≤ 14, 50 ≤ ρy2};
constraint2 = {vcon, ρlow2, ρhigh2, 7};
constraint3 = {ρa[2, rcut[[2]] - smooth[[2]], rcut[[2]] - smooth[[2]]] ≤
    vcon * ρa[2, r0 /. r0I[[7]], r0 /. r0I[[7]]],
   ρlow2 ≤ ρa[2, r0 /. r0I[[7]], r0 /. r0I[[7]]] < ρhigh2,
   Abs[ϕ[2, 2, rcut[[2]] - smooth[[2]], rcut[[2]] - smooth[[2]]]] ≤
    vcon * Abs[ϕ[2, 2, r0 /. r0I[[7]], r0 /. r0I[[7]]]],
   Abs[ϕ[1, 2, rcut[[3]] - smooth[[3]], rcut[[3]] - smooth[[3]]]] ≤
    vcon * Abs[ϕ[1, 2, ratiolat[[1, 1, 2, 1]] * (r0 /. r0I[[1]]),
       ratiolat[[1, 1, 2, 1]] * (r0 /. r0I[[1]])]]};
defrange = Join[{tmp-fit}, constraint1, constraint3];
ByteCount[defrange]
defrange /. initialPara
```

```
5 938 384
```

```
{2.91515, True, True, True, True, True, True, True,
 True, True, True, True, True, True, True, True, True}
```

## Auto fit

```
preVar1 = {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60,
   C2 → 11.003912875394953`, D12 → 0.234580672977548`,
   D22 → 0.07287155024003068`, G12 → 1.5125423949321948`,
   G22 → 2.4983485612790086`, α12 → 4.842354956626778`,
   α22 → 3.6749949240472164`, β12 → 2.1374260366912874`,
   β22 → 1.471049785271651`, δ2 → 1.323659492018399`};
iteration1 = 10 000;
Timing[afm1 = NMinimize[defrange,
   continueFit[preVar1, myvar], MaxIterations → iteration1,
   Method → "DifferentialEvolution"]]

preVar2 = {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60,
   C2 → 11.003912875394953`, D12 → 0.234580672977548`,
   D22 → 0.07287155024003068`, G12 → 1.5125423949321948`,
   G22 → 2.4983485612790086`, α12 → 4.842354956626778`,
   α22 → 3.6749949240472164`, β12 → 2.1374260366912874`,
   β22 → 1.471049785271651`, δ2 → 1.323659492018399`};
iteration2 = 10 000;
Timing[afm2 = NMinimize[defrange, continueFit[preVar2, myvar],
   MaxIterations → iteration2, Method → "SimulatedAnnealing"]]

preVar3 = {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60,
   C2 → 11.003912875394953`, D12 → 0.234580672977548`,
   D22 → 0.07287155024003068`, G12 → 1.5125423949321948`,
   G22 → 2.4983485612790086`, α12 → 4.842354956626778`,
   α22 → 3.6749949240472164`, β12 → 2.1374260366912874`,
   β22 → 1.471049785271651`, δ2 → 1.323659492018399`};
iteration3 = 10 000;
Timing[afm3 = NMinimize[defrange,
   continueFit[preVar3, myvar], MaxIterations → iteration3]]

preVar4 = {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60,
   C2 → 11.003912875394953`, D12 → 0.234580672977548`,
   D22 → 0.07287155024003068`, G12 → 1.5125423949321948`,
   G22 → 2.4983485612790086`, α12 → 4.842354956626778`,
   α22 → 3.6749949240472164`, β12 → 2.1374260366912874`,
   β22 → 1.471049785271651`, δ2 → 1.323659492018399`};
iteration4 = 10 000;
randoms4 = 101;
Timing[afm4 = NMinimize[defrange,
   continueFit[preVar4, myvar], MaxIterations → iteration4,
   Method → {"NelderMead", "RandomSeed" → randoms4}]]
```

```
{513. Second,
 {2.54646, {A2 → 0.000197047, B2 → 1.1886, C2 → 11.0025,
   D12 → 0.249454, D22 → 0.0661496, G12 → 1.50964,
   G22 → 2.5198, α12 → 4.82613, α22 → 3.67263, β12 → 2.13158,
   β22 → 1.47797, δ2 → 1.30927, ρx2 → 9.9978, ρy2 → 60.0155}}}
```

## Analysis

### General

```
generalPara =
  {A2 → 0.23754730586674475`, B2 → 0.000999998853380872`,
   C2 → 9.391168892063586`, D12 → 0.2674212501485524`,
   D22 → 0.12071811871765702`, G12 → 1.8080765547788364`,
   G22 → 1.35013306001794`, α12 → 4.631167441753063`,
   α22 → 9.999999855205578`, β12 → 3.0874449221733165`,
   β22 → 1.7729518767039203`, δ2 → 1.8160117926276569`};
Print[generalPara];
modϕ[2, 2, 0, generalPara];
eval[generalPara];
modϕ[2, 2, 0, generalPara];
```

### Initial

```
Print[initialPara];
modϕ[2, 2, 0, initialPara];
eval[initialPara];
modϕ[2, 2, 0, initialPara];
```

### This fit

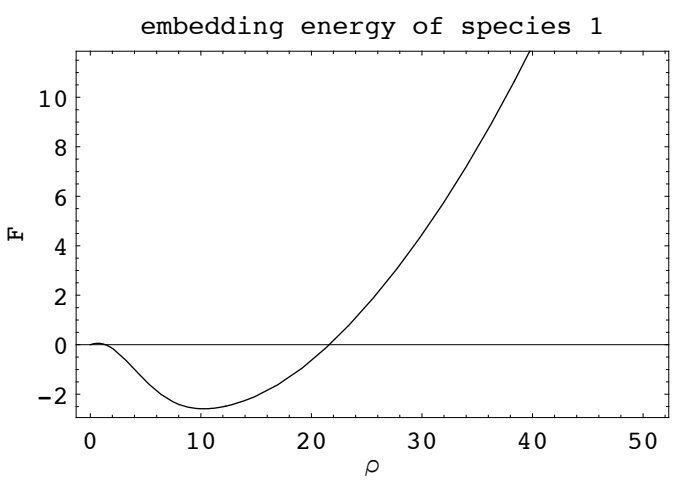
```
Print["continue from ", preVar2];
Print["MaxIterations = ", iteration2];
Print[afm2];
modϕ[2, 2, 0, afm2[[2]]];
eval[afm2[[2]]];
modϕ[2, 2, 0, afm2[[2]]];
```

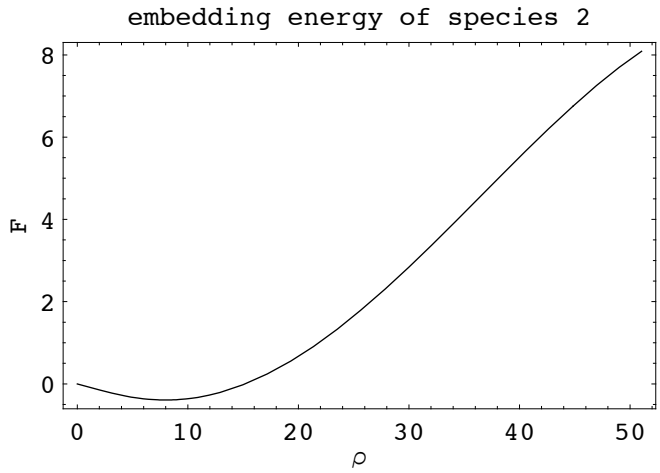
```
continue from {A2 → 0.0001, B2 → 1.2, ρx2 → 10, ρy2 → 60, C2 → 11.0039,
  D12 → 0.234581, D22 → 0.0728716, G12 → 1.51254, G22 → 2.49835,
  α12 → 4.84235, α22 → 3.67499, β12 → 2.13743, β22 → 1.47105, δ2 → 1.32366}

MaxIterations = 10 000

{2.54646, {A2 → 0.000197047, B2 → 1.1886,
  C2 → 11.0025, D12 → 0.249454, D22 → 0.0661496, G12 → 1.50964,
  G22 → 2.5198, α12 → 4.82613, α22 → 3.67263, β12 → 2.13158,
  β22 → 1.47797, δ2 → 1.30927, ρx2 → 9.9978, ρy2 → 60.0155}}

{5.35, 5.35, 4.95}

{{OC, {1, 0}, 2.74701}, {OC, {0, 1}, 2.39002}}
```

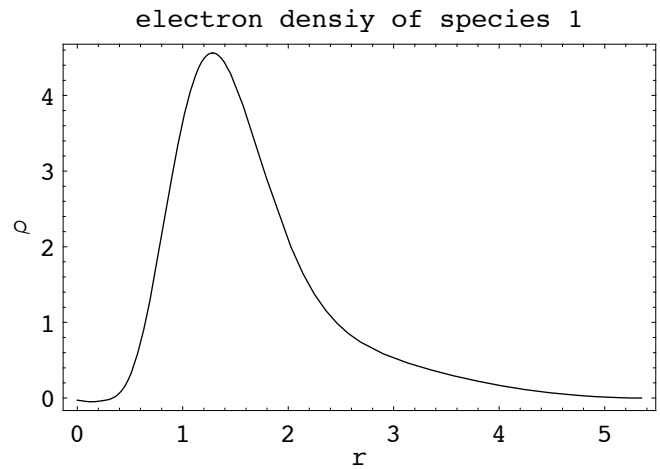
```
EABfit =
 {{OC, {1., 0.25}, {-3.5668, =}, {1.05, >}, {r0 → 2.78749}, {0.055, =},
   {0.459, =}}, {OC, {1., 0.5}, {-3.4677, =}, {1.05, >}, {r0 → 2.82892},
   {0.135, =}, {0.42, =}}, {OC, {1., 0.75}, {-3.4126, =},
   {1.05, >}, {r0 → 2.87036}, {0.164, =}, {0.382, =}},
  {OC, {1., 1.}, {-3.3807, =}, {1.05, >}, {r0 → 2.9118},
   {0.146, =}, {0.344, =}}, {TE, {1., 0.5}, {-3.5, >},
   {1.05, =}, {r0 → 2.82892}, {0.163, =}, {0.381, =}},
  {TE, {1., 0.75}, {-3.5, >}, {1.05, =}, {r0 → 2.87036},
   {0.163, =}, {0.381, =}}, {OC, {0., 1.}, {-2.119, =},
   {0.5, >}, {r0 → 2.39002}, {0.05, >}, {0.05, >}}}

Ewght = {0, 0, 0, 2, 0, 0, 2}

Bwght = {1, 1, 1, 1, 0, 0, 1}

Pwght = {2, 2, 2, 2, 1, 1, 2}

Cprimewght = {1, 1, 1, 1, 0.5, 0.5, 1}

C44wght = {2, 2, 2, 2, 0.5, 0.5, 1}
```

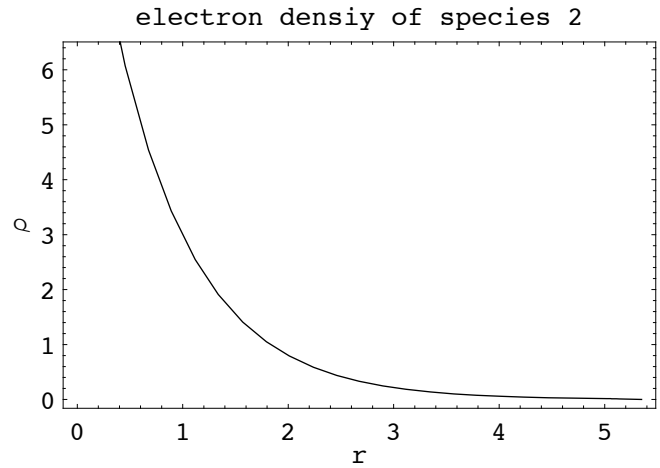
$$\text{constraint1} = \left\{0 \le D12 \le 10,\ \frac{1}{2} \le G12 \le \frac{7}{2},\ 0 < \frac{5\,\beta 12}{4} \le \alpha 12 \le \frac{5\,\beta 12}{2} < 10,\ 0 \le D22 \le 10,\ \frac{1}{2} \le G22 \le \frac{7}{2},\ 0 < \frac{5\,\beta 22}{4} \le \alpha 22 \le \frac{5\,\beta 22}{2} < 10,\ 0 \le C2 \le 60,\ 0 \le \delta 2 \le 10,\ 0 \le A2 \le 0.0005,\ 0.3 \le B2 \le 1.2,\ 3 \le \rho x2 \le 14,\ 50 \le \rho y2\right\}$$

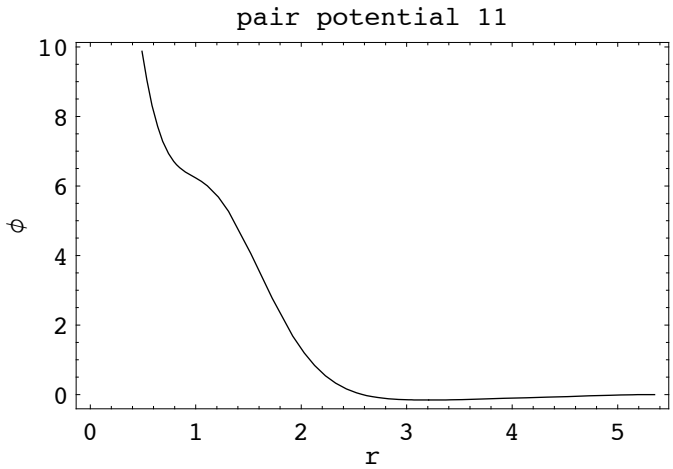
```
constraint2 = {0.05, 0.3, 0.8, 7}
```

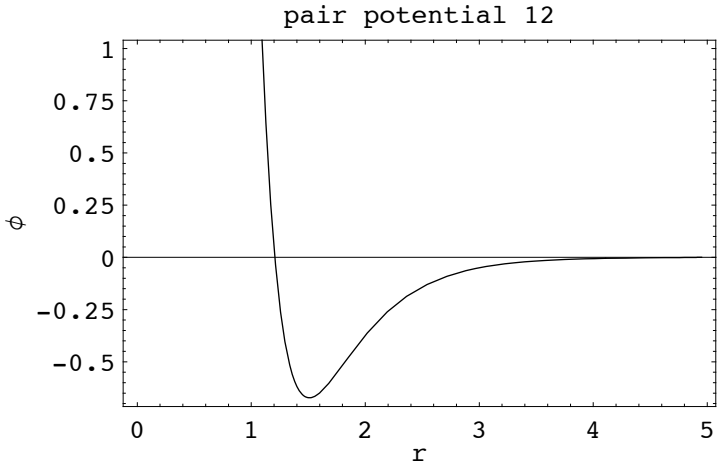


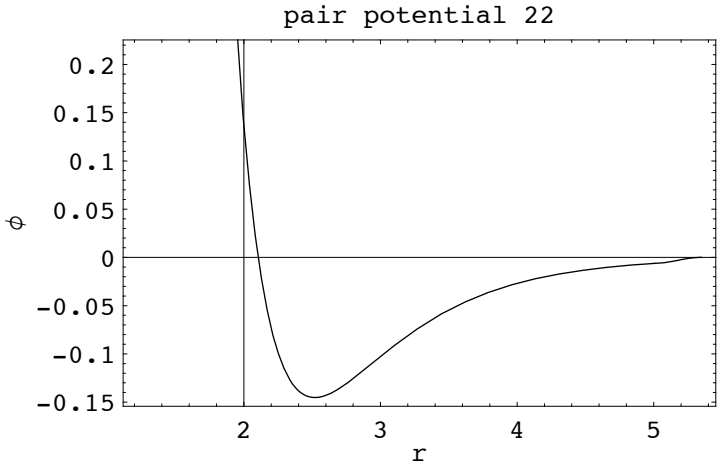

embedding energy of species 2
F
ρ
0
2
4
6
8
0
10
20
30
40
50

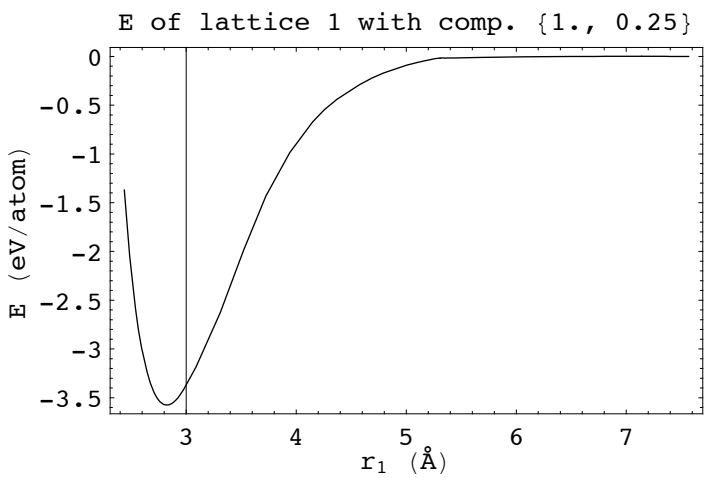

electron densiy of species 1
ρ
r
0
1
2
3
4
5

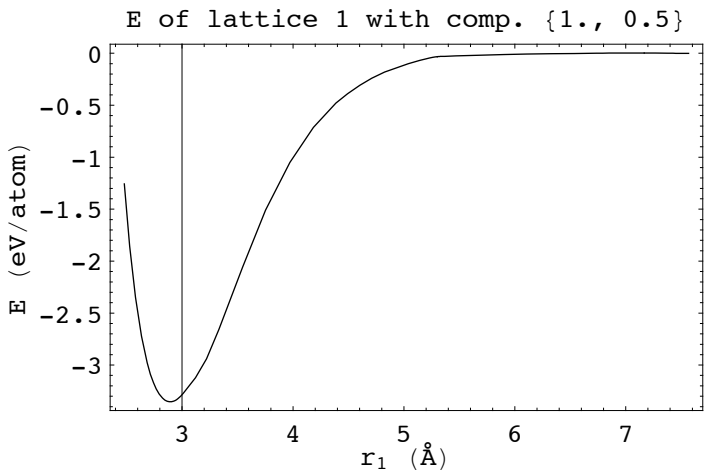

electron densiy of species 2
ρ
r
0
1
2
3
4
5
6

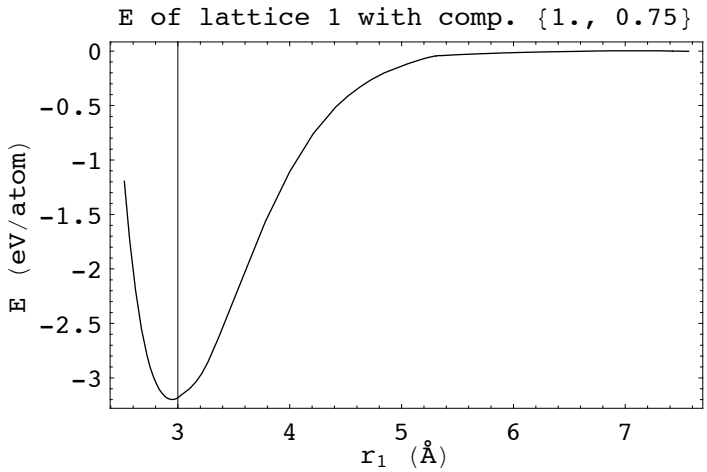

pair potential 11
10
8
6
4
2
0
ϕ
0
1
2
3
4
5
r

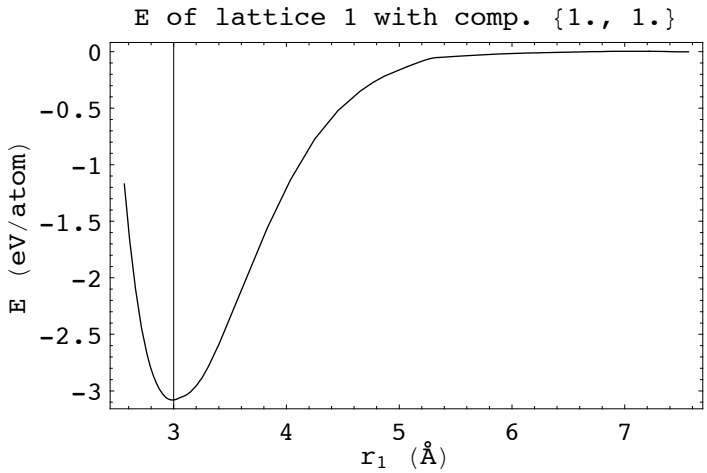

pair potential 12
1
0.75
0.5
0.25
0
-0.25
-0.5
ϕ
0
1
2
3
4
5
r

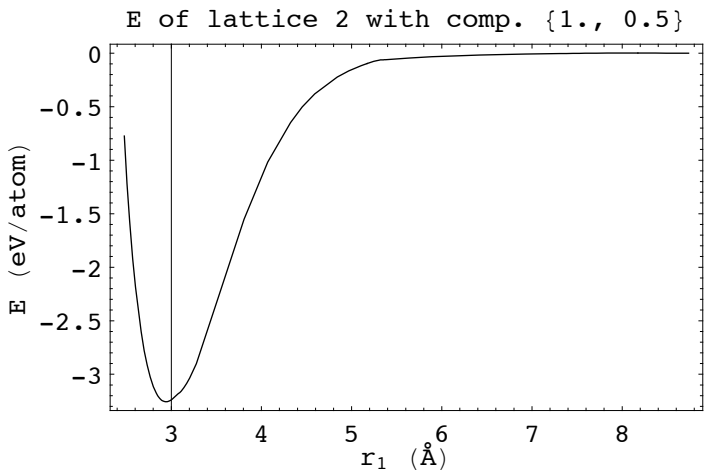

pair potential 22
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
ϕ
2
3
4
5
r

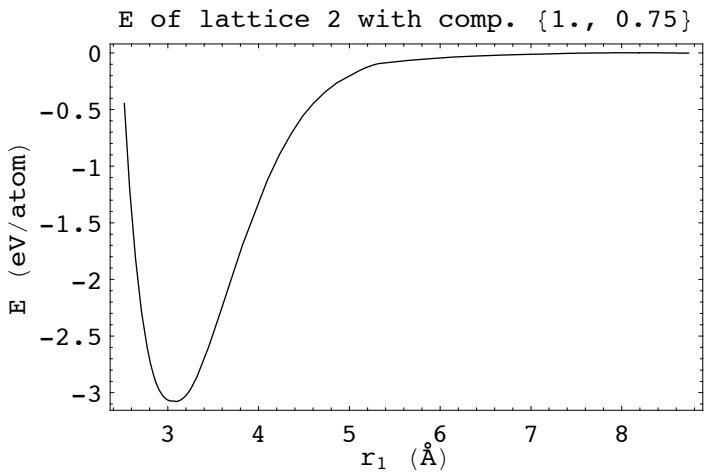

E of lattice 1 with comp. {1., 0.25}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
3
4
5
6
7
r1 (Å)

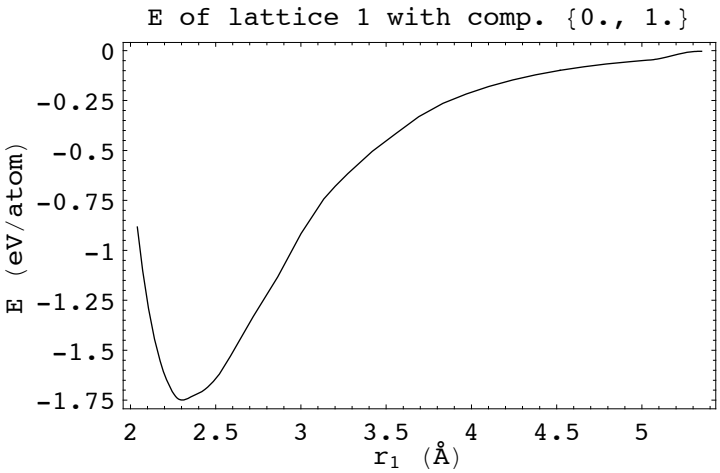

E of lattice 1 with comp. {1., 0.5}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
r1 (Å)

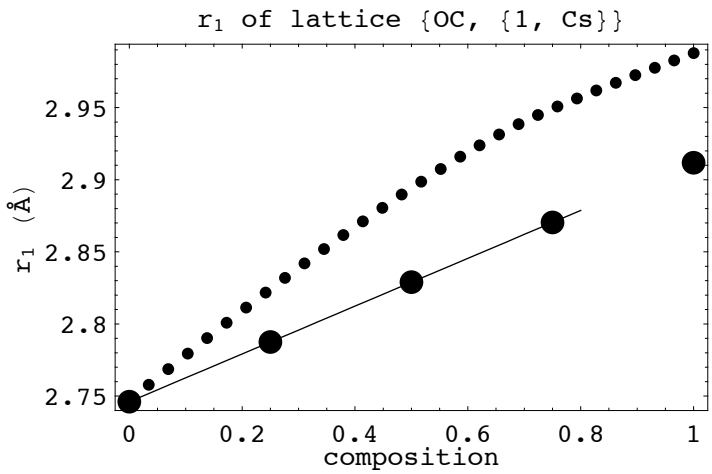

E of lattice 1 with comp. {1., 0.75}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
r1 (Å)

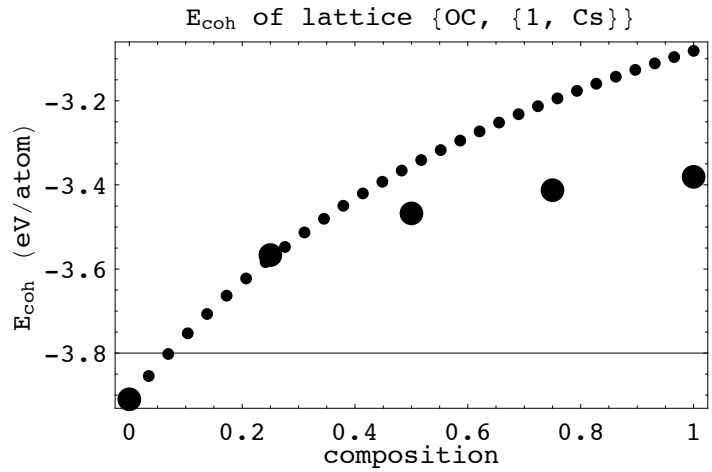

E of lattice 1 with comp. {1., 1.}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
r1 (Å)

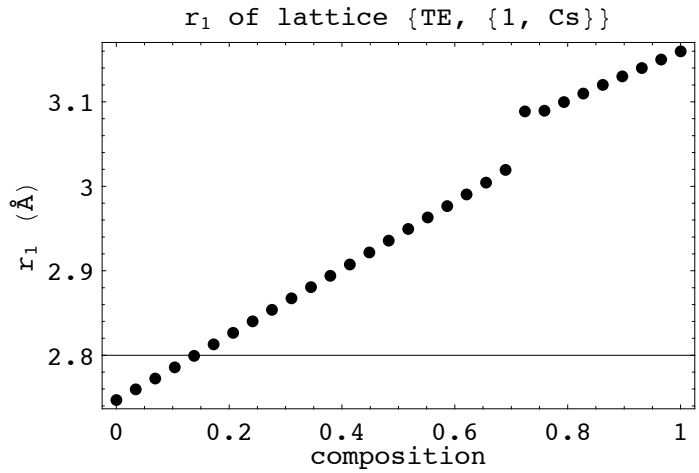

E of lattice 2 with comp. {1., 0.5}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
8
r1 (Å)

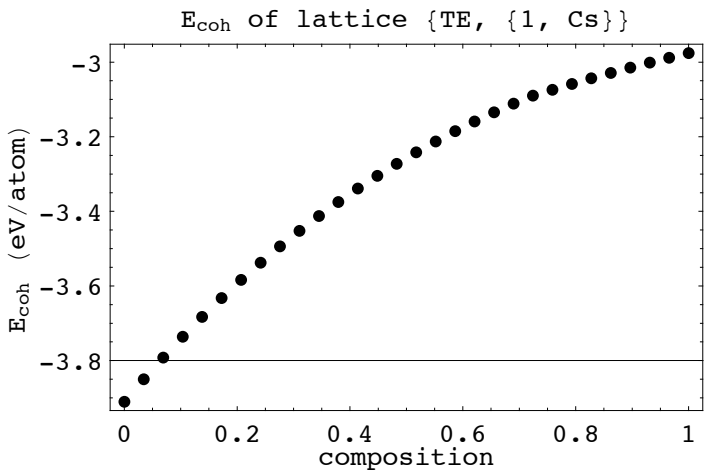

E of lattice 2 with comp. {1., 0.75}
E (eV/atom)
0
-0.5
-1
-1.5
-2
-2.5
-3
3
4
5
6
7
8
r1 (Å)

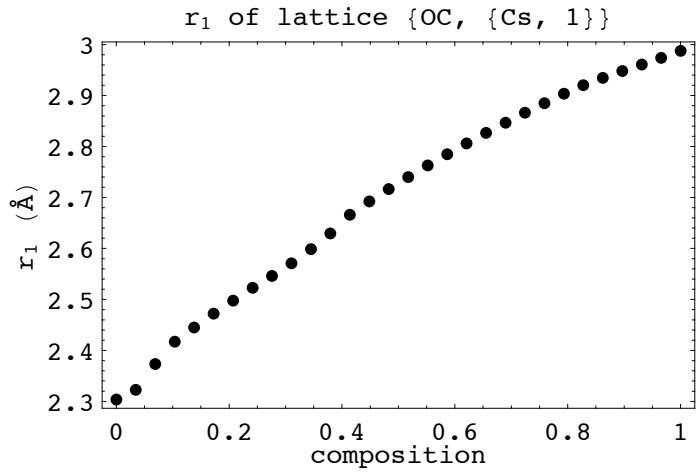

E of lattice 1 with comp. {0., 1.}
E (eV/atom)
0
-0.25
-0.5
-0.75
-1
-1.25
-1.5
-1.75
2
2.5
3
3.5
4
4.5
5
r1 (Å)

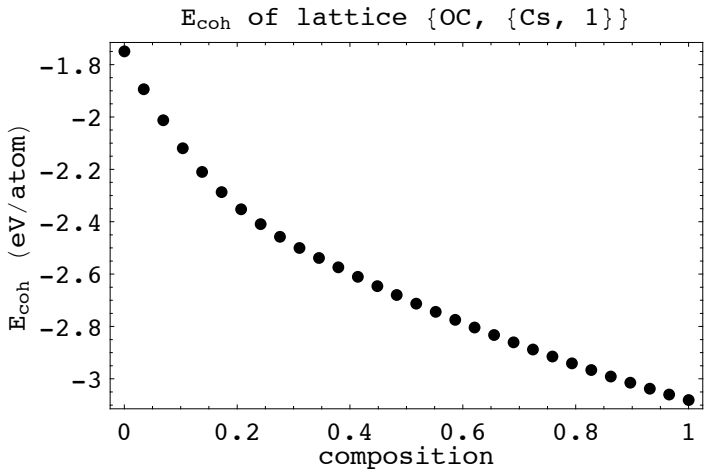

r1 of lattice {OC, {1, Cs}}
r1 (Å)
2.95
2.9
2.85
2.8
2.75
0
0.2
0.4
0.6
0.8
1
composition

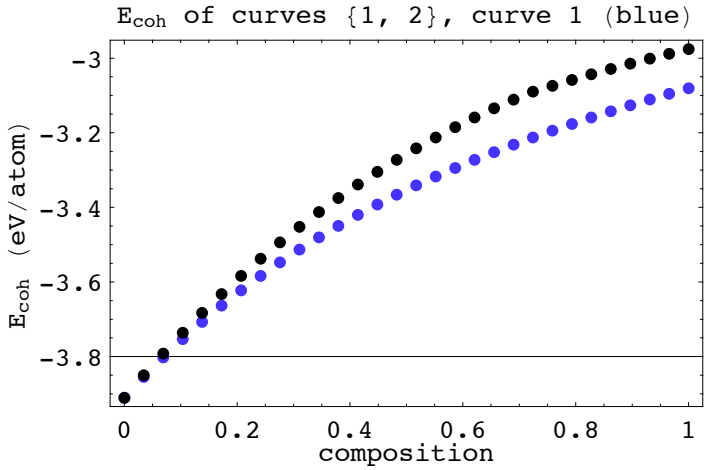

Ecoh of lattice {OC, {1, Cs}}
Ecoh (eV/atom)
-3.2
-3.4
-3.6
-3.8
0
0.2
0.4
0.6
0.8
1
composition


r1 of lattice {TE, {1, Cs}}
r1 (Å)
3.1
3
2.9
2.8
0
0.2
0.4
0.6
0.8
1
composition


Ecoh of lattice {TE, {1, Cs}}
Ecoh (eV/atom)
-3
-3.2
-3.4
-3.6
-3.8
0
0.2
0.4
0.6
0.8
1
composition


r1 of lattice {OC, {Cs, 1}}
r1 (Å)
3
2.9
2.8
2.7
2.6
2.5
2.4
2.3
0
0.2
0.4
0.6
0.8
1
composition




check cohesive energy:

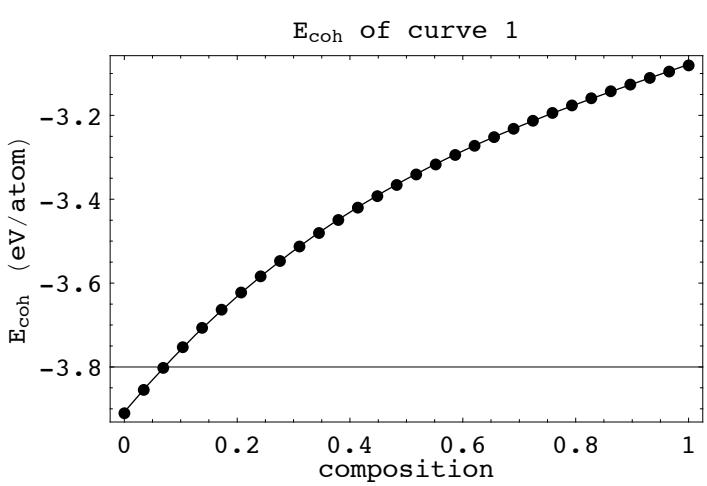
| | | | | |
|---|---|---|---|---|
| OC | { 1.00, 0.25} | -3.5668 | -3.5626 | 0.0000 |
| OC | { 1.00, 0.50} | -3.4677 | -3.3233 | 0.0000 |
| OC | { 1.00, 0.75} | -3.4126 | -3.1563 | 0.0000 |
| OC | { 1.00, 1.00} | -3.3807 | -3.0390 | 2.0000 |
| TE | { 1.00, 0.50} | -3.5000 | -3.1686 | 0.0000 |
| TE | { 1.00, 0.75} | -3.5000 | -2.9124 | 0.0000 |
| OC | { 0.00, 1.00} | -2.1190 | -1.7194 | 2.0000 |

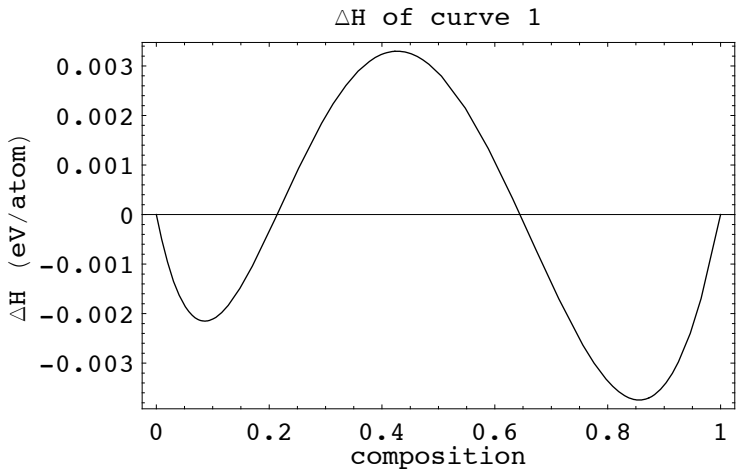
check bulk modulus:

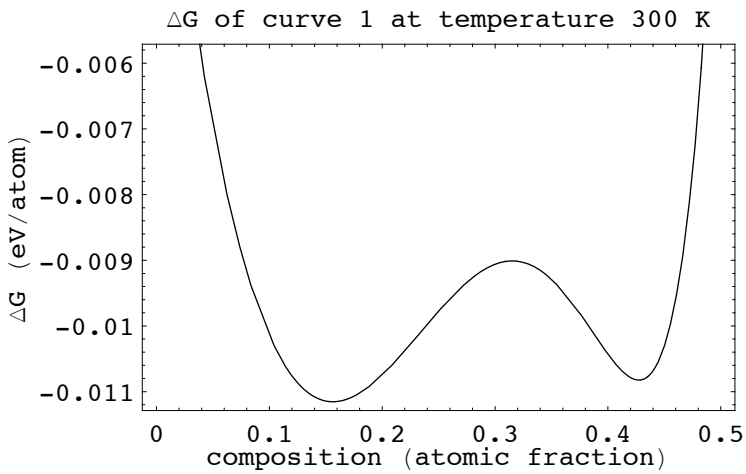
| | | | | |
|---|---|---|---|---|
| OC | { 1.00, 0.25} | 1.0500 | 1.3379 | 1.0000 |
| OC | { 1.00, 0.50} | 1.0500 | 1.3792 | 1.0000 |
| OC | { 1.00, 0.75} | 1.0500 | 1.4537 | 1.0000 |
| OC | { 1.00, 1.00} | 1.0500 | 1.5421 | 1.0000 |
| TE | { 1.00, 0.50} | 1.0500 | 1.5401 | 0.0000 |
| TE | { 1.00, 0.75} | 1.0500 | 1.8841 | 0.0000 |
| OC | { 0.00, 1.00} | 0.5000 | -0.2531 | 1.0000 |

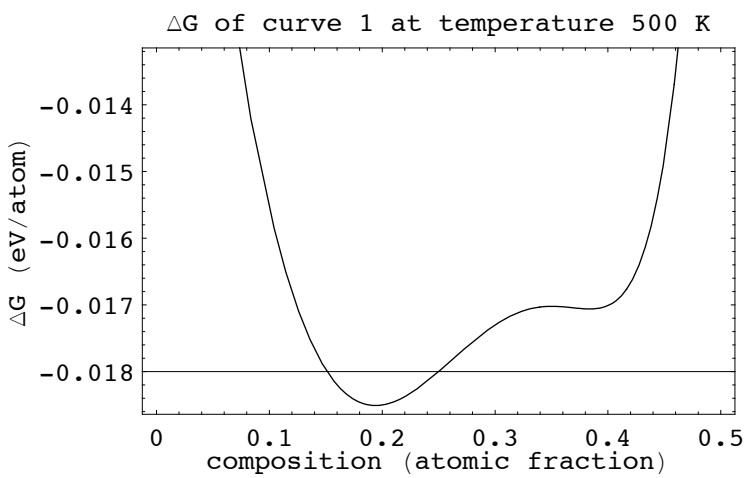
check pressure:

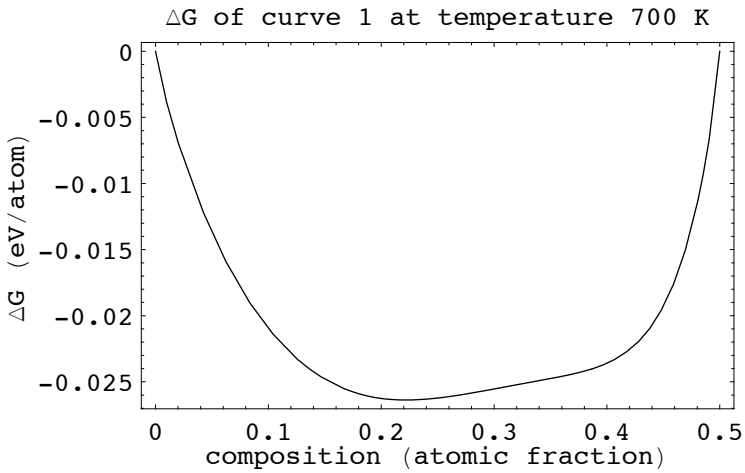
```
OC    { 1.00,  0.25}        0.0000        0.0494        2.0000
OC    { 1.00,  0.50}        0.0000        0.0843        2.0000
OC    { 1.00,  0.75}        0.0000        0.1087        2.0000
OC    { 1.00,  1.00}        0.0000        0.1246        2.0000
TE    { 1.00,  0.50}        0.0000        0.1508        1.0000
TE    { 1.00,  0.75}        0.0000        0.2172        1.0000
OC    { 0.00,  1.00}        0.0000       -0.0287        2.0000

check C':

OC    { 1.00,  0.25}        0.0550        0.1503        1.0000
OC    { 1.00,  0.50}        0.1350        0.1649        1.0000
OC    { 1.00,  0.75}        0.1640        0.1791        1.0000
OC    { 1.00,  1.00}        0.1460        0.1893        1.0000
TE    { 1.00,  0.50}        0.1630        0.2926        0.5000
TE    { 1.00,  0.75}        0.1630        0.6125        0.5000
OC    { 0.00,  1.00}        0.0500       -0.7857        1.0000

check C44':

OC    { 1.00,  0.25}        0.4590        0.3163        2.0000
OC    { 1.00,  0.50}        0.4200        0.2940        2.0000
OC    { 1.00,  0.75}        0.3820        0.2691        2.0000
OC    { 1.00,  1.00}        0.3440        0.2351        2.0000
TE    { 1.00,  0.50}        0.3810        0.3514        0.5000
TE    { 1.00,  0.75}        0.3810        0.4168        0.5000
OC    { 0.00,  1.00}        0.0500        0.1301        1.0000

check relaxed atomic spacing and cohesive energy:

OC    { 1.00,  0.25}        2.8243       -3.5743
OC    { 1.00,  0.50}        2.8942       -3.3532
OC    { 1.00,  0.75}        2.9493       -3.1986
OC    { 1.00,  1.00}        2.9878       -3.0808
TE    { 1.00,  0.50}        2.9427       -3.2567
TE    { 1.00,  0.75}        3.0888       -3.0779
OC    { 0.00,  1.00}        2.3037       -1.7491
```

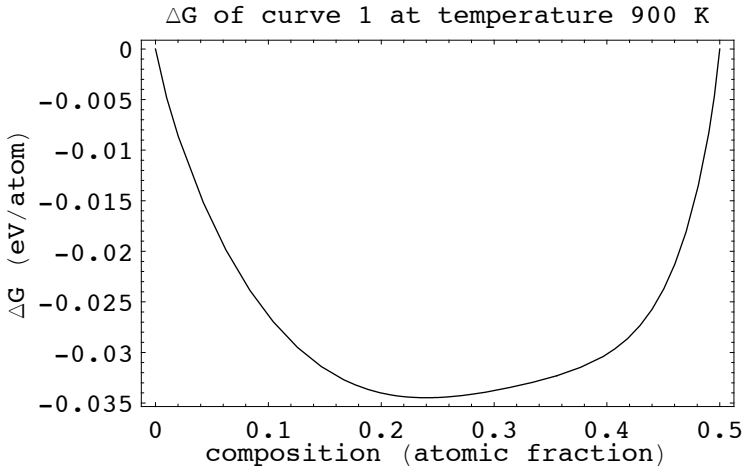


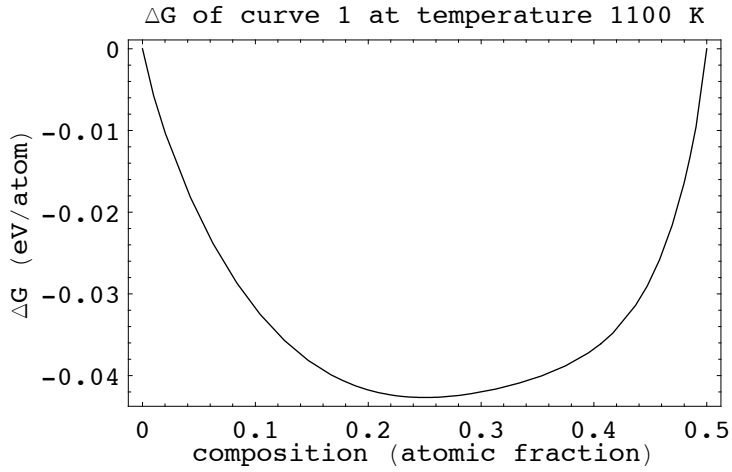

ΔH of curve 1
0.003
0.002
0.001
0
-0.001
-0.002
-0.003
ΔH (eV/atom)
0
0.2
0.4
0.6
0.8
1
composition

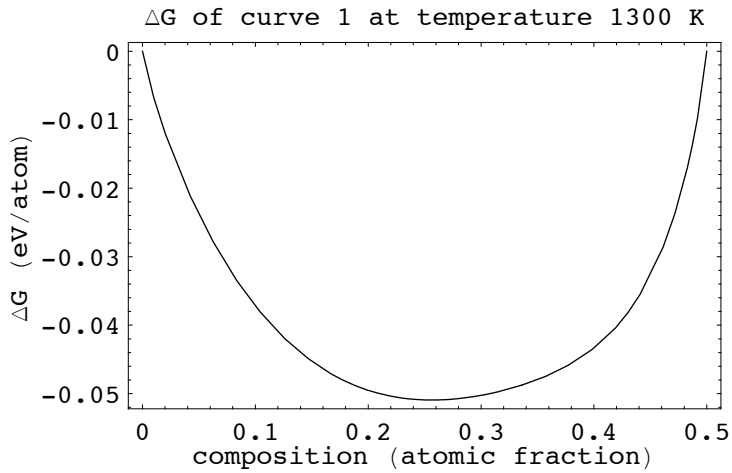

ΔG of curve 1 at temperature 300 K
-0.006
-0.007
-0.008
-0.009
-0.01
-0.011
ΔG (eV/atom)
0
0.1
0.2
0.3
0.4
0.5
composition (atomic fraction)

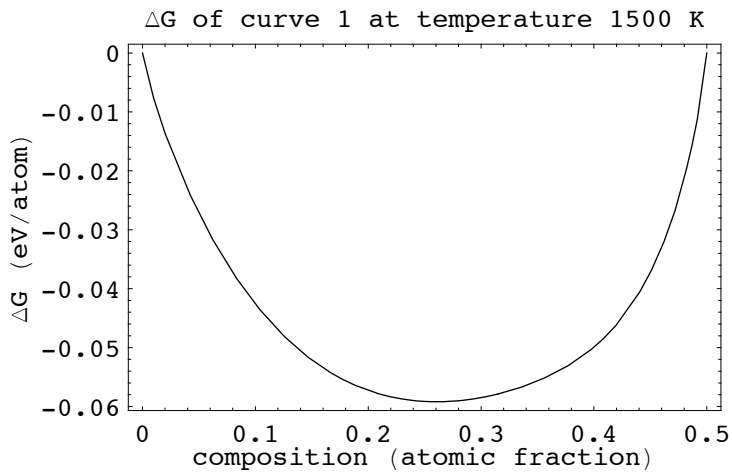

ΔG of curve 1 at temperature 500 K
-0.014
-0.015
-0.016
-0.017
-0.018
ΔG (eV/atom)
0
0.1
0.2
0.3
0.4
0.5
composition (atomic fraction)


ΔG of curve 1 at temperature 700 K
ΔG (eV/atom)
0
-0.005
-0.01
-0.015
-0.02
-0.025
0
0.1
0.2
0.3
0.4
0.5
composition (atomic fraction)


ΔG of curve 1 at temperature 900 K
ΔG (eV/atom)
0
-0.005
-0.01
-0.015
-0.02
-0.025
-0.03
-0.035
0
0.1
0.2
0.3
0.4
0.5
composition (atomic fraction)


ΔG of curve 1 at temperature 1100 K
ΔG (eV/atom)
0
-0.01
-0.02
-0.03
-0.04
0
0.1
0.2
0.3
0.4
0.5
composition (atomic fraction)




## ■ Create .set EAM Potential Tables

```
generalPara =
  {A2 → 0.00004995581291627853`, B2 → 0.9000017417206814`,
   C2 → 19.99999781875086`, D12 → 0.46585453076105293`,
   D22 → 0.2531516209718132`, G12 → 1.5494962682027826`,
   G22 → 2.5902281467314805`, α12 → 3.780341962477035`,
   α22 → 2.502382971835428`, β12 → 2.517841809096612`,
   β22 → 1.6682552998093128`, δ2 → 1.4978271398606449`,
   ρx2 → 10.484713893708347`, ρy2 → 60.06165705777814`};
modϕ[2, 2, 0, generalPara];
WriteEAM["files\ing2_Feb16_4.set",
  generalPara, 500, 500, ρmax];
modϕ[2, 2, 0, generalPara];
```

# Appendix B

# User's Manual for FitEAM

## B.1 Overview

FitEAM is a Mathematica code for parameterization and analysis of EAM interatomic potentials. In addition to fitting potentials, it can also read existing elemental EAM potential tables in the .set format and combine them to determine the remaining unknown potentials for alloys. Analytical equations are used to fit these unknown EAM functions. The analytical functions used are described here.

- Embedding energy $F(\rho) = A\rho \ln[\rho] + B\rho - S\rho$. Here $A$ and $B$ are parameters to be determined and $S$ is a normalization factor. The purpose of this normalization is to ensure that the first derivative of the embedding energy function equals zero for the equilibrium structure of each element. $S$ can be shown to equal to the first derivative of the embedding energy evaluated at $S = 0$, *i.e.* $S = F_p'(\rho_0)$, where the subscript "p" shows that the embedding energy referred to here is prior to normalization (equivalent to $S = 0$), and $\rho_0$ is the electron density at the atom site in the equilibrium crystal structure of the element. Using $S = F_p'(\rho_0)$, $S$ can be expressed in terms of other fitting parameters. In the code, $S$ as a function of other parameters is derived through execution of the function **normalize**[...].

  It is emphasized here that the normalization appears to be very important in a successful alloy EAM potential parameterization because the EAM expression of elements is not unique: a linear increase in the embedding energy can be exactly compensated by a corresponding decrease in the pair energy. During fitting, which consists of minimizing a user defined objective function, this ambiguous EAM property often results in a drift of the embedding energy and pair energy to unrealistically large values. The normalization removes this ambiguity, and ensures the physical ranges of the potential functions. While $B\rho - S\rho$ can be consolidated to a single linear term, the $B\rho$ term is retained so that the user can have a choice to not to normalize the function. Due to the importance of the normalization, however, the code is set to execute the function **normalize**[...] by default. When an existing elemental EAM potential is used to develop the alloy EAM potential, the existing elemental potential may not

be normalized. The normalization of the existing embedding energy function, $F_p(\rho)$, can be achieved similarly by using $F(\rho) = F_p(\rho) - S\rho$. In this case, however, a corresponding term involving the $S$ factor needs to be introduced in the pair energy so that the net effect of the normalization is to redistribute the embedding and pair energies without changing the total energy of the element.

- Electron density $f(r) = C\exp[-\delta r]$. Here C and $\delta$ are fitting parameters. As important as the normalization introduced in the embedding and pair energy functions, the electron density also needs to be normalized. This is because electron density for any element is only defined on a relative unit. This means that one can always multiply the electron density $\rho$ by a constant $c$ without changing the total energy of the element as long as a corresponding change is made to the embedding energy, *i.e.* replace $\rho$ with $\rho/c$ in the embedding energy function. This property will not affect the parameterization of elemental potentials but will affect the alloy potentials. While electron densities of different elements are expected to be different, they are not expected to be different by orders of magnitudes. Experience has shown that producing stable alloy potentials necessitates electron densities of a similar range. The normalization of electron density is to ensure that the electron densities of different elements operate in the same range. This can be effectively implemented using the constraint $\rho_{\text{low}} < \rho_{0,i} < \rho_{\text{high}}$ for different elements ($i = 1, 2, 3 \ldots$) during fitting, where $\rho_{0,i}$ is the equilibrium electron density at an atom site for element $i$, and $\rho_{\text{low}}$ and $\rho_{\text{high}}$ are the lower and upper bounds, respectively, for the electron density that are common to all elements.

- Pair energy $\phi(r) = D\{\beta\exp[-\alpha(r-r_0)] - \alpha\exp[-\beta(r-r_0)]\}$. Here $\alpha$, $\beta$, $D$ and $r_0$ are fitting parameters. The fitting parameters have physical meanings in that $r_0$ represents the equilibrium bond length, and $D$ equals the binding energy of a dimer. Most materials have a $\alpha/\beta$ ratio close to 2 (Morse potential, for example, satisfies $\alpha/\beta = 2$). Accordingly, we can frequently use the constraints $1.5 < \alpha/\beta < 2.5$, $\alpha > 0$, and $\beta > 0$ during fitting. Note that if normalization is performed on an existing embedding energy function, a corresponding adjustment must be made to the pair energy of the same element to ensure that the elemental EAM potential remains invariant. This adjustment can be expressed as $\phi(r) = \phi_p(r) + 2Sf(r)$, where $\phi_p(r)$ is the pair potential prior to the adjustment, and $S$ is the normalization factor described above.

It is important to implement a cutoff distance $r_c$ for both (pairwise) electron density and pair energy functions. Two steps are used. First, we need to ensure that both functions naturally decay to small values when $r$ gets close to $r_c$, say, $r = r_c - r_s$, where $r_s$ is a small distance from $r_c$. This can be achieved by using the constraints $\frac{f(r_c-r_s)}{f(r_1)} < \varepsilon$ and $\frac{\phi(r_c-r_s)}{\phi(r_1)} < \varepsilon$ during optimization of the function parameters, where $\varepsilon$ is a small number, and $r_1$ is the lattice spacing of the equilibrium structure (a nominal atomic separation that is commonly observed during simulations). The second step is to ensure that both pairwise functions indeed drop to zero at $r = r_c$. This can be done

by multiplying them with a cutoff function $\frac{1}{2}+\frac{1}{2}\cos\left[\frac{\pi(r-r_c+r_s)}{r_s}\right]$ when $r_c > r > r_c - r_s$. Both $r_c$ and $r_s$ are viewed as important potential parameters as they do affect the results. However, they are not implemented as the fitting parameters (*i.e.* they are not allowed to change during each parameterization run). This significantly improves the efficiency of the parameterization. On the other hand, $r_c$ and $r_s$ are relatively more intuitive to determine as a few trial-and-error efforts can lead to satisfactory values of $r_c$ and $r_s$.

The code is arranged in the following sections and sub-sections:

- Include Files and Utility Functions
- Pre-analysis
- Lattices
- Lattice Analysis
- EAM Functions
  - Initial parameters and initial EAM tables
  - Fitting embedding energy table
  - Fitting electron density table
  - Fitting pair energy table
  - Define EAM functions
  - Plot EAM functions
- Print Initial Cohesive Energy
- Fitting ...
  - Set up variables
  - Input properties
  - Analytical equations for EAB
  - Analytical equations for elastic constants
  - Quick test
  - Fit setups
  - Auto fit
- Analysis
  - General
  - Initial

  - This fit

- Create .set EAM Potential Tables

In general, these sections should be executed in sequence. A proficiency level that allows skip of sections is expected after numerous practices.

## B.2 Terminology

**Lattice spacing**: lattice spacing used in this code refers to the nearest neighbor distance in the base sublattice of the lattice. For instance, we can define a lattice called "OC" that contain two fcc sublattices occupied by Pd and H respectively where the H sublattice is composed of the octahedral interstitial sites of Pd and vice versa. In that case, lattice spacing is the nearest neighbor distance between Pd and Pd atoms or between H and H atoms (*i.e.* atoms in the same sublattice) instead of the shorter distance between Pd and H atoms.

**Relative neighbor distance**: relative means that the distance is normalized by the lattice spacing.

**Composition**: A lattice can contain more than one composition value, each corresponding to occupancy fraction on one of the sublattices. A structure is fully determined only when its lattice type, composition, and lattice constant are given. Composition can be abbreviated. In that case, the value(s) are aligned with the last element/sublattice and the remaining missing values are padded with 1.

## B.3 Structure of Mathematica notebook

1. Include Files and Utility Functions

   Includes the necessary Mathematica libraries, and defines utility functions for various calculations such as finding neighbors, calculating cohesive energy, plotting cohesive energy as a function of lattice spacing, calculating elastic constant, relaxing the lattice constant, printing out various properties, and writing the EAM table file for molecular dynamics simulations etc.

2. Pre-analysis

   Takes raw data from DFT calculations or experiments, converts them to the target values that the EAM is fitted to, and creates graphics of DFT or experimental data so that they can be compared to the predicted data during analysis of the potentials.

3. Lattices

   Defines lattices. Note that a lattice can contain several sublattices of different elements. A structure is defined only when its lattice, the occupancy probabilities of species on sublattices, and the lattice constant are all defined.

4. Lattice Analysis

   Determines coordinates and relative neighbor distances within the cutoff range of the potential from a center atom. The results are a function of lattice type, the species of the center atom, and the species of the neighbor atoms.

5. EAM Functions

   Defines EAM functions.

   (a) Initial parameters and initial EAM tables

   Defines EAM fitting parameters and reads in known EAM potential .set tables.

   (b) Fitting embedding energy table

   Converts the known embedding energy tables into polynomial functions.

   (c) Fitting electron density table

   Converts the known electron density tables into polynomial functions.

   (d) Fitting pair energy table

   Converts the known pair energy tables into polynomial functions.

   (e) Define EAM functions

   Defines universal EAM functions by consolidating the known EAM polynomials and the unknown EAM functions to be fitted.

   (f) Plot EAM functions

   Plots out all EAM functions. The unknown functions are plotted using the initial guessed EAM parameters.

6. Print Initial Cohesive Energy

   Plots out energy vs. atomic spacing (r1) curves for various structures defined earlier by users. The unknown functions are based upon the initial guessed EAM parameters.

7. Fitting ...

   Performs the fitting using four different methods: differential evolution, simulated annealing, default, and Nelder Mead optimization methods. All these methods attempt to find global minimum but may end up with finding a local minimum. The use of multiple methods improves the probability to find global minimum as different methods usually produce different results.

   (a) Set up variables

   Defines the fitting parameters.

(b) Input properties

Sets up the structures to be fitted, their target properties, and relative factors to weigh these properties in the objective functions.

(c) Analytical equations for EAB

Derives analytical equations for cohesive energy (E) , lattice parameter at zero pressure (A) and bulk modulus (B) for various structures to be fitted. In the Mathematica notebook, the lattice parameter is determined by targeting the pressure in the crystal to equal zero.

(d) Analytical equations for elastic constants

Derives analytical equations for elastic constants C and C44 for various structures to be fitted.

(e) Quick test

Calculates and outputs various properties using the initial guessed EAM parameters as a quick test to see if everything works.

(f) Fit setups

Sets up the objective function and all the needed constraints.

(g) Auto fit

Minimizes the objective function under the desired constraints using four different optimization methods.

8. Analysis

Performs analysis to characterize the EAM potentials.

(a) General

Performs the analysis on an arbitrary (user input) EAM potential.

(b) Initial

Performs the analysis on the initial (can be guessed or derived in previous runs) EAM potential.

(c) This fit

Performs analyses on all the four EAM potentials obtained using the four methods during this fit. Also outputs complete records that are necessary to reproduce the run.

9. Create .set EAM Potential Tables

Takes an input set of EAM parameters and writes out an MD-ready potential .set table file.

## B.4 User's input values

Get ready to run the code by coping it in your working directory. Within that directory, make a subdirectory called "files" which is the default directory that the program writes output to or reads information from. Then in the "Include Files and Utility Functions" section, assign your working directory as a character string to the argument of the function **SetDirectory**[...]. You are then ready to go. All input data that users may need to enter to perform the work are described below section by section.

1. Include Files and Utility Functions

   **Exist** − 0 for deriving analytical equations and 1 for reading in pre-derived equations from an existing file.

   The fitting is essentially to match the predicted values of cohesive energy, lattice constant, bulk modulus, and other elastic constants to the corresponding target values. Mathematica rather than FORTRAN or C++ is used because it provides symbolic operations, significantly simplifying the coding, modification of the codes, and analysis of the potentials. The code is programmed to perform the derivation of analytical equations for cohesive energy, pressure (related to lattice constant that measures crystal size at zero pressure), bulk modulus, and other elastic constants as a function of EAM fitting parameters for all structures to be fitted. Since the derivation of these equations takes time, the equations derived in previous run are saved to a file so that they are read in during subsequent runs without re-derivation. Any changes in the input data such as cutoff distance, the read in EAM .set tables, the structures to be fitted etc., change these derived analytical equations. In these cases, Exist must be reset to 0.

   **curves** − $\begin{pmatrix} \text{“}OC'' & \{1, C_s\} \\ \text{“}TE'' & \{1, C_s\} \\ \text{“}OC'' & \{C_s, 1\} \end{pmatrix}$

   "curves" is a variable used in function **eval**[...]. It allows user to define several two-dimensional curves, such as energy as a function of composition, lattice spacing as a function of composition, and Gibbs free energy of mixing as a function of composition, and to plot them during analysis of the potentials. Each row in the matrix of the curves variable defines one structure and one composition variable. In the matrix listed above, for instance, the first row is for the "OC" lattice (see the Lattices section for the lattice definition), where the occupancy fraction for the first element (Pd) on its corresponding sublattice is fixed at 1, and occupancy fraction for the second element (H) on its corresponding sublattice is the variable

composition ($C_s$) used for the property vs. composition curves.

**ncomp** – {2,1}

ncomp is also a variable used in function **eval**[...]. It lists several energy vs. composition curves predefined in the curves variable so that they will be plotted in the same figure to compare their relative stabilities. In the example listed above, curves 1 and 2 are selected in "ncomp". The first curve listed in "ncomp" (*i.e.* curve 2) will be plotted in blue.

**nGibbs** – 1

An integer representing the curve index. nGibbs is also a variable used in function **eval**[...]. It means that the nGibbs-th curve will be used for Gibbs free energy analysis. The loop variable "vT" in function **eval**[...] controls the temperatures at which the plots of Gibbs free energy of mixing are produced.

2. Pre-analysis

**DFTdataOC** – contains composition vs. (lattice constant vs. energy) data obtained from DFT calculations. The variable name "DFTdataOC" is made by users and the same name should be included in the name list variable "DFTdata" described below.

**DFTdata** – a list of character strings representing the names of variables holding DFT data of different structures.

**DFTcomp** – defines the composition variable used by each of the "DFTdata" elements (*i.e.* composition of which element).

**DFTfitpow** – the order of the polynomial used to fit the DFT lattice constant vs. energy data.

**pWolfa** – a list (corresponding to the list in "DFTdata") of plots showing experimental lattice constant as a function of composition. These plots are compared with the ones from DFT data and the ones from EAM predictions during analysis of EAM potentials.

This section plots DFT energy as a function of lattice constant at different compositions. These plots are then reduced to additional plots showing equilibrium atomic spacing $r_1$, cohesive energy $E_{\mathrm{coh}}$, and bulk modulus $B$ at different compositions. Numerical data for the equilibrium lattice constant, cohesive energy,

and bulk modulus are printed. A raw table containing composition, equilibrium atomic spacing, cohesive energy, and bulk modulus is printed out to facilitate user to copy and paste this raw table to the variable "DFTcaeb" for subsequent analysis. These data also provide the target values that need to be entered in the "Fitting ... /Input properties" section.

3. Lattices

**ntypes** – number of species.

**Pd, H** – element symbols are directly used to define structures (see "T###" below). Here the element names are cast to integers, *e.g.* Pd = 1, H = 2, etc.

**ielement** – atomic numbers for all the species.

**amass** – atomic masses for all the species.

**latcn** – $\begin{pmatrix} \text{“}OC'' & \{1,0\} & \frac{3.88486}{\sqrt{2}} \\ \text{“}OC'' & \{0,1\} & \frac{2.2840}{\sqrt{2}} \end{pmatrix}$

"latcn" variable defines structure information for all elements. The $i^{\text{th}}$ row of the "latcn" matrix defines the $i^{\text{th}}$ element. The first row in the matrix listed above, for example, defines an "OC" lattice. This "OC" lattice contains two fcc sublattices. 1,0 means that the first fcc sublattice is fully occupied by Pd atoms at an occupancy fraction of 1, and the second sublattice contains no H atoms because the occupancy fraction is 0. As a result, this defines an fcc Pd structure. The value 3.88486 is its lattice constant (*i.e.* a nearest neighbor distance of $\frac{1}{\sqrt{2}}3.88486$). Similarly, the second row of the matrix defines an fcc H with a lattice constant of 2.2840.

**rcut** – a list of all cutoff distances between atoms $i$ and $j$. The list is sequenced using a single index $ij$. The conversion from $i$ and $j$ to $ij$ follows the rule: $ij = i$ when $i = j$; $ij = i \cdot ntypes - \frac{i(i+1)}{2}$ when $i < j$; and $ij = j \cdot ntypes - \frac{j(j+1)}{2}$ when $i > j$.

**smooth** – a list of small distances ($r_s$) associated with the list of rcut for smoothing cutoff.

**Tlat** – users can easily define any lattice of their wish. All they need to do is to make up a lattice name, say ###, include this lattice name in the list defined by the variable "Tlat", and then define the lattice constant variable "A###", atom species variable "T###", and atom position

variable "P###". Here lattice constant, atom species, and atom position variables starts with A, T and P respectively, followed by the structure name. For the "OC" structure listed in the above, one can define Tlat = {..., "OC"}, and the variable names for lattice constant, atom species, and atom position will be AOC, TOC, and POC. Sublattices are naturally incorporated by defining the corresponding atom species and atom positions.

**A###** – lattice constant of structure ### in unit of the lattice spacing.

**T###** – species of atoms on the sites for lattice ###.

**P###** – normalized (by lattice constant) atom positions in lattice ###.

**IniRawC** – a list of initial compositions for the lattices that are used in preliminary analyses and tests prior to the fitting. Data in "IniRawC" are abbreviated. For instance, IniRawC = { {0.5}, {0.8} } is equivalent to IniRawC = { {1, 0.5}, {1, 0.8} }.

**r0guessed1** – a list of guessed lattice spacings for the initial structures defined by "Tlat" and "IniRawC". The data is used to determine the range of energy vs. lattice spacing curves so that the minimum energy point is included in the plots. Otherwise the data do not need to be accurate.

**r0guessed2** – a list of guessed lattice spacings for the fitted structures defined by the "EABfit" variable (see "Fitting .../Input properties" section). The data is used to determine the range of energy vs. lattice spacing curves so that the minimum energy point is included in the plots. Otherwise the data do not need to be accurate.

**Δ** – a distance range defining the lower bounds of the lattice spacing used to plot the energy vs. lattice spacing curves (the plots start at r0guessed1-Δ and r0guessed2-Δ respectively for initial and "EABfit" structures).

4. Lattice Analysis

   No inputs are required in this section. It outputs coordinates and relative neighbor distances as a function of structure, center atom species, and neighbor atom species.

5. EAM Functions

   (a) Initial parameters and initial EAM tables

Fitting is essentially an optimization of an objective function with respect to the unknown parameters. Mathematica automates the derivation of an analytical form of the objective function. This greatly facilitates the development of new potentials as many properties can be easily added to the fitting. One difficulty is that some of the known elemental potentials are expressed in tabular files. To combine such tables into alloys potential using the analytical objective function approach, these tables are first fitted to polynomial functions.

**initialPara** – sets initial EAM parameters (guessed or obtained in previous runs) that will be used for pre-testing purposes.

**ReadRestart** – reads existing .set EAM tables. The first argument of the function is an integer indicating which element the potential tables refer to. The second argument of the function is a character string indicating the file from which the tables are read.

(b) Fitting embedding energy table

**NpEmb** – the order of the polynomial function used to fit the input embedding energy table.

After execution, this section produces plots showing the fitted embedding energy as well as its first and second derivatives as a function of electron density (lines) and comparing them to the input tables (data points). The purpose is to allow users to adjust "NpEmb" to achieve satisfactory fits for function, and its first and second derivatives.

(c) Fitting electron density table

**NpRho** – the order of the polynomial function used to fit the input electron density table.

After execution, this section produces plots showing the fitted electron density as well as its first and second derivatives as a function of atom spacing (lines) and comparing them to the input tables (data points). The purpose is to allow users to adjust "NpRho" to achieve satisfactory fits for function, and its first and second derivatives.

(d) Fitting pair energy table

**NpPhi** – the order of the polynomial function used to fit the input pair energy table.

After execution, this section produces plots showing the fitted pair energy as well as its first and second derivatives as a function of atom spacing

(lines) and comparing them to the input tables (data points). The purpose is to allow users to adjust "NpPhi" to achieve satisfactory fits for function, and its first and second derivatives.

(e) Define EAM functions

No inputs are required and no outputs are produced in this section.

(f) Plot EAM functions

No inputs are required. It plots two sets of all EAM functions including embedding energy, electron density, and pair energy for all atom species. The first set of plots is without normalization, and the second set of plots is normalized.

6. Print Initial Cohesive Energy

No inputs are required. It plots energy vs. lattice spacing curves for all the structures defined by "Tlat" and "IniRawC" (see the "Lattices" section).

7. Fitting ...

(a) Set up variables

The variables used in the EAM Function section can be directly referred to here. In the EAM Function section, the embedding energy, electron density, and pair energy are defined respectively as

$$\begin{aligned} F(\rho) &= A\rho \ln[\rho] + B\rho \\ f(r) &= C \exp[-\delta r] \\ \phi(r) &= D\{\beta \exp[-\alpha(r-G)] - \alpha \exp[-\beta(r-G)]\} \end{aligned}$$

Correspondingly, we can define the following fitting parameters (null initial values): **A2, B2, C2, $\Delta$2, D22, G22, $\alpha$22, $\beta$22, D12, G12, $\alpha$12, $\beta$12**.

**myvar** − collects all fitting parameters as one variable.

In the example in the above, we assume that the Pd EAM potential is known. As a result, the remaining unknown EAM functions would include H embedding energy, H electron density, H-H pair energy, and Pd-H pair energy. The corresponding unknown parameters are A2, B2, C2, $\delta$2, D22, G22, $\alpha$22, $\beta$22, D12, G12, $\alpha$12, and $\beta$12, where the numbers indicate species 1 = Pd and 2 = H. To ensure that these are indeed unknown parameters, we can assign them with ".;", which overwrites them if they were previously assigned numerical values. All these fitting parameters are collectively represented by the "myvar" variable.

(b) Input properties

**EABfit** − a variable that allows the input of target values using an easy-to-see tablular format.

The inputs include lattice type, composition, cohesive energy, bulk modulus, lattice constant (or lattice spacing), and elastic constants $C'$ and $C_{44}$. "FitEAM" also allows specification of three fitting options: =, >, and <. Only when the predicted property violates the fitting option when compared to the target value will the corresponding deviation be used in the objective function for the least square minimization. The > and < options are useful when the exact values of target properties are not clear and yet they must be bounded to ensure phase stabilities. The use of > or < also imposes less constraints than the use of =, enabling the better fit of other (more important) properties.

**Ewght** – a list of relative weights for cohesive energy in the objective function. The list corresponds to the structures to be fitted.

**Bwght** – a list of relative weights of bulk modulus in the objective function. The list corresponds to the structures to be fitted.

**Pwght** – a list of relative weights of pressure in the objective function. The list corresponds to the structures to be fitted.

**Cprimewght** – a list of relative weights of elastic constant $C'$ in the objective function. The list corresponds to the structures to be fitted.

**C44wght** – a list of relative weights of elastic constant $C_{44}$ in the objective function. The list corresponds to the structures to be fitted.

(c) Analytical equations for EAB

No inputs are required in this section. When Exist = 0 (see "Include Files and Utility Functions"), it derives analytical equations for energy, bulk modulus, and pressure for all the structures to be fitted as a function of fitting parameters. The resulting equations are stored for future use. If Exist = 1, then previously stored equations are used. This saves time for setting up the run.

(d) Analytical equations for elastic constants

No inputs are required in this section. When Exist = 0 (see Include Files and Utility Functions), it derives analytical equations for elastic constants $C'$ and $C_{44}$ as a function of fitting parameters. The resulting equations are stored in a file for future use. If Exist = 1, then the equations previously stored in a file are used. This saves time for setting up the run.

(e) Quick test

No inputs are required in this section. It outputs tables containing cohesive energy, bulk modulus, pressure, elastic constants $C'$ and $C_{44}$ for all the

structures to be fitted using the initial EAM parameters (entered in "EAM Functions/Initial parameters and initial EAM tables" section). This helps check if everything up to this point works ok.

(f) Fit setups

**vcon** − a small number that enforce the decay of the pairwise functions when atomic spacing approaches the cutoff distance (*i.e.* "vcon" is equivalent to $\varepsilon$ described in "Overview" section of this manual).

$\rho$**low#** − lower bound of electron density of species # at nominal atomic spacing.

$\rho$**high#** − upper bound of electron density of species # at nominal atomic spacing.

**constraint1** − defines ranges for the fitting parameters.

**constraint3** − defines additional constraints for functions. For example, the constraints that implement cutoff distance and for electron density bound.

This section outputs total memory that the objective function uses, the value of the objective function using the initial EAM parameters, and logical flags indicating if the initial EAM parameters satisfy each of the constraints.

(g) Auto fit

Optimization is done using four different methods: differential evolution, simulated annealing, default, and Nelder Mead optimization methods.

**preVar#** − the code is designed to have two options to start the fitting: either from scratch or from where the last run finishes. In the former case, set preVar# = {}, while in the latter case, copy the fitting parameters obtained in the last run into the preVar# variable. Here # = 1, 2, 3, and 4 corresponding respectively to the differential evolution, simulated annealing, default, and Nelder Mead optimization methods.

**iteration#** − optimization continues for this many iterations unless it reaches the accuracy requirement earlier.

**randoms4** − the random number seed used in the Nelder Mead optimization method.

This strategy to use multiple methods for the fitting is to increase the probability to find global minimum as the four methods usually produce different results (unless the same initial parameters that are used for all methods in the continuation run are already close to the global minimum point). After execution, each method outputs its time of execution, value of objective function, and all the fitted parameters.

8. Analysis

   Analysis performs a series of evaluations on the potential:

   - plots out all the potential functions
   - plots out energy as a function of lattice spacing for all the structures fitted
   - plots out the equilibrium lattice spacing as a function of composition as defined by the "curves" in the "Include Files and Utility Functions" section
   - plots out cohesive energy as a function of composition as defined by the "curves" in the "Include Files and Utility Functions" section
   - plots out in the same figure the cohesive energy as a function of composition curves selected by "ncomp" variable in the "Include Files and Utility Functions" section
   - prints out a tables containing numerical values of the fitted and target values of cohesive energy, bulk modulus, pressure, $C'$, $C_{44}$, and their associated relative weights used in the objective function
   - prints out a table containing the relaxed (*i.e.* values expected at precisely zero pressure) lattice spacing and the associated cohesive energy for all the structures fitted.

   (a) General

   **generalPara** – the "General" section performs the analysis on an input set of EAM parameters that is assigned to the variable "generalPara". So copy and paste the EAM parameters to be analyzed here. Note that only use the "General" analysis for characterizing the potential. For keeping a complete record of the fit, use "This fit" analysis instead. "This fit" and "General" perform the same analysis except that "This fit" also produces a complete record of detailed setups of the fitting, such as constraints, target values, iteration# etc., to allow the reproduction of the fit.

   (b) Initial

   No inputs are required. Initial performs analysis on the initial EAM parameters entered in EAM Functions/Initial parameters and initial EAM tables section the same way as the General section.

(c) This fit

No inputs are required. "This fit" performs analysis on all the four sets of EAM parameters obtained correspondingly from the four fitting methods. The analysis is the same as that used in "General" and "Initial" sections. However, "This fit" also outputs complete information on fitting setup parameters such as constraints, target values, iteration# etc.

9. Create .set EAM Potential Tables

**WriteEAM** – writes out MD-ready .set tables of the EAM potential. The first argument of the function is a character string representing the name of the output file. The second argument of the function is the parameters of the EAM potential. The third and fourth arguments of the function are integer numbers for setting up the electron density and the atomic spacing grids. The fifth argument is the upper bound for the electron density used in the embedding energy function.

## B.5 Trial-and-Error

Many people think that potential development is an art rather than a science. It is indeed a rare experience to get a satisfactory alloy EAM potential without the trial-and-error efforts. The things that can be varied during the trial-and-error process are listed in the following. While it may take time, the manipulation of these variables often significantly improves the fitting:

(a) cutoff distances.

(b) weights.

(c) target values.

(d) constraints.

(e) use of $>$ or $<$ fitting options instead of $=$ options or vice versa.

DISTRIBUTION:

1 Ken Keeler
Los Alamos National Laboratory
P.O. Box 1663, C934
Los Alamos, NM 87545

1 Burt Davis
Los Alamos National Laboratory
P.O. Box 1663, G780
Los Alamos, NM 87545

1 David Lohmeier
Los Alamos National Laboratory
P.O. Box 1663
Los Alamos, NM 87545

1 Brad Meyer
Los Alamos National Laboratory
500B, C934
Los Alamos, NM 87545

1 Kevin Honnell
Los Alamos National Laboratory
500B, C934
Los Alamos, NM 87545

1 Chad Schmidt
Los Alamos National Laboratory
P.O. Box 1663
Los Alamos, NM 87545

1 Michael Baskes
Los Alamos National Laboratory
P.O. Box 1663
Los Alamos, NM 87545

1 Ricardo Schwarz
Los Alamos National Laboratory
P.O. Box 1663
Los Alamos, NM 87545

1 Paul Cloessner
Savannah River National Laboratory
Bldg. 773-A
Aiken, SC 29808

1 Scott West
Savannah River National Laboratory
Bldg. 773-42A
Aiken, SC 29808

1 Dennis Fish
Savannah River National Laboratory
Bldg. 773-A
Aiken, SC 29808

1 Steven Wyrick
Savannah River National Laboratory
Bldg. 773-A
Aiken, SC 29808

1 R. Thomas Walters
Savannah River National Laboratory
Savannah River Site
Aiken, SC 29808

1 Adam Schwartz
Lawrence Livermore National Laboratory
Bldg. L-353
Livermore, CA 94551

1 Kris Winer
Lawrence Livermore National Laboratory
Bldg. L-170
Livermore, CA 94551

1 Wilhelm Wolfer
Lawrence Livermore National Laboratory
Bldg. L-353
Livermore, CA 94551

1 Christopher Krenn
Lawrence Livermore National Laboratory
Bldg. L-353
Livermore, CA 94551

1 David Gelles
Pacific Northwest National Laboratory
P.O. Box 999
Richland, WA 99352

1 Steve Robinson
760 Kartrina St
Livermore, CA 94550

1 Cook Story
2468 Windsor Pl
Livermore, CA 94550

1 Jeffrey Hoyt
Burke Science Building Room B107
Materials Science and Engineering Dept.
McMaster University
Hamilton, Ontario, Canada L8S 4L7

1 MS 0321
J.S Peery, 1400

1 MS 0346
R.S. Chambers, 1524

1 MS 0346
E.D. Reedy Jr., 1526

1 MS 0351
H.E. Fang, 1000

1 MS 0372
J. Pott, 1524

1 MS 0380
H.S. Morgan, 1540

1 MS 0824
M.R. Prairie, 1510

1 MS 0825
C.W. Peterson, 00003

1 MS 0869
L.C. Beavis, 2715

1 MS 0878
J.F. Browning, 2725

1 MS 0878
M.O. Eatough, 2725

1 MS 0878
C.S. Snow, 2725

1 MS 0885
R.J. Salzbrenner, 1820

1 MS 0886
P.G. Kotula, 1822

1 MS 1056
B.L. Doyle, 1111

1 MS 1078
W.L. Hermina, 1710

1 MS 1146
P.J. Griffin, 1384

1 MS 1146
P.J. Cooper, 1384

1 MS 1231
R.G. Spulak, 5042

1 MS 1322
J.B. Aidun, 1435

1 MS 1411
C.C. Battaile, 1814

1 MS 1411
S.M. Foiles, 1814

1 MS 1411
E.A. Holm, 1814

1 MS 1411
R.A. Roach, 1814

1 MS 1411
E.B. Webb III, 1814

1 MS 1453
J.L. Moya, 2550

1 MS 9001
P.J. Hommert, 8000

1 MS 9001
P.A. Spence, 8000

1 MS 9004
J.M. Hruby, 8100

1 MS 9014
J.O. Harrison, 8238

1 MS 9035
C.H. Cadden, 8239

1 MS 9035
D.K. Balch, 8239

1 MS 9035
C.W. Pretzel, 8239

1 MS 9035
S.F. Rice, 8239

1 MS 9042
D.M. Kwon, 8770

1 MS 9042
M.L. Chiesa, 8774

1 MS 9042
A.A. Brown, 8774

1 MS 9042
J.W. Foulk III, 8776

1 MS 9102
W.R. Even Jr., 8220

1 MS 9151
C.T. Oien, 8940

1 MS 9153
C.L. Knapp, 8200

1 MS 9153
D.R. Henson, 12101

1 MS 9154
R.G. Miller, 8240

1 MS 9159
H.R. Ammerlahn, 8962

5 MS 9159
J. Griffin, 8962

1 MS 9159
P.D. Hough, 8962

1 MS 9159
T.G. Kolda, 8962

1 MS 9159
K.R. Long, 8961

1 MS 9161
W. Bauer, 8700

1 MS 9161
S.K. Griffiths, 8700

1 MS 9161
A.E. Pontau, 8750

1 MS 9161
S.W. Allendorf, 8756

1 MS 9161
D.F. Cowgill, 8758

1 MS 9161
R.H. Nilson, 8755

1 MS 9291
B.A. Simmons, 8755

1 MS 9402
J.E.M Goldsmith, 8772

1 MS 9402
R.A. Causey, 8758

1 MS 9402
E.H. Majzoub, 8755

1 MS 9402
B.P. Somerday, 8758

1 MS 9402
K.L Hertz, 8772

5 MS 9402
X. Zhou, 8776

1 MS 9403
T.J. Shepodd, 8778

5 MS 9403
B.M. Wong, 8778

1 MS 9403
L.A. Domeier, 8778

1 MS 9403
J.R. McElhanon, 8778

1 MS 9403
B.E. Mills, 8778

1 MS 9403
A.M. Morales, 8778

1 MS 9403
A. Vance, 8778

1 MS 9403
A.J. Antolak, 8772

1 MS 9404
E.P. Chen, 8776

1 MS 9405
R.W. Carling, 8700

1 MS 9409
R.E. Jones, 8776

1 MS 9409
C.D. Moen, 8757

5 MS 9409
J.A. Zimmerman, 8776

2 MS 9018
Central Technical Files, 8945-1

2 MS 0899
Technical Library, 4536