Silicon potentials investigated using density functional theory fitted neural networks

2008 J. Phys.: Condens. Matter 20 285219

(http://iopscience.iop.org/0953-8984/20/28/285219)

IOP PUBLISHING JOURNAL OF PHYSICS: CONDENSED MATTER

J. Phys.: Condens. Matter **20** (2008) 285219 (10pp) doi:10.1088/0953-8984/20/28/285219

Silicon potentials investigated using density functional theory fitted neural networks

E Sanville, A Bholoa, R Smith and S D Kenny

Department of Mathematical Sciences, Loughborough University, Loughborough, LE11 3TU, UK

Received 21 January 2008, in final form 7 May 2008
Published 24 June 2008
Online at stacks.iop.org/JPhysCM/20/285219

Abstract

We present a method for fitting neural networks to geometric and energetic data sets. We then apply this method by fitting a neural network to a set of data generated using the local density approximation for systems composed entirely of silicon. In order to generate atomic potential energy data, we use the Bader analysis scheme to partition the total system energy among the constituent atoms. We then demonstrate the transferability of the neural network potential by fitting to various bulk, surface, and cluster systems.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Today, there exist many schemes for the development of empirical potentials for application in molecular dynamics research. These schemes range from simple two-body potentials such as the Lennard-Jones potential, through potential functions of intermediate complexity such as the glue model [1], the embedded atom method [2, 3], and the Finnis–Sinclair potentials [4], to angular dependent forms including the Stillinger–Weber [5], and Tersoff [6] potentials. Potentials [1–4] involve both a sum of two-body interaction potentials, and a potential term which is a sum of a function of the generalized coordination of each atom. The coordination potential terms are formulated in such a way as to model changes in interatomic bond strength as a result of crowding and steric hindrance among neighboring atoms. While not of great importance in the modeling of noble gases, these terms are more significant in the modeling of metallic materials, especially those with increasing covalent bonding character. The Stillinger–Weber potential replaces the coordination terms of potentials [1–4] with a three-body angular term that introduces an energetic penalty which is a function of angular deviations from an angle of known stability. This angular potential energy term attempts to model atomic orbital hybridization, in addition to atomic crowding and changes in coordination. The Tersoff potential arguably improves on this concept by replacing the angular potential terms with a set of attractive terms for each interatomic interaction that are proportional to the bond order of each interaction. This bond order is in turn defined as a function of a coordination number corresponding to the bond ij. This bond coordination number is formulated using the bond angles and distances involving bonds ij and all ik ($k \neq j$). While potentials [5, 6] have found some success in describing certain aspects of semiconductor systems, they still demonstrate serious transferability problems.

Meanwhile, cheaper availability of more and more powerful computers has led to broader applicability of *ab initio* electronic structure methods in recent years. In particular, density functional theory has experienced a boost of popularity, for physical, chemical, and materials science applications. Two decades ago, molecular dynamics simulations in which ions or nuclei moved classically in a potential field calculated by using *ab initio* methods were pioneered by Car and Parrinello [7]. These calculations bypassed the issue of empirical potential development entirely, avoiding many of the problems inherent in these potentials. This approach effectively uses the Born–Oppenheimer approximation to separate the dynamics simulation into two parts: the calculation of forces on the ions by differentiation of the electronic structure energy, and the movement of the atomic coordinates according to classical physics using a numerical integration algorithm [8]. The vast majority of computation time corresponds to the calculation of the forces on the nuclei. MD simulations can thus, in principle, arbitrarily approach *ab initio* levels of accuracy without the vast computational expenses associated with such methods, by more accurately fitting the *ab initio* potential energy surfaces to more complex parametrized models, and

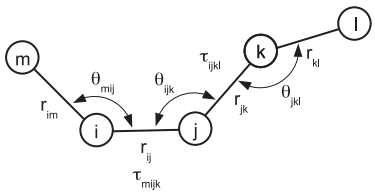

Figure 1. Input variables.

replacing the differentiation of the electronic structure energy by the evaluation of the energy from these models.

Neural networks have been used successfully to model various complex systems [9–14], without the formation of an explicit model or parametrized formula. Instead, the artificial neural network is composed of a set of interconnected nodes or neurons, which individually take a set of input values and produce an output value based on a simple formula that takes a small number of parameters. In a *feed-forward network*, the entire neural network is composed of a number of layers of nodes. The first layer is composed of input nodes, which produce the independent variables as outputs. Each node of the second layer maintains a connection to each of the input nodes with a specific weight, w^A_{QP}. The inputs to each of the second layer nodes are then simply the product of the weight with the output from each of the input nodes. This sum is then passed as the parameter of an activation function ϕ. The output from each node is then evaluated as the sum of this activation function and the bias of the node b^A_Q. A similar process is used for the third layer of nodes, using the weights w^B_{RQ}, and the outputs from the second layer, and so on. The final layer is called the output layer, and the outputs from this layer are the predicted values of the dependent variables of the model.

A major obstacle in the creation of empirical potentials has been a lack of transferability from one chemical environment to another. This problem arises because most current potentials have a fixed functional form, which is generally modeled to a subset of related chemical environments that are of interest. Unfortunately, this tends to limit their general applicability, especially to chemical systems that are far from equilibrium or far from the fitted chemical environments. Neural networks can be used to overcome this difficulty, because they require no fixed functional form. They also have the ability to model complex and nonlinear systems through a process of *training*, where the weights and biases are fitted to a data set of interest. This fitting process can be achieved by the method of *backpropagation*, where the gradient of the square of the energy error is calculated by differentiation. This energy residual is minimized with respect to the weights and biases.

If neural networks are to be used for atomistic energetics calculations, a method then needs to be created to generate input values from a set of atomic coordinates. One such method is presented in this paper, which uses interatomic distances, angles, and torsion angles to generate these input values.

2. Neural network

Previously, we have reported a neural network potential, similar in design to the present algorithm, except with fewer

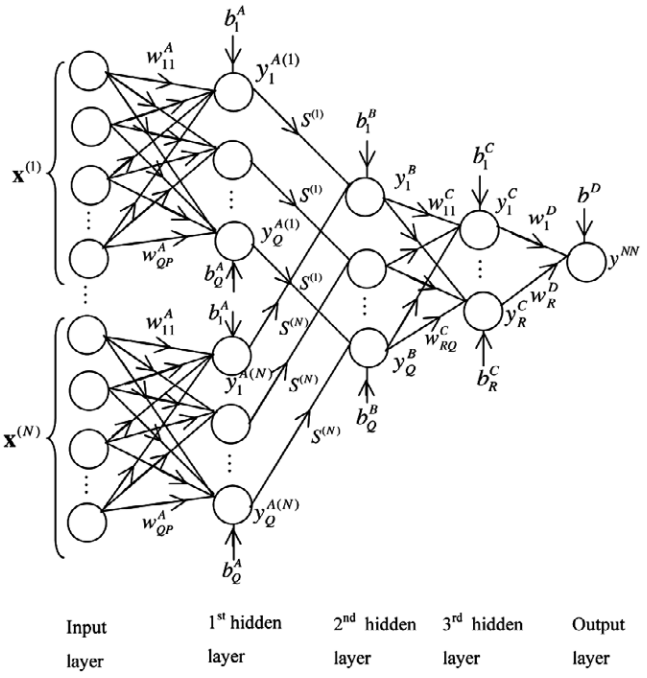

Figure 2. Neural network model.

input variables, that was fit to a smaller subset of tight-binding energetics data [13]. The present algorithm improves on this design by fitting a larger subset of *ab initio* data, using some additional input variables.

Recently, an algorithm for generating an interatomic potential for silicon using a neural network was reported [15]. This neural network used a set of *symmetry functions* to characterize the environment of each atom, and used these symmetry functions as input to the neural network. An advantage of this network design was that the number of input variables for each atom was constant, and independent of the geometry of the neighborhood of the atom.

2.1. Input layer

In this paper, we present a different method for generating a many-body interatomic potential from *ab initio* data, using a feed-forward network topology. In contrast to previously reported network topologies, this network has a variable number of input variables for each atom in the system. Additionally, the complete geometric environment of each atom is explicitly taken into account, including local bond lengths, angles, and torsion angles. Input to the neural network is composed of a series of vectors, (labeled as $\mathbf{x}^{(1)}$ to $\mathbf{x}^{(N)}$ in figure 2). Each of these input vectors corresponds to a five-atom chain found within the system, and has a set of P components. In the neural network presented here, $P = 13$, meaning that each vector has a total of 13 variables, nine of which completely describes the geometry of the five-atom chain: r_{ij}, r_{jk}, r_{kl}, r_{im}, $\cos\theta_{ijk}$, $\cos\theta_{jkl}$, $\cos\theta_{mij}$, τ_{ijkl}, and τ_{mijk}, (see figure 1). In addition to these geometric input variables, the following additional variables are used:

$$N_i = \sum_j S_{ij} \tag{1}$$

$$N_j = \sum_{k \neq i} S_{jk} \tag{2}$$

$$N_m = \sum_{k \neq i} S_{mk} \tag{3}$$

$$N_{\text{inputs}} = \sum_{j \neq i} S_{ij} \left(1 + \sum_{k \neq i,j} S_{jk} \left(1 + \sum_{l \neq i,j,k} S_{kl} \times \left(1 + \sum_{m \neq i,j,k,l} S_{im} \right) \right) \right), \tag{4}$$

where i, j, k, l, and m are the indices of atoms, and S_{ij} is the bond screening factor between atoms i and j (described below). The sums with index j are over the neighbor list associated with atom i, while the sums with index k are over the neighbor list associated with atom j, and so on. The neighbor list was generated with a cutoff distance of 8 Å. The two-center bond screening factors S_{ij} were calculated as the product of all three-center bond screening factors:

$$S_{ij} = f(r_{ij}) \prod_{k \neq i,j} S_{ikj}, \tag{5}$$

where the product is over all atoms k which are found on the neighbor lists of both atom i and atom j, and $f(r_{ij})$ is a distance screening function, (distances measured in Ångstroms):

$$f(r_{ij}) = \begin{cases} 1 & r_{ij} < 6.5 \\ \frac{1}{2}\left[\cos\left(\frac{\pi}{1.5}(r_{ij} - 6.5)\right) + 1\right] & 6.5 < r_{ij} < 8.0 \\ 0 & r_{ij} > 8.0. \end{cases} \tag{6}$$

The three-center bond screening factors S_{ikj} were calculated using the scheme of Baskes [16]. A total screening factor $S^{(n)}$ is calculated for each input vector $\mathbf{x}^{(n)}$:

$$S^{(n)} = S_{mi} S_{ij} S_{jk} S_{kl}. \tag{7}$$

All possible five-atom chains involving atom i with $S^{(n)} \neq 0$ are used as input for the neural network. The output of the neural network is the predicted energy of atom i, (figure 2).

2.2. Hidden layers

Each input vector is associated with its own individual first hidden layer, composed of Q nodes, as in figure 2. In the fitted neural network reported here, $Q = 13$. From each input vector $\mathbf{x}^{(n)}$, the outputs from the first hidden layer $\mathbf{y}^{A(n)}$ are calculated as follows. The activation function for all nodes in the network is the sigmoid function:

$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}}. \tag{8}$$

This means that the output for the qth node in the nth first hidden layer is:

$$y_q^{A(n)} = f\left(\sum_{p=1}^{P} w_{qp}^{A} \mathbf{x}_{\mathbf{p}}^{(\mathbf{n})} + b_q^{A} \right), \tag{9}$$

where w_{qp}^{A} are the first hidden layer weights, $\mathbf{x}_p^{(n)}$ are the input values, and b_q^{A} are the first hidden layer biases.

There is always the same number of nodes (Q) in the second hidden layer as there are for one of the N first hidden layers. The inputs to the second hidden layer are then calculated as the linear combination of the corresponding outputs from all N of the first hidden layers. The weight of each first hidden layer $\mathbf{y}^{A(n)}$ is the total screening factor $S^{(n)}$. The third hidden layer is then calculated using a linear combination of the outputs from the second hidden layer, with their corresponding biases, which is then fed into the sigmoid activation function. The total number of nodes R in the third hidden layer is equal to 13 for the network reported here. The output from the final node, y^{NN}, is then a scaled representation of the energy of the atom i.

2.3. Data fitting

All inputs and outputs are linearly scaled such that the entire data set occurs in the range $0.1 < x < 0.9$. The set of geometry and energy data was divided up on a per-atom basis, in which 60% of the atoms were assigned to the training set, 20% were assigned to the validation set, and 20% were assigned to the test set. All weights and biases were optimized to minimize the sum of the squares of the errors within the training set, while not increasing the error of the validation set. The neural network was composed of 13 nodes in each of the second and third hidden layers. The optimized weights and biases were calculated by minimization of the square of the error of the predicted energies, R:

$$R = \sum_{\text{data set}} (E_{\text{predicted}} - E_{\text{actual}})^2. \tag{10}$$

The method of backpropagation was used to calculate the required gradients of R with respect to each weight and bias in the neural network. Using these gradient vectors, R was minimized by using the Levenberg–Marquardt algorithm [17].

2.4. Bader analysis

One of the challenges of using *ab initio* data to fit empirical potentials is the division of the total system energy among the atoms composing the system. Here, the atoms in molecules approach is used to assign regions of space to each atom in the system [18]. These regions of space are called *atomic basins*. They are enclosed by a unique set of surfaces, through which the flux of the electronic charge density gradient is zero. The atoms in molecules scheme has several advantages when used to allocate properties among atoms in a system. The scheme can be performed is relatively independent of the atomic basis sets used for the *ab initio* calculation. In fact, these basins can be calculated using experimental charge densities, or charge densities derived from calculations that use a plane wave basis. In this paper, we use the Bader DFT energy allocation method previously reported by us [19]. The band energy of the system was allocated among the atoms using a Mulliken-like scheme, as previously described. The double-counting energy terms, which are dependent only upon the density of the system, and therefore are not directly related to the localized atomic basis functions, was then partitioned among the atoms by integrating it over each atomic basin.

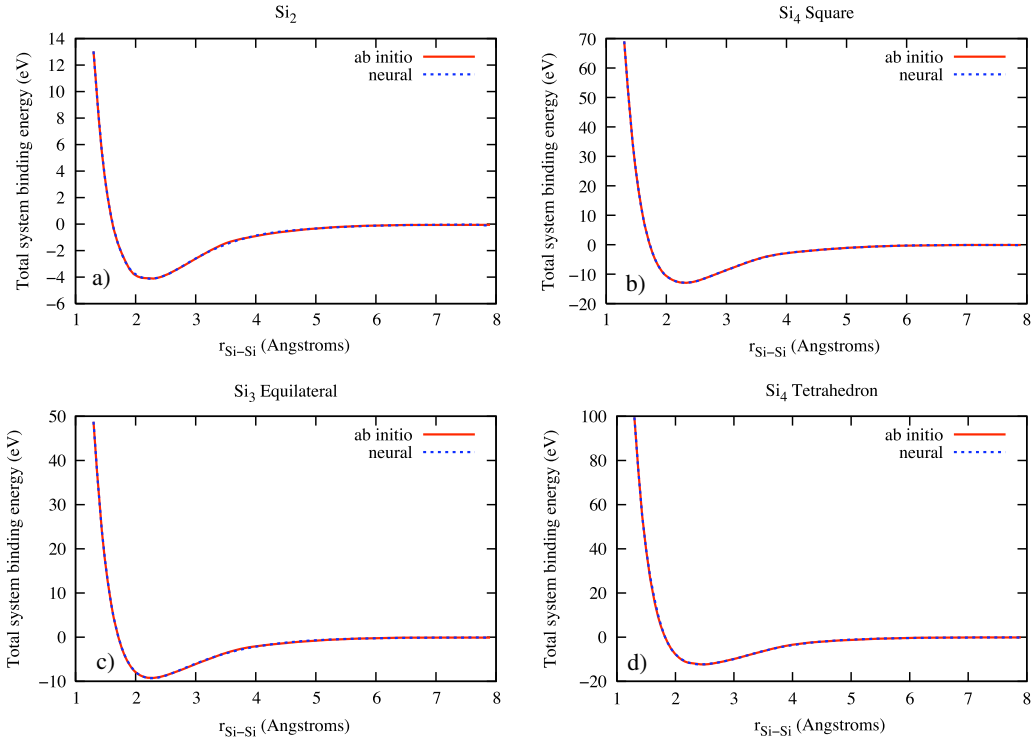

Figure 3. Neural network fits for symmetric clusters: (a) Si_2, (b) Si_4 square, (c) Si_3 equilateral triangle, (d) Si_4 tetrahedron. Energetics for the clusters are compared for r_{Si-Si} between 1.2 and 8.0 Å.

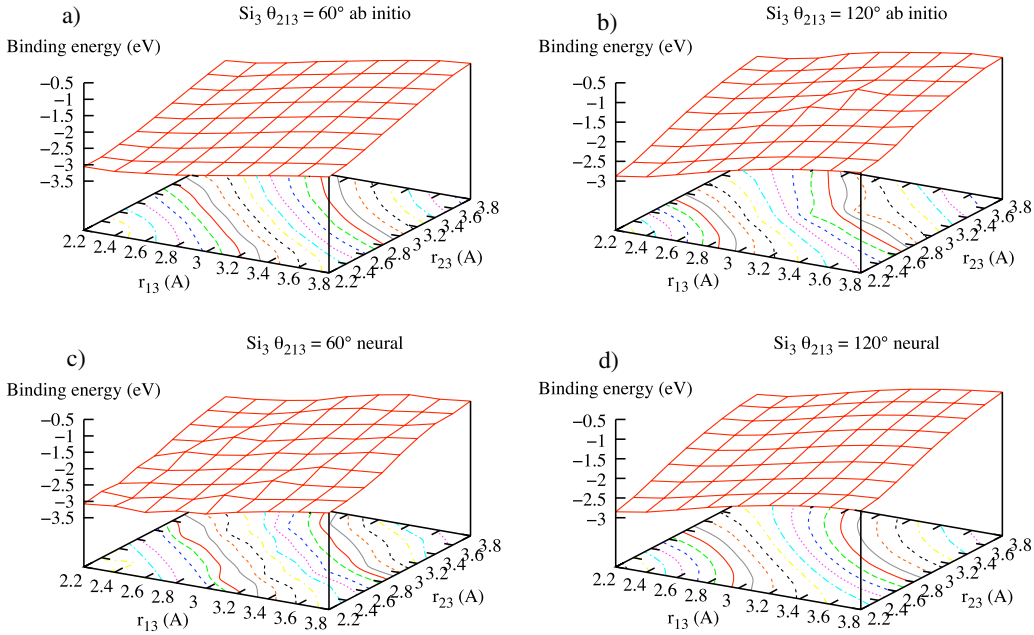

Figure 4. Neural network fits on potential energy surfaces for asymmetric Si_3 clusters: (a) and (b) show *ab initio* calculations for the potential energy surface with one angle of the triangle held constant, (60° and 120°, respectively). (c) and (d) show the same potential energy surfaces as predicted by the neural network.

3. Results

The results of the neural network fitting procedure are shown in figures 3–9. The weights and biases for the final fitted network are listed in tables A.1–A.4. The data set contained 260 single point symmetric clusters, for Si_{2-4}. For the Si_4 clusters, both squares and tetrahedra were used in the data set, (figure 3). Additionally, 225 asymmetric Si_3 clusters were used, with interatomic angles varying between 0° and 180°, and interatomic distances ranging between 1 and 8 Å, (figure 4). A set of 330 distorted bulk diamond systems

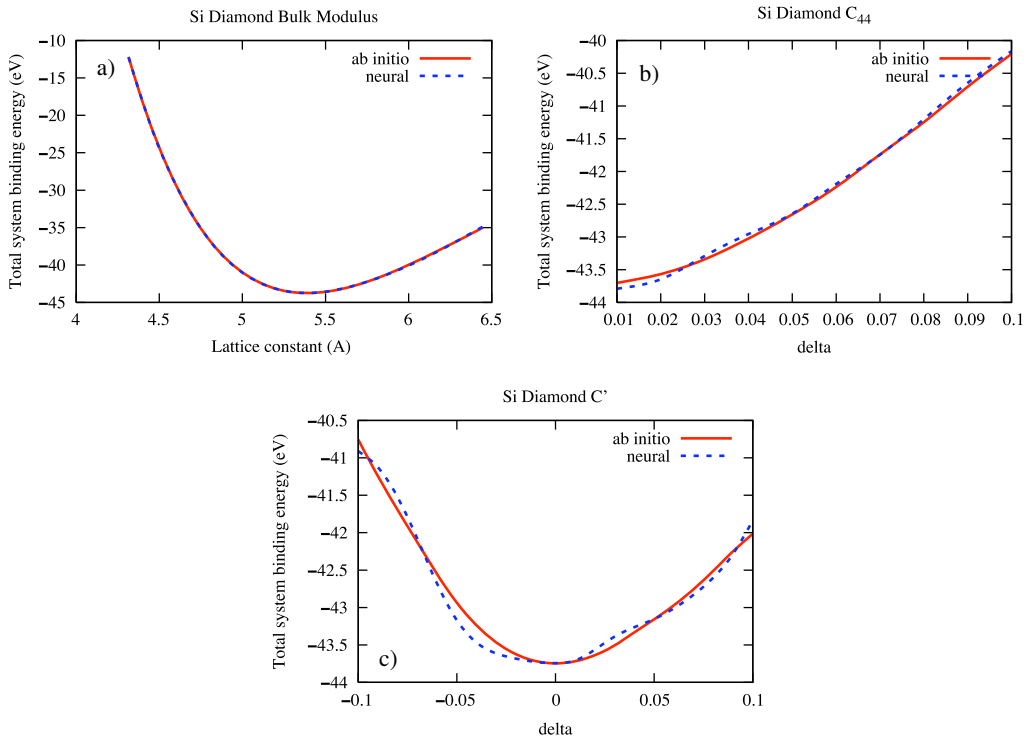

Figure 5. Neural network fits for silicon diamond lattice distortions: (a) hydrostatic (bulk modulus), (b) C_{44}, and (c) C'.

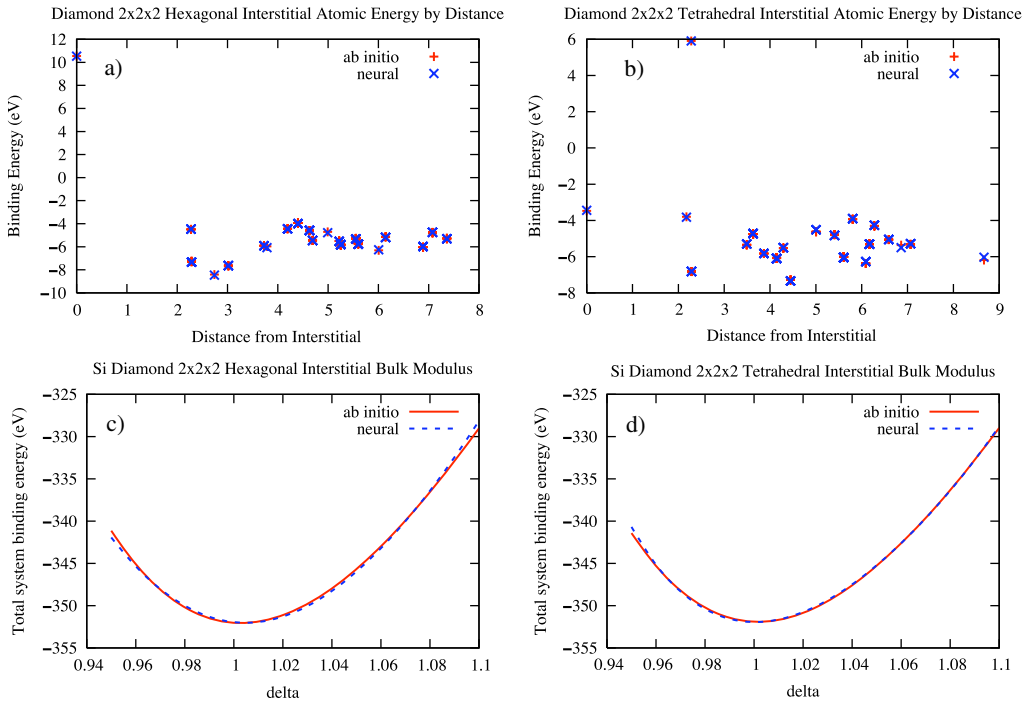

Figure 6. Neural network fits for silicon diamond lattice interstitials. (a) and (b) Energies of individual lattice atoms are plotted with respect to their distance from a hexagonal and tetrahedral interstitial, respectively. (c) and (d) The $2 \times 2 \times 2$ unit cells undergo hydrostatic compression.

were used, with varying hydrostatic compression strains, C' distortion strains, and C_{44} distortion strains. The predicted and *ab initio* energies of these bulk diamond systems are shown in figure 5. Both hexagonal and tetrahedral interstitial systems were present within the data set, including 32 systems composed of a $2 \times 2 \times 2$ unit cell with various cell parameters. The resulting stress–strain curves, and plots showing the predicted and actual energies of the lattice silicon atoms with respect to their distances from the interstitial are shown in figure 6. Relaxed 2×1 and 2×2 reconstructed (001) model systems were also included, as slabs of 14-atom thickness. A set of 20 2×1 reconstructed (001) surface slabs with varying

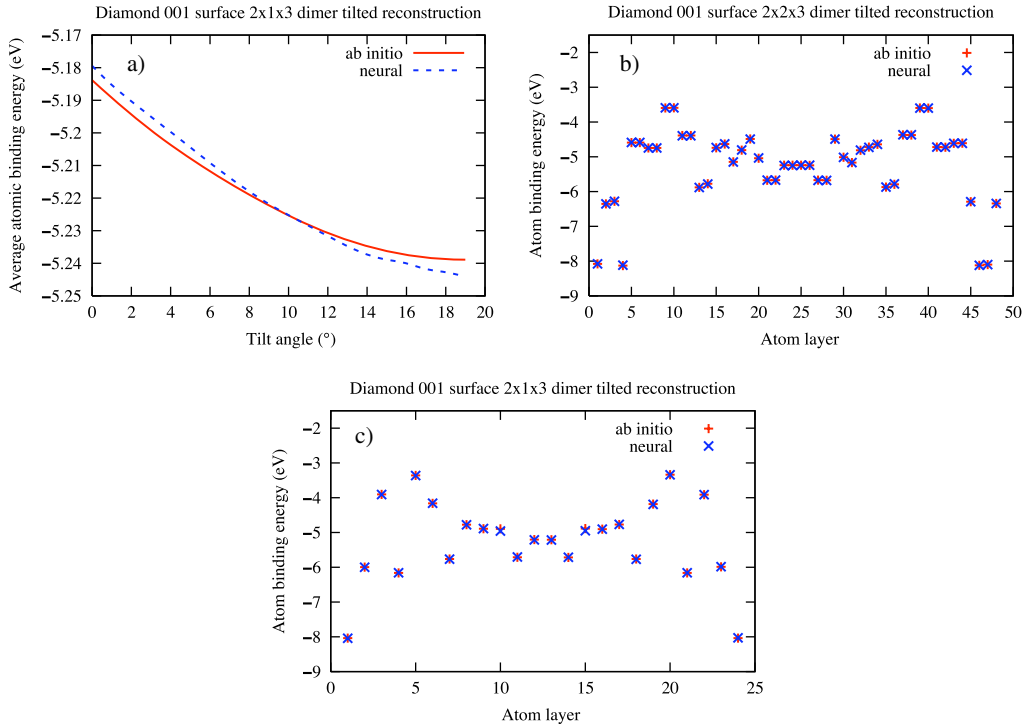

Figure 7. Neural network fits for diamond (001) 2 × 1 and 2 × 2 reconstructed surfaces. (a) Energy is plotted against dimer tilt angle, with the tilt angle varying from 0° to 19°. (b) and (c) Energies of individual atoms are plotted with respect to their depth into the slab, for the 2 × 2 and 2 × 1 reconstructions, respectively.

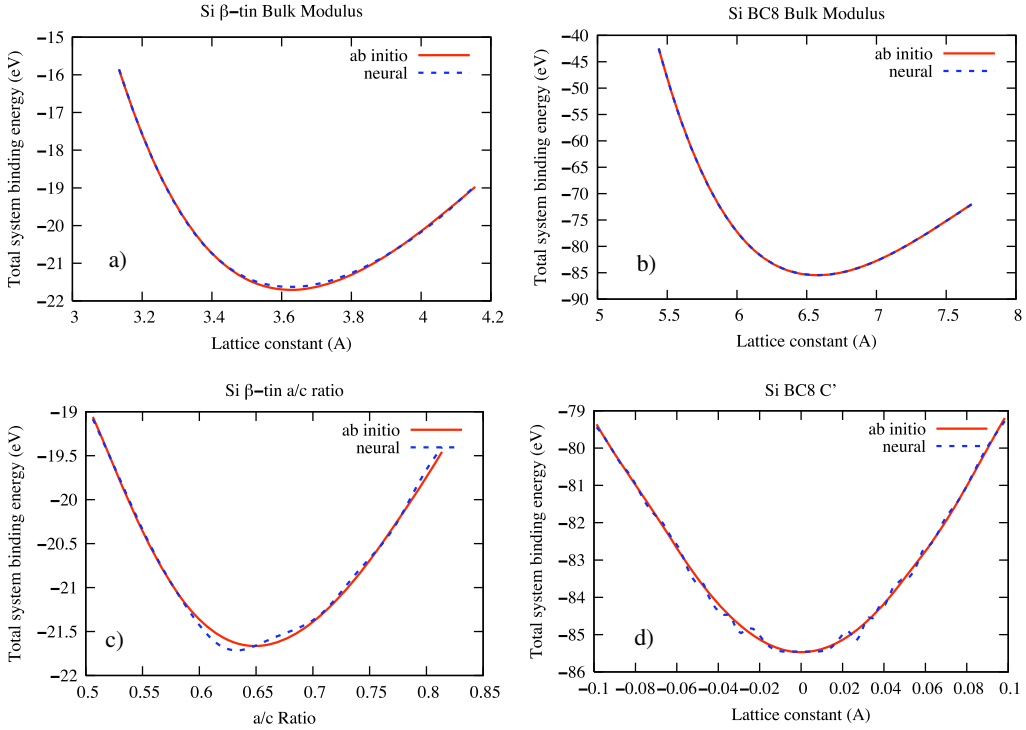

Figure 8. Neural network fits for silicon β-tin and BC8 lattices: (a) and (b) hydrostatic compression, (c) β-tin a/c ratio, and (d) BC8 C' distortion curves.

dimer tilt angles was also included in the data set, (figure 7). The tilt angles ranged between 0° and 19°, with 19° being the energy minimum. Finally, silicon β-tin, BCT5, and ST12 systems were included with various hydrostatic compression strains, and a/c ratios were included in the data set, along with BC8 systems with both hydrostatic compression strains, and C' strains, (figures 8, 9). The optimized neural network weights and biases are given in the appendix.

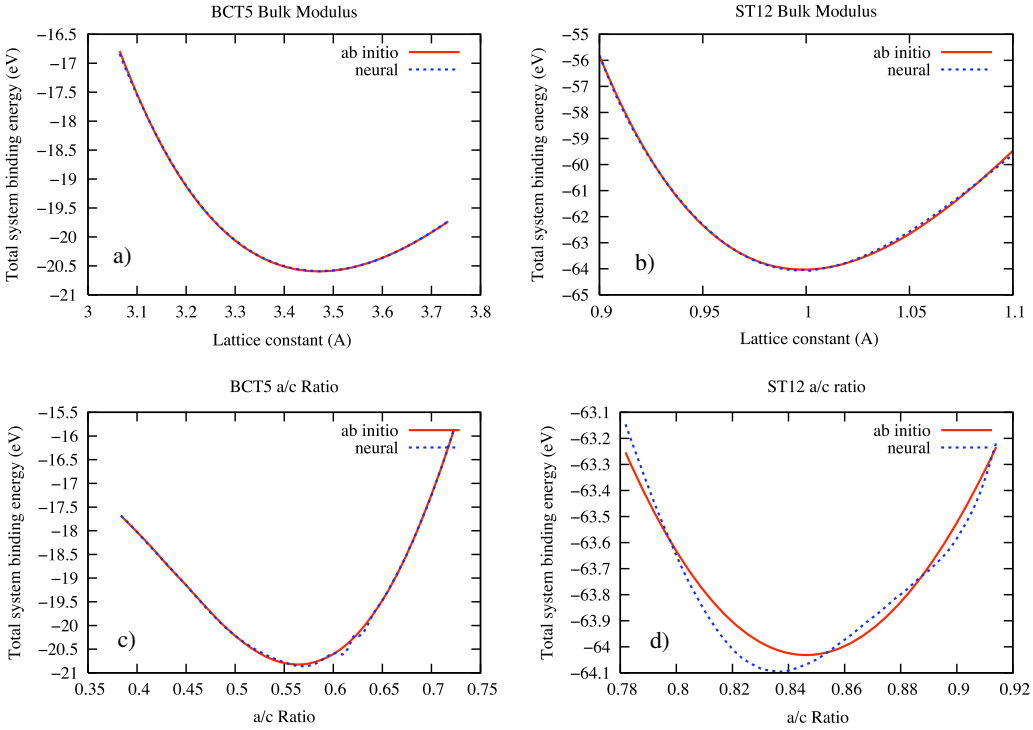

Figure 9. Neural network fits for silicon ST12 and BCT5 lattices: (a) and (b) hydrostatic compression, (c) and (d) a/c ratios.

4. Discussion

The results have shown that neural networks can be used to fit the potential energy surfaces associated with a diverse set of *ab initio* data for silicon. Small clusters, distorted Si-diamond, β-tin, BCT5, BC8, and ST12 unit cells, interstitial defects, and reconstructed surfaces have been fit using the screened neural network scheme described in this paper. The energetics of silicon clusters, bulk, and diamond (001) surfaces have been successfully modeled simultaneously with the same novel potential, based upon the results of *ab initio* density functional theory calculations. Additionally, the energy dependence upon the silicon diamond (001) surface dimer tilt angles has been fit successfully. It has been demonstrated that the neural network design is capable of fitting not only the total system energies, but also the individual atomic energies from the energy partitioning scheme. The neural network design allows the use of a variable number of inputs, by using a screening factor to combine the first hidden layer nodes for each of a set of input vectors. Each input vector fully describes the geometry of a five-atom chain containing the atom of interest. The first hidden layer nodes for each input vector are combined using a screening factor which is the product of the screening factors associated with each of the four interatomic bonds within the five-atom chain. The interatomic screening factors are products of a distance screening function, and an atom screening function previously reported [16]. The distance screening function ranges from 0 to 1, and has a smooth first derivative. Due to the fact that the geometry is characterized in terms of only interatomic distances, angles, and torsion angles, the input representation is independent of the axes used to describe the atomic coordinates, and is invariant to rotation of the system. Due to the nature of the screening factors, the input vectors are continuous and smooth with respect to movement of atoms within the system, which is a necessary condition for obtaining smooth potential energy curves from the neural network's output.

Two limitations of the neural network design are the large number of input parameters required for some systems, (particularly the BC8 systems), and a lack of the ability to extrapolate to some system geometries outside of the training set. The major reason for the large number of input parameters is the redundancy of the input data, as each interatomic distance, angle and torsion can often be present several times in the input data. This is because each bond, angle and torsion angle can be part of several different five-atom chains. The lack of extrapolation ability is an inherent limitation of any neural network, and can be remedied by taking systematically larger sets of data into the training set.

5. Future prospects

Despite these limitations, the neural network potentials could be useful to obtain *ab initio* accuracy at a small fraction of the computational cost. Forces can easily be calculated by differentiating the total system energy with respect to the coordinates of the atoms in the system. This allows the possibility of molecular dynamics simulations based upon the neural network potential. Additionally, the network can be generalized to multi-component systems. This would be accomplished most easily by adding a set of five additional input variables, which represent the element of each atom in the chain, and by fitting interatomic potentials between multiple different elements.

Appendix. Neural network parameters

Table A.1. Biases.

	Layer 1	Layer 2	Layer 3	Layer 4
b_1	−24.194 887 57	−2.453 917 947	−1.731 891 107	235.821 4374
b_2	7.855 650 659	−2.666 861 068	−83.914 995 43	
b_3	−28.209 536 58	0.663 331 165	−5.506 684 641	
b_4	−6.000 666 175	1.270 167 597	−2.917 664 551	
b_5	1.355 351 589	−1.743 372 541	14.581 724 57	
b_6	-4.93×10^{-2}	0.115 779 703	30.864 429 75	
b_7	65.306 090 95	0.984 555 389	−12.387 800 15	
b_8	13.872 303 91	−1.008 096 389	−119.012 648 7	
b_9	14.606 556 81	0.250 268 36	−38.052 562 82	
b_{10}	15.713 059 35	0.346 107 045	−6.666 571 419	
b_{11}	24.438 237 84	−0.215 688 631	4.408 088 51	
b_{12}	−22.081 793 55	0.818 524 672	−17.278 466 27	
b_{13}	−4.677 000 448	−0.419 191 786	21.416 144 12	

Table A.2. Weights A.

w_{ij}^A	i				
j	1	2	3	4	5
1	9.461 963 084	−20.440 775 85	−19.633 911 37	−12.616 044 89	7.790 530 219
2	−6.476 773 757	1.009 275 34	−22.777 405 46	−2.283 720 521	−4.905 044 473
3	15.359 785 91	−8.314 029 502	24.877 780 28	3.111 485 278	−17.497 874 11
4	−1.845 622 721	−4.540 684 498	4.987 401 41	0.179 084 083	−3.956 729 862
5	0.483 804 234	−6.713 045 382	7.035 476 613	0.820 614 138	12.311 569 97
6	18.406 941 78	−3.454 207 762	17.055 1604	4.701 254 12	−10.401 318 28
7	4.622 497 864	12.603 485 26	−21.220 830 72	7.443 793 936	−12.332 928 49
8	−4.517 640 107	−7.258 905 479	−25.979 942 24	−4.566 760 786	−0.359 210 81
9	3.061 970 2	−3.757 521 144	3.090 223 106	−4.128 690 031	7.076 648 59
10	39.911 853 25	3.579 947 725	56.630 975 37	4.514 400 644	27.174 201 49
11	0.425 452 957	−13.754 607 26	−32.767 092 82	−5.658 170 093	−20.975 244 41
12	−17.985 576 69	24.829 496 78	28.194 0987	8.363 833 038	0.531 683 196
13	−56.463 018 59	3.806 683 698	−29.543 670 51	4.670 613 522	−12.605 866 24
j	6	7	8	9	10
1	−3.395 954 035	81.189 947 78	−21.350 328 85	22.947 844 55	1.780 084 99
2	0.650 159 733	84.242 3039	1.418 745 777	2.233 182 86	0.834 558 921
3	14.673 108 31	−186.555 2164	−8.615 825 601	−2.665 803 274	−38.517 793 56
4	14.678 560 35	−15.714 861 21	−0.957 659 426	−4.400 316 504	−3.932 550 473
5	−1.708 907 626	−47.176 989 04	1.455 487 292	−9.039 747 044	−10.428 956 37
6	7.59×10^{-2}	18.023 235 91	−17.726 855 53	3.583 088 661	−0.882 705 823
7	7.521 569 215	9.397 421 886	6.119 598 278	−9.871 760 612	12.836 997 94
8	13.615 704 7	−13.427 837 13	−5.180 494 846	4.602 055 195	−2.419 326 148
9	0.587 780 948	−1.911 801 507	4.757 399 246	−7.134 793 967	−20.601 468 23
10	−31.444 290 51	−2.974 290 699	4.645 659 378	−2.834 087 738	−7.453 378 403
11	19.365 009 56	−14.342 854 15	−13.084 810 31	6.245 148 008	2.152 761 505
12	−18.042 705 21	7.886 091 286	6.986 650 979	−10.499 1478	−0.105 569 345
13	25.250 26	−34.481 286 15	20.227 315 03	−8.975 026 859	−0.822 776 451
j	11	12	13		
1	2.688 011 573	−17.443 693 95	−4.262 926 602		
2	−19.865 393 55	−3.759 714 138	2.669 316 86		
3	0.951 869 084	−17.147 613 05	12.663 829 83		
4	5.046 154 211	−10.214 290 13	−7.081 107 604		
5	−13.867 714 43	13.896 529 85	4.196 947 351		
6	−14.178 2213	26.169 745 13	7.017 171 559		
7	−15.433 874 03	−24.839 773 32	10.620 387 76		
8	−12.548 791 86	−12.407 586 33	12.762 777 02		
9	−5.412 492 575	−30.784 694 33	−1.405 575 552		
10	13.368 177 28	63.072 846 15	−14.008 482 88		
11	−27.134 1802	−41.952 190 27	−4.688 905 298		
12	13.153 582 59	45.661 910 43	4.882 555 594		
13	49.905 451 05	−12.621 303 76	12.612 754 42		

Table A.3. Weights C.

w_{ij}^C	i				
j	1	2	3	4	5
1	−1.202 023 323	−35.525 194 25	−11.457 820 68	−2.241 729 699	15.828 193 16
2	−1.155 470 927	−94.564 796	0.199 819 673	−4.920 619 951	49.389 888 53
3	−1.584 969 915	113.305 8827	−2.918 013 493	−4.774 123 161	−166.833 1578
4	−0.893 163 571	169.462 8138	3.414 866 597	8.459 631 626	−54.594 542 96
5	−0.619 151 556	11.734 5258	1.394 553 633	1.793 400 921	58.398 9406
6	0.473 042 635	52.986 779 48	1.898 383 384	2.348 990 355	14.673 583 48
7	−0.363 095 154	−51.564 667 02	1.198 738 757	0.656 264 325	74.832 300 22
8	−0.944 699 692	−79.512 631	0.891 859 259	3.217 610 689	117.360 1142
9	−0.370 810 23	48.096 740 74	0.754 604 515	1.050 526 286	121.235 384
10	−5.028 041 395	−271.268 430 7	−12.798 1033	−19.580 108 18	18.301 251 28
11	−1.112 884 2	−14.612 977 61	−3.360 990 006	−9.473 884 072	−19.171 672 98
12	-4.75×10^{-2}	49.917 152 88	5.75 166 4225	6.076 556 42	−27.886 009 29
13	−2.338 650 262	25.058 321 49	−9.956 172 095	−20.101 004 26	−112.136 246 1
j	6	7	8	9	10
1	−34.660 699 43	−80.228 115 38	−102.608 3295	26.683 911 54	−2.898 591 684
2	−33.475 305 27	−24.147 375 88	18.339 144 64	−9.284 152 879	8.246 078 896
3	−43.990 856 72	56.115 794 45	−138.143 684 3	−16.134 153 65	7.771 869 081
4	1.634 132 811	−63.281 9568	112.956 9836	46.674 888 73	−65.090 057 38
5	110.463 475 8	−24.494 214 05	93.331 842 54	15.214 349 78	−47.504 767 92
6	18.477 810 52	122.444 201 7	−26.880 907 31	−34.266 784 85	37.210 181 87
7	10.211 576 66	−33.457 813 45	98.506 254 91	15.101 910 19	−13.037 141 03
8	22.589 518 38	36.207 638 57	103.283 194 7	−18.650 546 13	23.229 895 85
9	−32.252 006 02	−28.467 299 92	110.284 7001	13.329 999 43	−32.115 649 24
10	89.750 8315	95.584 812 01	50.186 2502	25.808 523 72	56.036 748 22
11	12.869 291 36	−37.352 155	−42.085 924 3	−20.972 206 63	44.212 654 07
12	−47.511 649 14	−7.364 238 907	−47.426 586 48	23.301 374 44	−7.038 400 094
13	−75.384 452 41	−61.170 403 37	−79.597 4037	−10.872 810 86	16.865 051 73
j	11	12	13		
1	33.709 725 54	−55.182 761 64	−39.261 025 57		
2	−15.572 271 85	−48.059 747 16	20.946 300 83		
3	−45.706 299 19	128.642 1196	46.545 436 31		
4	22.152 446 41	14.962 566 14	−24.996 826 86		
5	−62.685 881 72	18.745 860 01	64.942 634 42		
6	−69.546 883 73	−26.511 290 92	74.094 077 19		
7	19.201 902 94	−27.114 555 64	−21.084 0884		
8	−2.059 265 461	84.180 585 81	2.501 332 706		
9	1.94×10^{-3}	124.132 1055	4.362 968 182		
10	20.230 364 03	−40.107 830 17	−74.451 362 89		
11	47.353 350 92	−101.453 743 3	−47.739 023 03		
12	−22.768 286 54	−108.451 955 2	24.239 7533		
13	48.196 704 58	5.177 957 459	−46.323 217 64		

Table A4. Weights D.

Weight	Value
$w_{1,1}^D$	−1.581 981 887
$w_{1,2}^D$	−81.172 3725
$w_{1,3}^D$	−11.255 384 92
$w_{1,4}^D$	−20.835 658 31
$w_{1,5}^D$	−73.536 719 11
$w_{1,6}^D$	−3.092 725 528
$w_{1,7}^D$	−9.683 353 238
$w_{1,8}^D$	−47.072 683 77
$w_{1,9}^D$	−104.408 3203
$w_{1,10}^D$	−89.939 111 01
$w_{1,11}^D$	−55.574 878 94
$w_{1,12}^D$	−6.479 432 706
$w_{1,13}^D$	−53.408 407 23

References

[1] Ercolessi F, Parrinello M and Tosatti E 1988 *Phil. Mag.* A **58** 213
[2] Daw M S and Baskes M I 1984 *Phys. Rev.* B **29** 6443
[3] Foiles S M, Baskes M I and Daw M S 1986 *Phys. Rev.* B **33** 7983
[4] Finnis M and Sinclair J 1984 *Phil. Mag.* A **50** 45
[5] Stillinger F and Weber T A 1985 *Phys. Rev.* B **31** 5262
[6] Tersoff J 1988 *Phys. Rev.* B **37** 6991
[7] Car R and Parrinello M 1985 *Phys. Rev. Lett.* **55** 2471
[8] Verlet L 1967 *Phys. Rev.* **159** 98
[9] González-Romera E, Ángel Jaramillo-Morán M and Carmona-Fernández D 2007 *Comput. Ind. Eng.* **52** 336
[10] Bhadeshia H 1999 *ISIJ Int.* **39** 966
[11] Metzbower E, deLoach J, Lalam S and Bhadeshia H 2001 *Sci. Technol. Weld. Join.* **6** 116
[12] Shi X, Wang L, Kariuki N, Zhong C and Lu S 2006 *Sensors Actuatos* B **117** 65

[13] Bholoa A, Kenny S D and Smith R 2007 *Nucl. Instrum. Methods Phys. Res.* B **255** 1
[14] Hobday S, Smith R and BelBruno J 1999 *Modelling Simul. Mater. Sci. Eng.* **7** 397
[15] Behler J and Parrinello M 2007 *Phys. Rev. Lett.* **98** 146401
[16] Baskes M 1997 *Mater. Chem. Phys.* **50** 152
[17] Levenberg K 1944 *Q. Appl. Math.* **2** 164
[18] Bader R F W 1994 *Atoms in Molecules: A Quantum Theory* (Oxford: Clarendon)
[19] Sanville E, Kenny S D, Smith R and Henkelman G 2007 *J. Comput. Chem.* **28** 899