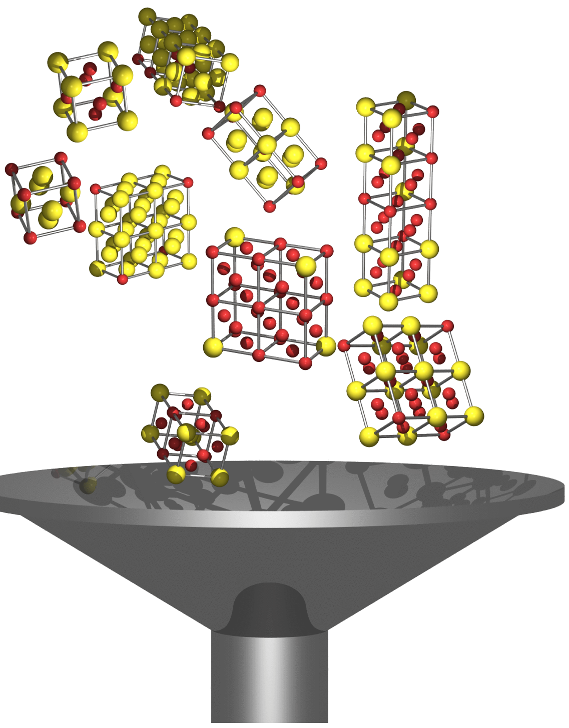
Hands-On Tutorial on *Ab Initio* Molecular Simulations

Berlin, July 12 – 21, 2011



Tutorial IV: Multiscale modeling of configurational energetics Manuscript for Exercise Problems

Prepared by Volker Blum, Gus Hart, Norina Richter

July 18, 2011

Title image: Ralf Drautz, Bochum

Background

This section gives the physical and mathematical background again (with equations) that you heard in the morning talk. You can read it, of course, but you may want to just skip ahead to "The present tutorial" and begin immediately with the actual exercises.

Multiscale modeling

First-principles simulations are powerful tools, but even in the best of cases, their reach is limited today to perhaps a few thousand atoms in a molecule or periodic supercell. Likewise, the time scales accessible by direct molecular dynamics (see tutorial number 5, tomorrow) to track atomic motion, rare events (like diffusion jump) or even the "simple" path of a system from a non-equilibrium conformation to its equilibrium order are typically outside the direct reach of the method.

Assuming the Born-Oppenheimer approximation (static nuclei) and also assuming that the electrons are more or less in their ground state, we know that at least the energy landscape which underlies most materials properties can be written as a simple function of the nuclear coordinates $\{\boldsymbol{R}_I\}$ (for I=1,...,M atoms):

$$E \equiv E_{\mathrm{BO}}^{\mathrm{gs}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M) \tag{1}$$

Of course, implicitly each nuclear position $\boldsymbol{R}_I$ is additionally associated with a specific element type, given by the atomic number Z_I.

Wouldn't it be good if $E_{\mathrm{BO}}^{\mathrm{gs}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M)$ were smooth? Of course we know that this is sometimes not the case, but still, for many cases, $E_{\mathrm{BO}}^{\mathrm{gs}}$ is a simple function of $3M$ nuclear coordinates. This function determines the ground state order, dynamics, statistical mechanics, and thermodynamics of any system at reasonable temperatures. If we could precompute this entire function in a closed, parameterized form, we could later extract the entire statistical mechanics, dynamics etc. of the system in a much faster way than by solving the Kohn-Sham equations.

Likewise, we know that at some scale, and for some problems, even the exact underlying atomic structure itself becomes irrelevant. For instance, an engineer does not need to know where the atoms in a bridge are to know how much the bridge will bend under the weight of a truck. Knowledge of the elastic properties of the supporting beams, together with the material they are made of, is sufficient—as long as the response of the material to elastic deformations is known. We can still calculate the microscopic parameters of a continuum model *if* we know the general form $E_{\mathrm{BO}}^{\mathrm{gs}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M)$ well enough.

In this fashion, if we had a set of workable physical models at all length and time scales, we could compute the microscopic parameters of the larger scale model from the next scale down, and thus be done with a first-principles model of the world as a whole.

Obviously, in this general form, a simple enough, generic parameterization $E_{\mathrm{BO}}^{\mathrm{gs}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M)$ remains a pipe dream to this day, and most likely will always remain so. However, if we restrict our ambitions to a more specific set of systems where we know certain properties in advance, we can still proceed and build an appropriate "multiscale" hierarchy of models at different length and time scales. An example of the first step of such a multiscale model is what this tutorial is about.

Binary alloys: Configurational energetics on a lattice

A classic example of a multiscale model is the *cluster expansion method.* In many materials, particularly (but not only) many metal alloys, there are distinct phases that differ only by the arrangement of individual elements on a lattice, but the underlying spatial lattice (bcc, fcc, etc.) remains in principle the same for several of these phases. (For a cartoon example, skip ahead to Fig. 2. Each of the structures shown in the figure is merely a different configuration on a square lattice.)

If the underlying lattice is known, we know the actual positions $(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M)$ in principle, we just don't know which kind of atom sites on each site—we just don't know the configuration. E becomes a simple function of the occupation of these sites by the different elements (the *configuration*),

$$E^{\text{gs}}_{\text{BO}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M) \to E_{\text{conf}}(Z_1, \ldots, Z_M) \quad . \tag{2}$$

In the particularly simple case of a binary alloy with only two element types, A and B, we do not even need to record the occupation of each site by different Z. Instead, we can further reduce the problem to spin-like variables σ_I, where σ_I=+1 if site I is occupied by element A, and $\sigma_I = -1$ if site I is occupied by element B. (Peek ahead to Fig. 1 for a picture.) You will note that we have now also (implicitly) thrown out any local lattice relaxations due to different occupations $\boldsymbol{\sigma} \equiv (\sigma_1, \ldots, \sigma_M)$ of the lattice, any temperature dependence of the internal energy E, and perhaps similar details. However, as long as there is a unique correspondence between the sites of a hypothetical, fixed lattice and the actual relaxed structure (or average of thermal positions) of a given configuration, a unique correspondence

$$E \equiv E(\sigma_1, \ldots, \sigma_M) \tag{3}$$

still exists. We will come back to this issue below.

The nearest-neighbor Ising model

The problem of different occupations of a fixed lattice is, of course, an old one, most famously addressed by Ising for the case of a spin system with nearest-neighbor pair interaction energies J_{2-1}, where the "2" denotes a pair of lattice sites (as opposed to a single site, triple, quadruple, etc., of sites) and the "1" denotes the shortest type of pair of lattice sites (as opposed to the second shortest, third shortest, etc.). In that case, we can write:

$$E(\sigma_1, \ldots, \sigma_M) = E^{\text{Ising}}(\sigma_1, \ldots, \sigma_M) = J_0 + J_1 \sum_I \sigma_I + J_{2-1} \sum_{I=1}^{M} \sum_{J \text{NN to} I} \sigma_I \sigma_J \quad . \tag{4}$$

In this case, J_0 is a kind of average energy of the entire lattice (a constant offset). The sums in the pair term drop to zero if the number of like and unlike neighbors is the same on average across the entire lattice (like in Fig 1), and becomes maximal (minimal) when the number of like (unlike) pairs becomes maximal on average across the entire lattice. The term involving only single sites only counts the overall number of atoms A and B on the lattice, accounting for possibly different total energies of atom types A and B.

Obviously, there are only three parameters in Eq. (4): J_0, J_1, and J_{2-1}. If this equation were an exact description of a real alloy systems, we could therefore determine these parameters completely by computing the first-principles total energies of only three arbitrary configurations on a lattice: For example, (i) the lattice occupied by pure A, (ii) the lattice occupied by pure B, and (iii) one arbitrary "mixed" configuration $\boldsymbol{\sigma}_{\text{fit}}$ (composition A_xB_{1-x}, $0< x <1$) with both elements present on the lattice. The energies of all other structures would then follow from the simple sum in Eq. (4).

This simple model would already define a multiscale model. Unfortunately, there is no reason for Eq. (4) to be exact in real life. So, the real first-principles energy of any configuration other than $\boldsymbol{\sigma}_{\text{fit}}$ would not be predicted exactly, but with some unknown error.

The generalized Ising model ("Cluster Expansion")

While a nearest-neighbor Ising model will never be exact in practice, it is sometimes useful. When it isn't accurate enough, a less severely truncated model with more a few more interactions may be useful. We call this model a cluster expansion because it includes different interaction types ("clusters") such as multiple pair interactions, triplet and quadruplet interactions and so forth.

There is a general proof that one can always *map* the configurational energies $E(\boldsymbol{\sigma})$ of all possible configurations $\boldsymbol{\sigma}$—*if* that model includes all possible types of *clusters* (also called "figures") f that can be found among the lattice sites: all inequivalent pairs, triples, quadruplets, quintuplets, etc., up to (unfortunately) the M-body interaction which includes the entire lattice. (So the untruncated expansion is not useful in a practical sense. How to truncate the expansion becomes an important consideration in practice.)

As we will be interested in periodic lattices, we will from now on denote by $E(\boldsymbol{\sigma})$ the energy of a given configuration *per lattice site* (rather than the total energy), and we will generalize Eq. (4) in the following way:

$$E(\boldsymbol{\sigma}) = E^{\text{CE}}(\boldsymbol{\sigma}) = \sum_f J_f \Pi_f(\boldsymbol{\sigma}) \tag{5}$$

This, in a nutshell, is the defining equation of a *cluster expansion* on a lattice.

- J_f denotes the "effective interaction strength" (an energy term) associated with a particular combination of lattice sites, f. (Finding these unknown coefficients in our expansion is the primary object of this tutorial.)

- The sum runs over all possible inequivalent "types" of lattice site combinations (figures) f—for example, nearest-neighbor pairs, second-nearest neighbor pairs, a nearest-neighbor triplet, etc. Examples of simple "inequivalent" figures on a square lattice are shown in Figure 1.

- Finally, $\Pi_f(\boldsymbol{\sigma})$: These are the spin-products (like $\sigma_I \sigma_J$ in Eq. 4) averaged over the entire lattice. They are different for each given configuration $\boldsymbol{\sigma}$. For example, by comparing to Eq. (4) and remembering that Eq. (5) is formulated per lattice site, we have for nearest-neighbor pairs:

$$\Pi_{2-1}(\boldsymbol{\sigma}) = \frac{1}{M} \cdot \sum_{I=1}^{M} \sum_{J \text{ NN to} I} \sigma_I \sigma_J \tag{6}$$

(we have not accounted for any double counting of lattice sites, incidentally; a factor 1/2 is therefore implicit in our definition of J_{2-1}).

The interesting thing about Eq. (5) is that it is *exact* in the sense that there are exactly as many possible configurations $\boldsymbol{\sigma}$ as there are possible figures f, on any given lattice. (In other words, there are as many basis functions in our expansion as there are possible configurations.) As long as we do not limit the sum over figures in any way, we have an exact expression.

In practice, we will benefit from the intuitive idea that interactions are negligible beyond a certain distance, and therefore the *relevant* figures in the sum must be limited. We should thus get a very accurate approximation to the true $E(\boldsymbol{\sigma})$ for *any* configuration $\boldsymbol{\sigma}$ *even when truncating the sum* to only a few relevant figures. Finding the relevant figures in a way that is robust is the aim of a good cluster expansion code.

A final note: While it is intuitive that a sum over figures should be restricted, this need not always be true. A large number of tiny long-range interactions can conceivably still sum up to large terms: for example, if the lattice as a whole contracts or expands differently for different configurations. Truncating infinite sums is something that should be done only after careful testing.

That said, the test case used for the present exercise – Ni-Al alloys – will turn out to be benign, at least in the range that is of interest here.

The present tutorial

In this tutorial, we will investigate how to parameterize, from first principles, a cluster expansion model for a given binary alloy.

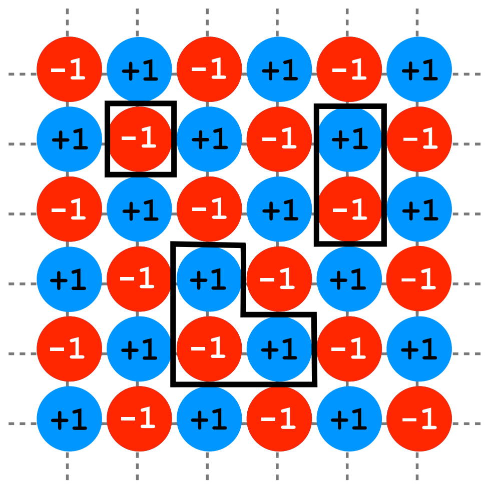
The alloy of choice is fcc Ni-Al, a classic system for which configurational energetics actually matters in practice. For very Ni-rich Ni-Al alloys, a mixture of the ordered Ni_3Al phase and a disordered Ni-rich solid solution phase occur on the same underlying lattice. The regions of the ordered Ni_3Al phase block lattice defects (primarily dislocations) from propagating and thus inhibit plastic deformation, making the alloy much stronger.

We will here investigate the basic ideas of a cluster expansion, using Ni-Al on a square lattice, a two-dimensional case (the actual ordering plane in Ni_3Al). In fact, all the methodology is the direct equivalent of a surface cluster expansion, for example an Al-rich layer (Al segregates to the surface) on top of a Ni-rich bulk alloy [1].

Contents

These are the problems we will practice for the next 3.5 hours in this session:

- Problem I: A 2D cluster expansion fit by hand *(20 min.)*
- Problem II: Using the cluster expansion code *(20 min.)*
- Problem III: Input energies: Ni-Al on a square lattice *(40 min.)*
- Problem IV: Minimal cluster expansion for 2d Ni-Al ... and some predictions *(80 min.)*
- Problem V: Order-disorder transitions *(60 min.)*
- Problem VI (bonus): The random alloy



$$(\bar{\Pi}_0, \bar{\Pi}_1, \bar{\Pi}_2, \bar{\Pi}_3) = (1, 0, -1, 0)$$

Figure 1: The $c(2 \times 2)$ structure and the Π's for the empty cluster, Π_0, (always 1), on-site cluster, Π_1, (sum over all sites), nearest-neighbor pair, Π_2, and smallest triplet cluster, Π_3.

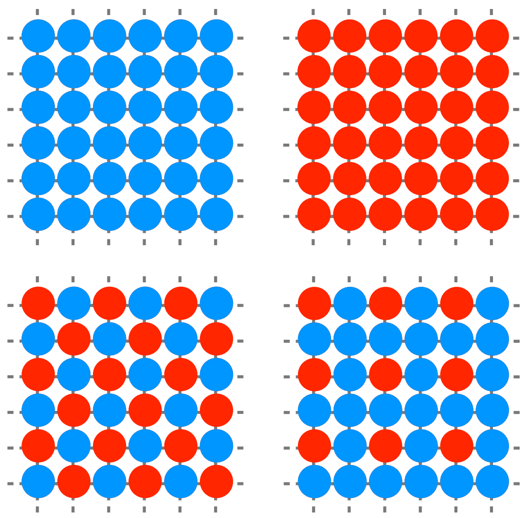
1 Problem I: 2D CE by hand

For this first exercise, we first consider only four structures (see Fig. 2). In this example, we use two hypothetical elements "red" and "blue" that form alloys on a two-dimensional square lattice. The energies for the structures we use here are *completely fictional*. In problem 3, you will use `FHI-aims` to calculate "real" energies for problem 4. The first two problems are designed just to help you get the hang of doing a cluster expansion.

1. pure blue, $E = -0.01$ eV/atom (upper left)
2. pure red, $E = -0.02$ eV/atom (upper right)
3. a so-called $c(2 \times 2)$ arrangement, $E = -.065$ eV/atom, (lower left).
4. a square supercell (2D analog of $L1_2$), $E = -.05$ eV/atom (lower right)

1.1 Average lattice occupations (the "Π's", also called "correlations")

Task: Following the same procedure that we did in the introduction to this tutorial, calculate the Π's for each of the three remaining structures in Fig. 2. The $c(2 \times 2)$ structure for which we calculated the Π's together (during in the introductory remarks) is shown in Fig. 1. The Π's we computed as a class are shown in the figure as the



$$\begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & -1 & 0 \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$

Figure 2: Four simple binary structures on a square lattice. The matrix on the right is the Π matrix with the 3rd row filled out with the answers we found in the introduction.

third row in a "Π matrix". Fill out the rest of the matrix. Π_0 is always 1. The other three columns will be for the on-site cluster, the pair cluster, and the triplet cluster, respectively.

After you have filled out the matrix, double check your results with the answer shown in the Appendix or ask one of the tutors—they have a copy of all the answers in their handout.

1.2 Finding the effective interactions (the "J's")

Conceptually, finding the J's in Eq. 5 is a simple linear algebra problem. For each configuration $\boldsymbol{\sigma}$ we have an equation with a unique value of E, unique values for the Π's, and unknown coefficients J. This system of linear equations form a simple matrix inversion problem. Given the energies for the four structures in the example, and having computed the Π matrix, we can find the J's by inversion:

$$\begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \end{pmatrix} = \begin{pmatrix} \Pi_{1,1} & \Pi_{1,2} & \Pi_{1,3} & \Pi_{1,4} \\ \Pi_{2,1} & \Pi_{2,2} & \Pi_{2,3} & \Pi_{2,4} \\ \Pi_{3,1} & \Pi_{3,2} & \Pi_{3,3} & \Pi_{3,4} \\ \Pi_{4,1} & \Pi_{4,2} & \Pi_{4,3} & \Pi_{4,4} \end{pmatrix} \begin{pmatrix} J_1 \\ J_2 \\ J_3 \\ J_4 \end{pmatrix} \tag{7}$$

$$\Downarrow$$

$$\begin{pmatrix} J_1 \\ J_2 \\ J_3 \\ J_4 \end{pmatrix} = \begin{pmatrix} \Pi_{1,1} & \Pi_{1,2} & \Pi_{1,3} & \Pi_{1,4} \\ \Pi_{2,1} & \Pi_{2,2} & \Pi_{2,3} & \Pi_{2,4} \\ \Pi_{3,1} & \Pi_{3,2} & \Pi_{3,3} & \Pi_{3,4} \\ \Pi_{4,1} & \Pi_{4,2} & \Pi_{4,3} & \Pi_{4,4} \end{pmatrix}^{-1} \begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \end{pmatrix} \tag{8}$$

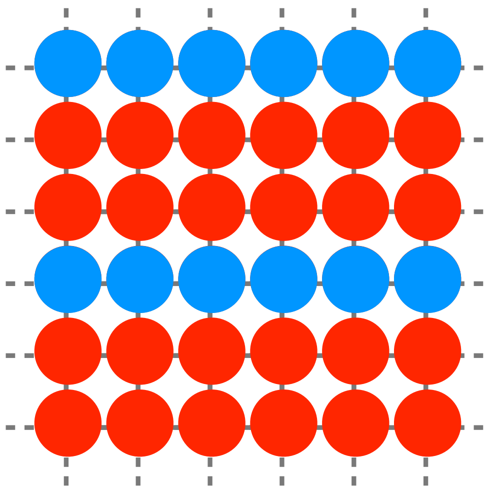


Figure 3: A new structure to use in predicting with your cluster expansion.

Task: Using the E's given above for each structure, invert the Π matrix that you found and use it to find the J's. You could invert the matrix by hand (but who would?!) or you can use a ready-made tool. For example, see `http://www.euclideanspace.com/maths/algebra/matrix/functions/inverse/fourD/index.htm`

1.3 Predictions and refining the fit

Task: Now that you have a set of J's, you can use them to predict the energy of a structure that wasn't used in the input set. Calculate the Π's (for the same clusters as before) for the structure shown in Fig. 3. Use your Π-vector for this structure with your J's and compute the energy of this structure. You may want to check your answer in the appendix (or with the tutors) before continuing.

2 Problem II: Using the cluster expansion code

2.1 Simple fits with UNCLE

We will now do exactly the same problem as in problem I, except that we will use the "universal cluster expansion code" (UNCLE) [2] to do all the bookkeeping, matrix inversion, and other computations. UNCLE is a general-purpose tool to perform cluster expansion fits, make predictions, and do many kinds of "physics" output tasks for configurational problems. Similar codes include the Automated Alloy Theoretic Toolkit, ATAT [3], and the CLUPAN code [4]).

To run a simple fit like the one we just did by hand (problem I), you will need 4 input files for UNCLE:

1. `lat.in` defines the underlying lattice of all the configurations (sometimes called the *parent lattice*)

2. `structures.in` lists the *input* structures (i.e., configurations) and the corresponding energies used in the fitting (in other words, the E vector and the structures that yield the Π matrix, like in problem I)

3. `CEfitting.in` parameters for the fitting

4. `clusters.out` contains the clusters (i.e., figures) to used in the expansion

The entries in the input files are relatively self-explanatory. Extensive comments have been added before each entry so that you do not have to use UNCLE as a black box if you don't want to. The input files are free format (`#` lines are comments) but the input is not keyword based like `aims`—the inputs have to be given in a set order.

Each of the 4 input files you need to run a simple fit can be found in the `problem_I_and_II` directory inside of the `prepared_input` directory. Each one has an extension `.set1`. Copy them to a working directory but *without* the `.set1` extension.

Task: We'll run UNCLE to perform the fit and not worry too much at the contents of the input files for now. Use the UNCLE executable called `uncle.x` that you will find in the `prepared_input` directory. We will run UNCLE using "mode 12" which directs UNCLE to perform a fit. If you copy the executable to your working directory, the command is:

```
./uncle.x 12
```

to do a simple fit. UNCLE will echo much of the information it reads in to the screen. There will be about 3 screenfuls of output. Ignore this for now, if you want. Look first at the file `PI_matrix.out`. You should see that this file contains the same matrix that you computed by hand in the first part of the activity (though the rows may be re-ordered as UNCLE sorts the structures according to concentration).

Task: Make your terminal window as wide as possible and look at fourth column of the file `fittingErrors.1.out` and you'll see that UNCLE found an exact fit (no

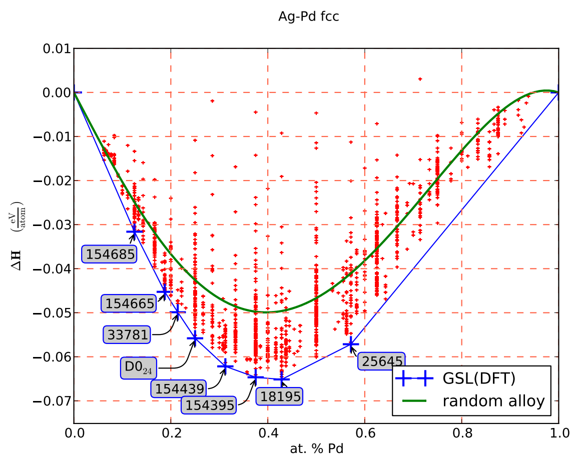


Figure 4: Convex hull for first principles enthalpies for Ag-Pt. The red +'s are the formation enthalpies for different configurations. The blue line is the *convex hull* for all the enthalpies. The configurations marked with a blue + are configurations that are thermodynamically stable (at $T = 0\,\mathrm{K}$). The green line is the energy of a completely random configuration, as predicted by the cluster expansion.

errors). This file lists input energies as well as formation enthalpies (called DHf_DFT and DHf_CE).

Definition: Often, the cluster expansion equations Eqs. (4) and (5) are not applied to straight total energies (which are large objects), but rather to formation enthalpies

$$\Delta H_f(\boldsymbol{\sigma}) = E_{\mathrm{tot}}(\boldsymbol{\sigma}) - x\,E_{\mathrm{tot}}(\text{pure A}) - (1-x)E_{\mathrm{tot}}(\text{pure B}) \quad . \quad (9)$$

$\mathrm{A}_x\mathrm{B}_{1-x}$ denotes the overall composition of a particular configuration $\boldsymbol{\sigma}$. Obviously, the formation enthalpy of the end points (pure A and pure B, respectively) is zero by definition.

The first two columns of the file are the concentrations of "red" and "blue". The third column contains the input energies. The fourth column contains the differences between the DFT input energies and the CE fitted values (should be zero here). The fifth column is CE energies. Ignore the 6th column (unnecessary detail for today). Columns 7 and 8 contain the DFT and CE *formation enthalpies*, respectively.

The file called `J.1.summary.out` lists the J values from the fit. The J values shown in this file will be the same as the ones you calculated by hand.

In a more general case, it would be nice to know what the chosen figures actually look like. To find out, open the longer file `J.1.out` . Look at the file and see whether you can figure out which interaction value pertains to which actual figure by drawing the vertices. In this 2D case, the clusters like in the y-z plane (so ignore their x-coordinates. If you are tempted to skip this part, be aware that we will do this again for slightly more complicated cluster expansions later.

Another interesting outcome from a cluster expansion are the formation enthalpies as a function of composition x ($0 \leq x \leq 1$). Figure 4 shows an example from a cluster

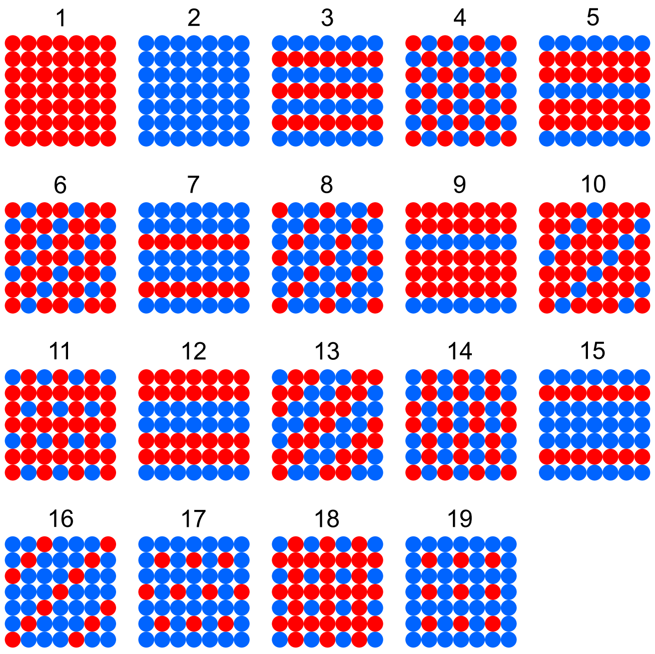


Figure 5: All possible binary configurations for unit cells of 4 atoms/cell or less.

expansion for Ag-Pt. If plots the enthalpies versus concentration, one can immediately sees whether the alloy is *ordering* (some $\Delta H_f < 0$) or *phase separating*, like "oil and water" (all $\Delta H_f > 0$). For ordering alloys, it is also possible to read the thermodynamically stable structures (at T=0) from such a plot of ΔH_f vs. x (like the blue crosses in Fig. 4). These structures are the ones between which one can draw straight line segments (in order of increasing x), such that no structure lies below the resulting line (the "convex hull") connecting x=0 and x=1 (the blue line in the figure).

You can plot the convex hull of the input structures using gnuplot, `gnuplot gsl_plot.gp`. (The script `gsl_plot.gp` is in the input directory, `prepared_input/problem_I_and_II`. This creates a file called `gsl.pdf`. You can use `evince` to view pdf files.) In this simple case, *all* of the structures are on the convex hull. You can also plot the fitted values and the input values (gnuplot script `errors_plot.gp`) but doing so isn't informative in this case because the errors are zero. (We'll do this in coming problems where it makes more sense.)

2.2 Ground State Searches

Task: Now that you have a list of J's, you can use them to make a prediction, just as before, for other structures. Consider all possible binary superstructures of a square lattice up to 4 atoms/cell. A picture of this is shown in Fig. 5. We can predict the energy for each one of these structures using UNCLE(thankfully, UNCLE will compute all the Π's). Copy the `groundstatesearch.in.set1` file from the input directory (remove the `.set1` extension as before). The `groundstatesearch.in` specifies the set of unit cells for which the predictions are made, sizes 1–4.

Type: `./uncle.x 21`

to run a ground state search for all cell sizes from 1 to 4 ("mode 12"). UNCLE generates a list of structures (`struct_enum.out`) and then computes the energy and formation enthalpy for each one, listed in the file `gss.out`. The structure you predicted in the first part of the activity (shown in Fig. 3) is number 5 in the enumerated list (top row, last entry in Fig. 5). Look in `gss.out` to find the energy of structure number 5 and compare it to what you got by hand.

Plot the complete results of the ground state search using the `gss_plot.gp` script in the input directory. The plot file is called `gss.pdf`. You will find that there are some structures that are predicted to have formation enthalpies lower than our input structures. If this weren't a made up example (fictional energies), we would probably want to calculate some more DFT energies for these low-lying structures and add them to the input set and make a new fit.

3 Problem III: Input energies: Ni-Al on a square lattice

We will now substitute the made-up example of the previous section with an actual, DFT-computed computed system. We begin by treating Ni-Al on a square lattice, as a free-standing thin film. While this example is still somewhat artificial (hard to build an airplane out of free-standing Ni-Al thin film superalloys), it suffices to demonstrate many simple principles that will benefit us in three dimensions. This example is, however, a little more useful than that. In Ni-Al alloys, Al tends to *segregate* to the outermost plane of the crystal. A *surface cluster expansion* of an Al-rich surface plane on a Ni-rich alloy underneath would follow the exact same formalism, and in fact, such behavior has been observed [1].

The ***default*** calculational settings for FHI-aims

- Parameters for FHI-aims `control.in`:

By default, for things like grid spacings, we shall use *light* settings, but *we will still test the convergence with respect to the number of basis functions.* In a real cluster expansion, we need converged enthalpies to get accurate interaction energies (J's). Testing the basis set is for the DFT calculations is standard practice, as is a test of all other numerical parameters.

For a header of control.in, consider these settings:

```
    xc                pw-lda
    charge            0.
    relativistic      atomic_zora scalar
    occupation_type   gaussian 0.1
#
    mixer             pulay
      n_max_pulay              10
      charge_mix_param         0.2
    sc_accuracy_rho   1E-4
    sc_accuracy_eev   1E-2
    sc_accuracy_etot  1E-6
    sc_iter_limit     100
```

We use LDA (`xc pw-lda`), and "atomic ZORA" type scalar relativity. Since the structures in question are metallic, we use a Gaussian broadening of 0.1 eV for all calculations by default.

In the output file, we will be mostly interested in per-atom energies for this exercise (which will then be converted into per-atom formation enthalpies). In addition, since this is a metallic system, we will use the "$T \rightarrow 0$" (i.e. Gaussian smearing width towards zero) extrapolated values of the total energy in the FHI-aims output file:

```
  | Total energy (T->0) per atom              :      -15315.50440449 eV
```

Again, we only use the extrapolation for a real metallic system, for which it is intended, not for example in the case of fractionally occupied atomic or molecular energy levels.

An initial(!) input file `control.in.begin` is also included in the directory `prepared_input/problem_III` .

Parameters for FHI-aims geometry.in:

We begin with a simple series of calculations on a fixed lattice that neglects all lattice relaxations. The experimental lattice parameters of fcc NiAl alloys lead to the following nearest-neighbor distances (see appendix):

- fcc Ni: 2.492 Å
- $L1_2$ Ni_3Al: 2.520 Å
- B2 NiAl: 2.509 Å
- fcc Al: 2.863 Å

Thus, the nearest-neighbor distance in the range of interest here (the Ni-rich range) almost does not vary at all. Only in the Al rich range do we observe a significant change. We shall therefore choose a default lattice parameter of 2.50 Å for any unrelaxed calculations in this exercise.

To separate the individual Ni-Al-planes, we use a vacuum thickness of 40 Å for all two-dimensional calculations by default. For FHI-aims and the example of a pure Ni plane, this means:

```
# fcc Ni, lattice parameter 2.50 AA
#
  lattice_vector   2.50   0.00   0.00
  lattice_vector   0.00   2.50   0.00
  lattice_vector   0.00   0.00  40.00
#
  atom   0.0   0.0   0.0   Ni
#
```

You should be able to set up all other needed structures in a similar way by extending the given (2D) unit cell and adding the necessary atoms.

Making sure that the DFT values are converged.

We use formation enthalpies [see Eq. (9)] for our cluster expansion, and also to test the convergence of our DFT settings. Again, for a real cluster expansion it is essential to verify the convergence of all input energy differences to a few meV or better, since larger uncertainties can easily alter the physical behavior of the resulting cluster expansion for larger systems.

Each formation enthalpy must be calculated as total energy differences from three separate first-principles calculations (pure Ni, pure Al, and the mixed structure we are looking for).

Task: Set up `control.in` and `geometry.in` files for FHI-aims for the pure Ni, pure Al, and c(2×2) structures found in Fig. 2. (Blue corresponds to Ni and red to Al.)

Basis set choice

Task: Begin by testing the influence of the basis set for fixed, safely converged k-space grid 24×24×1 for each of the investigated structures.

```
  k_grid    24  24  1
```

Normally (especially for a three-dimensional structure), you would want to verify the convergence of your k-space grids explicitly, especially for metals, starting from somewhat lighter settings. In the interest of (human) time, we here prescribe a dense k-grid for the smallest unit cell structures. Be sure to reduce the density of this grid for larger unit cells. We will come back to the k-grid below.

Add the *light* species defaults for Al and Ni to `control.in`.

Test the following basis sets (by uncommenting the respective basis functions):

- 1: *light* default settings
- 2: Al: *tier* 1 + *gd*, Ni: *tier* 1 + *dp* (i.e., *uncomment* the next higher radial functions in the species defaults for either element):

```
#
  species        Al

  [...]

#  "First tier" - improvements: -199.47 meV to -10.63 meV
     ionic 3 d auto
     ionic 3 p auto
     hydro 4 f 4.7
     ionic 3 s auto
#  "Second tier" - improvements: -5.35 meV to -1.57 meV
     hydro 5 g 7
     hydro 3 d 6
#     hydro 2 s 11.6
#     hydro 2 p 0.9

  [...]

  species        Ni

  [...]

#  "First tier" - improvements: -123.08 meV to -11.61 meV
     hydro 3 p 6
     hydro 4 f 9
     hydro 5 g 12.4
     hydro 3 d 5.2
     ionic 4 s auto
#  "Second tier" - improvements: -6.71 meV to -1.07 meV
     ionic 4 p auto
     hydro 4 d 6
#     hydro 6 h 18
#     hydro 4 f 9.4
#     hydro 4 f 16.4
#     hydro 1 s 0.75

  [...]
```

- 3: The full *tier* 2

Which basis set do you find to be converged to a few meV?

k-space grid

If you are short on time, you may want to skip this task, we are already using a safely converged 24×24 k-space grid above. However, make sure to read through the notes anyway. In a real calculation involving metals, you would always want to make sure that your k-space grid is converged enough for the task you are addressing.

Task: For the accurate basis set specified above (in fact the default basis set for *tight* settings in FHI-aims), try out the following additional k-space grids:

- 12×12×1
- 16×16×1

What do you find? For which k-space grid is ΔH_f converged to a few meV?

Of course, the k-space grids above are given in relative units of the reciprocal lattice vectors—but as the real-space unit cell grows to include more atoms, the reciprocal lattice vectors shrink accordingly. For practical calculations (Brillouin zone integrals), it's the *density* of the k-space grid that counts. So, for the 2-atom $c(2 \times 2)$ unit cell, we would get away with a lighter k-space grid than for the single-atom unit cells. For example, if 24×24 was appropriate for the primitive cell, 16×16 might have been appropriate for the larger 2-atom cell—and computationally much cheaper. We will take such considerations into account below.

The other structure

We have now identified suitably converged k-grid and basis settings for the remainder of this tutorial.

Task: Using these definitive settings, compute the energy of the final missing input structure in Fig. 2 (use the composition Ni_3Al)

You will see that both unit cell vectors are simply doubled in this case, compared to the primitive unit cell. What does this mean for the k-space density? Which `k_grid` settings would you use?

Structure	Total energy [eV/atom]	ΔH_f [eV/atom]
Ni		
Al		
c-(2×2)-NiAl		
p-(2×2)-Ni_3Al		

4 Problem IV: Minimal cluster expansion for 2d Ni-Al ... and some predictions

Based on the previous task, we now have the formation enthalpies of four structures (Fig. 2) for the 2D Ni-Al system. We can use the results to create the same simple cluster expansion as in Problem II and explore the results.

4.1 Cluster expansion

Task: Repeat the cluster expansion (UNCLE's mode 12) using only the four structures of Fig. 2. You can merely alter the input file (`structures.in`) that we used before. Just replace the energies from the made up case of Problem II with the new enthalpies you found in Problem III. (What was blue should become Ni, red should become Al.) Remember, the J's are in the `J.1.summary.out` file and the fitting errors are listed in `fittingErrors.1.out` (but, as before, there are no errors because we have 4 input structures and 4 clusters so there is an exactly-invertible solution).

What interaction values do you get for each interaction type?

4.2 First predictions

Task: Use the UNCLE code to predict the formation enthalpies of some additional structures. To do this, just run UNCLE again in mode 21 (`./uncle.x 21`) to do a ground state search. (The file `groundstatesearch.in` controls how many structures are included in the search.) If you are running your calculation in the same directory as before, you will see that UNCLE won't overwrite an old `gss.out` file; note how it complained. Just delete the file and run again.

Plot the "ground state line": Plot the results (`gnuplot gss_plot.gp`). (Rename the file `gss.out` so that you can refer to it later.) Are there "new" ground state structure candidates (outside the four that we know in DFT)?

4.3 Extending the expansion

Let us add four additional DFT input structures to the expansion, both to verify the accuracy of the results so far, and to extend the input database. We will focus on the three-atom structures in Fig. 5, numbers 5, 6, 7, 8 in the `gss.out` list.

Where are these structures in the ground state search plot?

Task: Compute the DFT-LDA enthalpies of formation of these four structures, which are called (in surface science notation):

- $p(3{\times}1)$, Ni_2Al and $NiAl_2$, nos. 5 and 7
- $c(3{\times}3)$diag, Ni_2Al and $NiAl_2$, nos. 6 and 8

Use FHI-aims and the converged settings obtained in the previous exercise. Think about dense enough, but not too dense k-space grids for either unit cell.

If you are short on time at this stage: The results of this task and further, precomputed formation enthalpies can also be found in two tables at the end of this exercise. You may want to pick out the relevant values from there. Even if so, be sure to read through the text below and include the resulting unrelaxed and relaxed formation enthalpies in your previous results.

Note that these structures both have atomic positions that could relax inside the unit cell, i.e., they are not constrained by symmetry.

- First, obtain total energies and formation enthalpies for the unrelaxed structures. How close are the values to the predictions?
- Second, obtain total energies after relaxing only atomic positions inside the unit cell (but not the unit cell shape itself). To do so, edit the control.in file in the following way:

  ```
  relax_geometry bfgs 1.e-2
  sc_accuracy_forces  5E-4
  ```

  (Choose the smallest possible unit cell, so that the calculation does not take too much time. Scale your converged k-space grid accordingly, if you have used a different unit cell before.)

What are the results? Is relaxation important? Could the high-symmetry structures of Fig. 2 have been used to predict relaxed or unrelaxed formation enthalpies accurately?

New ground state predictions?

We now have DFT-LDA computed input energies for eight input structures. We will focus on the *relaxed* structures from here on.

Task: Update the UNCLE input files to use these 8 input structures. First, add the 4 additional structures to the `structures.in` file. (The format for the `structures.in` file should be obvious, but there is a template in the input directory called `structures.in_8` if you need it.) Because we have more input data, we can use more fitting parameters (i.e., more clusters) in the expansion.

Which extra figures should we use? In general, answering this question is difficult. The reason for the difficulty is that there is no natural hierarchy to the clusters themselves—Is a *third* nearest-neighbor *pair* more or less important than a *first* nearest-neighbor *triplet*? Is a compact, 5-vertex cluster less important than a long range pair-cluster? It is not easy to say.

One way to truncate the cluster expansion (in other words, to pick which clusters to use) is to use an evolutionary approach: pick the clusters at random for many candidate expansions, evaluate the predictive capability of each expansion, then refine the good expansions by "mating" and "mutation," and continue until a suitably robust expansion is found. In short, use a genetic algorithm to pick the clusters [5].

Task: To use the genetic algorithm of UNCLE, we first need to make a longer list of clusters in the `clusters.out` file. To do this, edit the `lat.in` file. Near the bottom of the file, in the "Cutoffs" section, there are five numbers on a single row. These are maximum distances[1] for pair clusters, triplet clusters, and so on up to clusters with six vertices. Change the cutoffs so that the maximum length of a pair cluster is 3.0, for a triplet 1.6, and 1.4 for the quadruplet. Leave the other two at zero. Generate the new `clusters.out` file by running UNCLE in mode 10, `./uncle.x 10`. UNCLE will tell you that the new cluster list includes one on-site cluster, six pair clusters, five triplets, and one quadruplet, 13 clusters in all (14 including the constant term, the so called "empty cluster").

Task: Finally, edit the `CEfitting.in` file and change the first un-commented line from `simple` to `optimize` ("optimize" means use the GA rather than a simple least-squares-type fit.) Also, change the number of clusters included in the fit from 4 to 5 (first uncommented line in the "Clusters section". There are $\binom{14}{5} = 2002$ different possible cluster expansions if we choose 5 clusters from a pool of 14. We'll let the GA algorithm search for a near-optimal one. (It is often the case that the cluster pool is hundreds, rather than the 14 here, and the number of clusters included is a few dozen, so the search space can easily be bigger than Avogadro's number. Because we can't directly test each candidate in those cases, the genetic algorithm becomes essential.)

> There are *lots* of parameters you could play with to do a GA-based cluster expansion (take a quick look through the `CEfitting.in` file.) We won't discuss them at all during the exercises today, but if you are interested ask Gus Hart, Lance Nelson, or Volker Blum.

Task: Run the fit again, `./uncle.x 12`, with the larger number of input structures, the larger number of clusters, and the GA option for cluster selection. Plot the errors using the gnuplot script `errors_plot.gp` (creates `errors.pdf`) from the `prepared_input/problem_IV` directory. The fit is pretty close.

Which interactions were chosen? Which ones dominate?

Use the new J's that you found to do a ground state search (GSS). Increase the maximum size of structures included in the GSS from 4 to 8 (edit the `groundstatesearch.in` file). Plot the output of the GSS (run `./uncle.x 21` and use the gnuplot script `gss_plot.gp`). Does the CE predict any new ground states? (It doesn't require much more cpu time to go a GSS for structures up to cell sizes of 12, 16, or 20. Try that too if you want.)

4.4 Compare your results to DFT-LDA

You will see in your GSS plot that there appears to be a ground state predicted at 75% Al. The following somewhat arcane unixism will list all the structures at that

[1]They aren't actual distances but an average distance from the "center of gravity" of the cluster to all its vertices.

concentration in the order of their formation enthalpies.
`grep "0.75000  0.25000" gss.out | sort -k 8`.
Note that structure numbers 11 and 18 are the lowest and are also degenerate with one another. Look at Fig. 5 to see what structures these are.

We could next compute the enthalpies of enthalpies structures 11 and 18, as well as several others, by direct DFT. If you have time, you are welcome to do so—or, you may even want to try out predicted structures for larger unit cells.

In the interest of time for the present tutorial, we have precomputed the formation enthalpies for the 19 structures in Fig. 5, both in their unrelaxed and their physically more reasonable, RELAXED variants (see the Tables below). Are structures 11 and 18 degenerate according to DFT?

The cluster expansion incorrectly predicts that the two structures are degenerate whereas DFT does not. This does not mean that the CE is flawed *per se* but only that the CE did not have enough “training data” (input structures) to distinguish between structures 11 and 18. We can further refine the cluster expansion by using all 19 structures as input and make an expansion with additional clusters.

Task: There is a file `structures.in_19` in the `prepared_input/problem_IV` directory that contains all 19 of the structures from Fig. 5 and the relaxed enthalpies from the table below. Increase the size of your cluster pool (edit `lat.in`, use for example `3.2 1.8 1.6 0 0` in the "Cutoffs" section and use mode 10). Then, re-run the GA allowing it to use, say, 10 clusters from the pool and all 19 structures as input. (The number of clusters used is specified in `CEfitting.in`, first number in the “Clusters section”.)

Once again, look at the predicted interactions. Which are the dominant ones?

Plot the errors (`gnuplot error_plot.gp`) and perform another GSS (mode 21). How does UNCLE do now at reproducing the energies of the 19 inputs? Are there newly predicted ground states?

Formation enthalpies of 19 gss structures from Fig. 5, UNRELAXED:

Number	Structure	Energy [eV]	ΔH_{DFT} [eV/atom]
1	Ni	-41495.7023	0.0
2	Al	-6588.3014	0.0
3	p-(2×1)-NiAl	-24042.4372	-0.4354
4	c-(2×2)-NiAl	-24042.8048	-0.8030
5	p-(3×1)-$NiAl_2$	-18224.3771	-0.2754
6	c-(3×3)diag-$NiAl_2$	-18224.6163	-0.5145
7	p-(3×1)-Ni_2Al	-29860.1905	-0.2885
8	c-(3×3)diag-Ni_2Al	-29860.4176	-0.5156
9	p-(4×1)-$NiAl_3$	-15315.3506	-0.1990
10	c-(4×4)diag-$NiAl_3$	-15315.5193	-0.3677
11	c-(4×2)-$NiAl_3$	-15315.5088	-0.3572
12	p-(4×1)-Ni_2Al_2	-24042.2124	-0.2106
13	c-(4×4)diag-Ni_2Al_2	-24042.4225	-0.4208
14	c-(4×2)-Ni_2Al_2	-24042.6248	-0.6230
15	p-(4×1)-Ni_3Al	-32769.0547	-0.2025
16	c-(4×4)diag-Ni_3Al	-32769.2168	-0.3647
17	c-(4×2)-Ni_3Al	-32769.2236	-0.3715
18	p-(2×2)-$NiAl_3$	-15315.5044	-0.3528
19	p-(2×2)-Ni_3Al	-32769.2366	-0.3845

Formation enthalpies of 19 gss structures from Fig. 5, RELAXED:
Internal lattice relaxation of some structures at a=2.50 Å, fixed.

Number	Structure	Energy [eV]	ΔH_{DFT} [eV/atom]	Relaxation energy [eV/atom]
3	p-(2×1)-NiAl	-24042.4372	-0.4354	0.0
4	c-(2×2)-NiAl	-24042.8048	-0.8030	0.0
5	p-(3×1)-$NiAl_2$	-18224.4614	-0.3597	-0.0843
6	c-(3×3)diag-$NiAl_2$	-18224.7630	-0.6613	-0.1468
7	p-(3×1)-Ni_2Al	-29860.2041	-0.3021	-0.0136
8	c-(3×3)diag-Ni_2Al	-29860.4178	-0.5158	-0.0002
9	p-(4×1)-$NiAl_3$	-15315.4550	-0.3034	-0.1044
10	c-(4×4)diag-$NiAl_3$	-15315.6748	-0.5232	-0.1555
11	c-(4×2)-$NiAl_3$	-15315.6215	-0.4699	-0.1127
12	p-(4×1)-Ni_2Al_2	-24042.3083	-0.3065	-0.0959
13	c-(4×4)diag-Ni_2Al_2	-24042.6735	-0.6717	-0.2510
14	c-(4×2)-Ni_2Al_2	-24042.7043	-0.7025	-0.0795
15	p-(4×1)-Ni_3Al	-32769.0547	-0.2025	0.0
16	c-(4×4)diag-Ni_3Al	-32769.2300	-0.3779	-0.0132
17	c-(4×2)-Ni_3Al	-32769.2437	-0.3916	-0.0201
18	p-(2×2)-$NiAl_3$	-15315.5044	-0.3528	0.0
19	p-(2×2)-Ni_3Al	-32769.2366	-0.3845	0.0

5 Problem V: Order-disorder transitions

With the last exercise completed, we now have a total of 19 DFT-LDA formation enthalpies for all structures up to four atoms per unit cell. In a real cluster expansion, we would now attempt to extend the expansion towards larger-scale/less trivial input structures. In the interest of time, we will assume that the cluster expansion already has sufficient predictive power, and we will use it to explore one of the most important phenomena in alloy theory: Order-disorder transitions.

> **Variant:** A single plane of Ni-Al does make for an interesting cluster expansion, but you might want to cluster expand something "real" instead. If you wish, we have also pre-tabulated "formation enthalpies" (per surface atom) for the same square-lattice Ni-Al structures, but supported on a pure fcc Ni slab (five pure Ni planes in addition to the topmost plane)—see Fig. 7 and the table at the end of this problem. You could use these instead. In the limit of very Ni-rich Ni-Al alloys, an Al-enriched top layer would actually be the physically expected situation for the "clean" surface [1].

We will use canonical Monte Carlo simulations and two different expansions: A naive, short-ranged one, and one that represents "the best we can do" right now (not necessarily the best we could do given much more time).

5.1 Nearest-Neighbor-only CE

Task: Create a nearest-neighbor pair only cluster expansion, using the 19 *relaxed* structures from the previous exercise as DFT-LDA input. Let's also take the `clusters.out`, `lat.in` and `CEfitting.in` file from the previous exercise to start with. You can create a cluster expansion that is restricted to nearest-neighbor (NN) interactions only by editing the `clusters.out file`. Delete (or comment out) the first cluster in the file (the on-site term [it has only one vertex]), so that the first cluster listed in the file is the NN pair cluster. In the `CEfitting.in` file, change the fitting scheme back to `simple`. You also need to change the number of clusters used in the fit. The first un-commented line in the "Clusters section" is the number of clusters that is used in the fitting (you've changed this already several times). Change this from 10 to 2 (*not* to 1—UNCLE always includes the constant term as well [UNCLE will set it to zero in this example though]).

Task: Run UNCLE with mode 12 again to create a fit. The errors will be large. How large? (`gnuplot errors_plot.gp` and look in the `fittingErrors.1.out` file.)

Now, do a Monte Carlo (MC) simulation by using mode 30 of UNCLE. You can find the input file `MCpar.in` in the `prepared_input/problem_V` directory. The MC run takes several minutes. Plot your results using the gnuplot script `Tc_plot.gp`.

5.2 "Best we can do" CE

Task: Starting from the same set of input structures as before, this time create a cluster expansion using the GA, as you did in problem IV (use the `optimize` keyword).

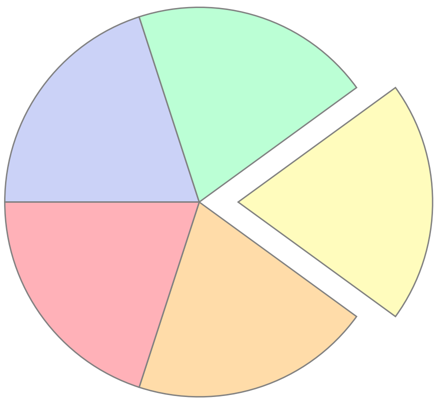


Figure 6: The goodness (i.e., the prediction capability) of a cluster expansion is measured by a *cross validation score*. The input data is divided into groups (5 groups in the cartoon to the left). A fit is made leaving out each group in turn. Each fit is tested to assess how well it predicts the group of input data that was "left out". In this way, the cluster expansion is tested for its ability to *predict*. (The predicted data is actually known but not included to make the fit.)

This time we'll use all the clusters in the `clusters.out` file. In the `CEfitting.in` file, you'll probably want to use around 10 clusters in the fit ("clusters" section), 5 populations in the "populations" section, and a *k*-fold cross-validation setting ("Cross validation" section) of `5 8`, see Fig. 6). Because this case still runs relatively fast, you may want to change the number of generations to 500 as well.

Task: What does the expansion look like? By how much does our fitting error change? (Make a plot.) Predict ground states up to 12 or 16 atoms/cell (modify `groundstatestructure.in`) and see if things look any different than when you only went up to 8. (Rename the generated `gss.out` and `struct_enum.out` files.) We could reduce the fitting errors even more by adding more input data and increasing the maximum number of clusters used during the GA runs.

Task: With your new expansion, try the Monte Carlo again and see if the ordering transition happens at a different temperature. Because you are using more than just a single NN interaction it will run considerably slower.

Task: The Monte Carlo run creates a whole bunch of files of the form `MCcell###.out`, one for each temperature step. Each file contains the data for the final configuration at each temperature step. You can use these files to visualize the simulation cells. UNCLE mode 31 will reformat the data files for plotting (you'll need the `MCevaluation.in` file from `prepared_input/problem_V`). Then run `gnuplot MCcell_plot.gp` and look at the `MCcell.pdf` file. Edit the `MCevaluation.in` file and repeat the procedure to make a plot for configurations above, below, and right at the transition temperature. (Have a look into `MCsimanneal.out` to see which `MCcell###.out` corresponds to what temperature.) You could increase the cell size to 40×40 in `MCpar.in` for more interesting plots. For plotting the bigger cell set the pointsize in `MCcell_plot.gp` to 1.3.

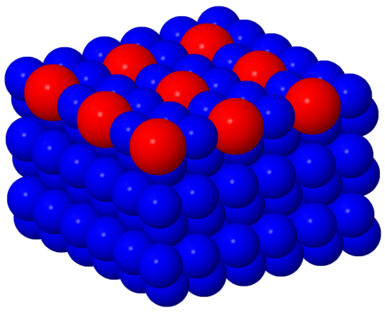


Figure 7: Slab model: The p-(2×2)-Ni_3Al structure in the first layer of a six layer fcc Ni slab

5.3 Order-disorder transition at 80% Ni

Repeat the previous exercise (run MC), but this time in the Ni-rich range, say 80 % Ni or so, at a concentration where we do not know of any ordered ground state structure yet (based on the DFT input data). Your plots will be more interesting if you increase the cell size to 40 × 40 and you'll want to change the temperature parameters. You can set the number of temperature schedules to 2, starting from 10000 K by steps of 400 K down to 2000 K and continuing from there by steps of 40 K down to 120 K. Edit the `MCpar.in` file to make these changes. (Note the space between "-" and the stepsize, e.g. "- 40" in the input format.)

What does the short range order look like after a MC cooling schedule (use mode 31 and edit `MCevaluation.in` as in the previous exercise)? You might want to look below, above, and at the transition temperature again.[2] Are there any order-disorder transitions along the way (use `Tc_plot.gp`)?

Based on the Monte Carlo results and also based on the CE ground state line, which structure would you verify next in a direct density functional theory calculation? If you choose to do a direct DFT calculation including relaxation (not so hard), how well does the result agree with your cluster expansion prediction?

The following table lists the formation enthalpies of our 19 structures calculated as monolayers (as before) and the same structures calculated within a more realistic slab model. If you have time you can repeat the previous exercise using the slab energies. Are the results different from what you got using the monolayer model?

Formation enthalpies of 19 gss structures from Fig. 5, relaxed:

[2]Hint: If for some reason the answer is not obvious, T=200 K, 1000 K, 5000 K worked for us.

Number	Structure	Energy [eV]	ΔH_{DFT} [eV/atom] (monolayer)	ΔH_{DFT} [eV/atom] (slab)
3	p-(2×1)-NiAl	−24042.4372	−0.4354	−0.5194
4	c-(2×2)-NiAl	−24042.8048	−0.8030	−0.7984
5	p-(3×1)-$NiAl_2$	−18224.4614	−0.3597	−0.4596
6	c-(3×3)diag-$NiAl_2$	−18224.7630	−0.6613	−0.6132
7	p-(3×1)-Ni_2Al	−29860.2041	−0.3021	−0.3814
8	c-(3×3)diag-Ni_2Al	−29860.4178	−0.5158	−0.4955
9	p-(4×1)-$NiAl_3$	−15315.4550	−0.3034	−0.3690
10	c-(4×4)diag-$NiAl_3$	−15315.6748	−0.5232	−0.5527
11	c-(4×2)-$NiAl_3$	−15315.6215	−0.4699	−0.4711
12	p-(4×1)-Ni_2Al_2	−24042.3083	−0.3065	−0.3685
13	c-(4×4)diag-Ni_2Al_2	−24042.6735	−0.6717	−0.5523
14	c-(4×2)-Ni_2Al_2	−24042.7043	−0.7025	−0.5995
15	p-(4×1)-Ni_3Al	−32769.0547	−0.2025	−0.3010
16	c-(4×4)diag-Ni_3Al	−32769.2300	−0.3779	−0.4293
17	c-(4×2)-Ni_3Al	−32769.2437	−0.3916	−0.3910
18	p-(2×2)-$NiAl_3$	−15315.5044	−0.3528	−0.4746
19	p-(2×2)-Ni_3Al	−32769.2366	−0.3845	−0.4482

6 Problem VI: The “random alloy”

The last exercise is, at this point, a “bonus” exercise. We have seen how to create a cluster expansion that gives us access to essentially the entire configurational energetics on a lattice by a simple sum over interactions. We have also seen how to generate various ordered or random alloy states at different temperatures using Monte Carlo simulations.

There is, in fact, a considerable amount of interest in the “random alloy limit” in parts of the literature: What is the energy of an ideal random alloy (no short-range ordering at finite temperatures)? What is the electronic structure of such a completely random alloy?

One way that is popular in the literature to address this question is the so-called “coherent potential approximation”, which creates effective scatterers out of the elements on the actual lattice. Unfortunately, with effective scatterers on a lattice, it is difficult in practice to take local lattice relaxations into account.

Can we capture the random alloy limit (energetics, electronic properties, any other property) in a different way, without having to roll the dice on a huge Monte Carlo cell at infinite temperature, and then run direct density functional theory on that huge structure? It turns out that there is a way.

Consider the random alloy limit for the Π_f of any figure f. For the random alloy, all site occupations are uncorrelated, and therefore

$$\Pi_{\text{random}} = (1 - 2x)^{N_f} \tag{10}$$

where N_f is the number of vertices in the figure.

Based on this knowledge, we can search specifically for small unit cell structures, where the Π_f's for the first few short-range interactions come as close as possible to the ideal random-alloy value. Such structures are known as *special quasirandom structures* — but because they are based on actual site occupations, local relaxations can be taken into account as usual.

Task: As a bonus exercise for the present tutorial, we provide 3 special quasirandom structures of size 8 for x=0.5 and 0.625 (`prepared_input/problem_VI`). Compute the unrelaxed and relaxed total energies for any of these structures.

How big is the influence of relaxation?

Based on the cluster expansion of the previous exercise, do the DFT formation enthalpies match what you would have expected for the random alloy limit?

Appendix

Crystallographic data

Experimental lattice parameters and nearest neighbor distances in simple structures:

- Nickel: fcc structure, a=3.524 Å. (Probably room temperature. Source: http://www.webelements.com.)
  NN distance: 2.492 Å
- Aluminium: fcc structure, a=4.0495 Å. (Probably room temperature. Source: http://www.webelements.com .)
  NN distance: 2.863 Å
- Ni_3Al: $L1_2$ structure (fcc based), a=3.5642 Å. (Room temperature. Source: Wang, Liu, Chen 2004, cited there.)
  NN distance: 2.520 Å
- NiAl: B_2 structure (bcc based), a=2.897 Å. (Room temperature. Source: Wang, Liu, Chen 2004, cited there.)
  NN distance: 2.509 Å

References

[1] R. Drautz, H. Reichert, M. Fähnle, H. Dosch, and J. M. Sanchez, Phys. Rev. Lett. **87**, 236102 (2001).

[2] D. Lerch, O. Wieckhorst, G.L.W. Hart, R.W. Forcade, and S. Müller, Modell. Simul. Mat. Sci. Eng. **17**, 055003 (2009).

[3] A. van de Walle, M. Asta, G. Ceder, Calphad **26**, 539-553 (2002); http://www.its.caltech.edu/ãvdw/atat/ .

[4] A Seko, Y Koyama and I Tanaka, Phys. Rev. B **80**, 165122 (2009); http://clupan.sourceforge.net/index-e.html .

[5] G.L.W. Hart, V. Blum, M.J. Walorski, and A. Zunger, Nature Materials **4**, 391 (2005).