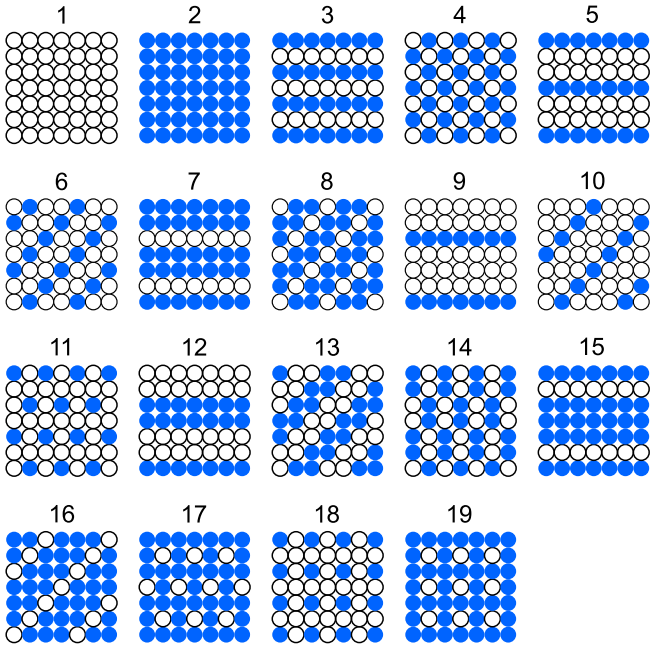
Hands-On Tutorial on Cluster Expansion

IPAM
Los Angeles, California, July 2014

Modeling of configurational energetics
Manuscript for Exercise Problems



Adapted from a tutorial originally prepared by
Volker Blum, Gus Hart, Norina Richter July 18, 2011
Gus L. W. Hart gus.hart@byu.edu
Conrad W. Rosenbrock rosenbrockc@gmail.com
Björn Bieniek bieniek@fhi-berlin.mpg.de
Volker Blum blum@fhi-berlin.mpg.de

Background

This section gives the physical and mathematical background again (with equations) that you heard at the start of the tutorial. 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 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. 5. 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 sits on each site—we don't know the configuration. E becomes a simple function of the occupation of these sites by the different elements (the *configuration*),

$$E_{\mathrm{BO}}^{\mathrm{gs}}(\boldsymbol{R}_1, \ldots, \boldsymbol{R}_M) \rightarrow E_{\mathrm{conf}}(Z_1, \ldots, Z_M) \quad . \tag{2}$$

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. 4 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 of a given configuration (or the average of thermal positions), then 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:

$$E(\sigma_1, \ldots, \sigma_M) = E^{\mathrm{Ising}}(\sigma_1, \ldots, \sigma_M) = J_0 + J_1 \sum_I \sigma_I + J_{2,1} \sum_{I=1}^{M} \sum^{D_{2,1}} \sigma_{I_1} \sigma_{I_2} \quad . \tag{4}$$

where the double-sum for $D_{2,1}$ is over the degenerate states of σ_I and will be explained in more detail later.

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 4), 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 system, we could 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 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 says 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 4.
- 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 $\boldsymbol{\sigma}$. They can be calculated in the following manner:

$$\bar{\Pi}_{\text{cluster}}(\text{configuration}) = \frac{1}{\substack{\text{number of}\\\text{vertices}}} \cdot \frac{1}{\substack{\text{size of}\\\text{unit cell}}} \cdot \frac{1}{\text{degeneracy}} \sum_{\substack{\text{unit}\\\text{cell}}} \sum_{\substack{\text{unique}\\\text{cluster}\\\text{orientations}}} \left(\substack{\text{product of spins}\\\text{in the cluster}}\right) \tag{6}$$

Examples of how the products are found using various clusters (on a 2 dimensional lattice), for different $D_{k.n}$ can be found in Figures 1 and 2. If we write the above relations more rigorously it takes the more common form:

$$\bar{\Pi}_{k,n}(\boldsymbol{\sigma}) = \frac{1}{k} \cdot \frac{1}{M} \cdot \frac{1}{D_{k,n}} \sum_{I=1}^{M} \sum^{D_{k,n}} \sigma_{I_1}\sigma_{I_2}\cdots\sigma_{I_k} \tag{7}$$

Remember that M is the number of atomic sites in the unit cell. The k factor here is the number of vertices in the cluster and the n is the type of the lattice sites covered by the cluster. (Previously we had not accounted for any double counting of lattice sites, incidentally; a factor 1/k is therefore included in our new definition of $J_{k,n}$). Note that we have introduced another normalizing factor $D_{k,n}$ that is the "geometric degeneracy" factor of the cluster. This number represents the number of clusters of this type that are equivalent under rotational/reflectional symmetries of the lattice. (This is similar to the 1/k mentioned two sentences earlier but arises because of rotational symmetry rather than translational symmetry). These normalizing factors keep the values of the $\bar{\Pi}$'s between -1 and $+1$. The bar over the Π reminds us that the quantity is an average over lattice sites and rotated/translated clusters on each of those sites.

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. The aim of a good cluster expansion code is to find the relevant figures in a robust way.

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.

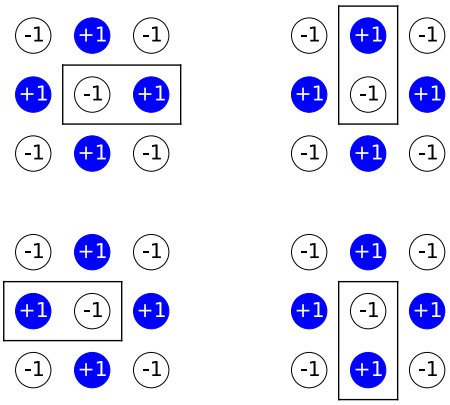


Figure 1: An illustration of all the unique orientations of a cluster containing two vertices.

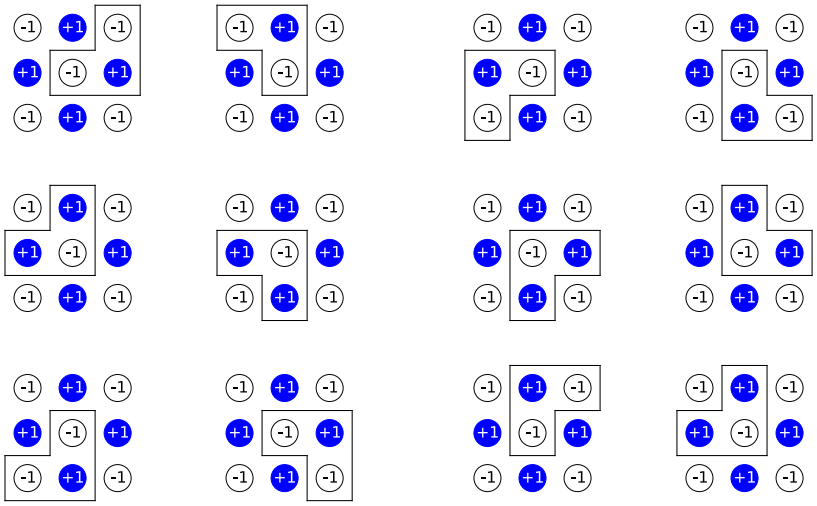


Figure 2: An illustration of all the unique orientations of a cluster containing three vertices. The cluster must enclose the lattice point being considered, which in our case is the one in the center. The rows in the first column are translations of the upper left cluster. For a given row, the columns are 90° rotations. When calculating $\bar{\Pi}$, this process is repeated for every lattice point and every possible cluster.

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 matter 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 of 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 few hours in this session:

- Problem I: Getting a feel for the cluster expansion basis—calculating correlations
- Problem II: A 2D cluster expansion fit by hand
- Problem III: A simple fit with actual Ni-Al data (two-dimensional example)

1 Problem I: Correlations by hand

Consider a one-dimensional lattice. Figure 3 shows all possible symmetrically-distinct configurations with a periodicity (unit cell) of 4. For the clusters shown in the figure calculate the correlations ($\bar{\Pi}$) for each structure. Represent "gray" (blue) atom with $+1$ and a "white" with -1. Note that the "empty cluster" always has value 1 (by definition).

Remember that the correlations are averaged over each site in the unit cell (using periodic boundary conditions when a vertex exceeds the unit cell). Also remember that the correlations are normalized (by the number of sites in the unit cell) so that they are always between -1 and $+1$. As you construct the Π matrix remember that each column of the matrix represents a cluster and each row represents a structure, or configuration, as shown in the figure below. The answers are given at the back of this tutorial. If your answers don't agree with the anwers in the back, ask the instructor (or your neighbor) for help.

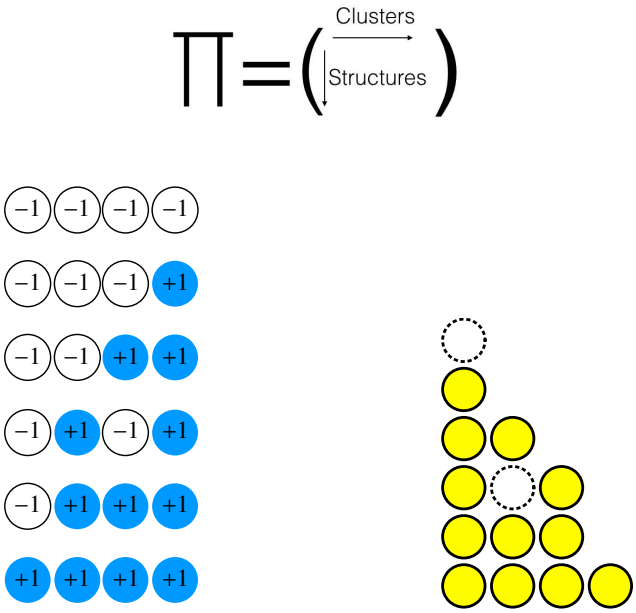
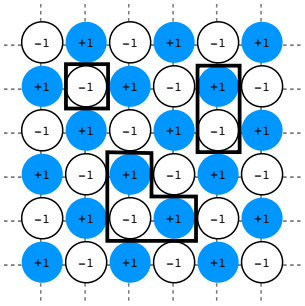
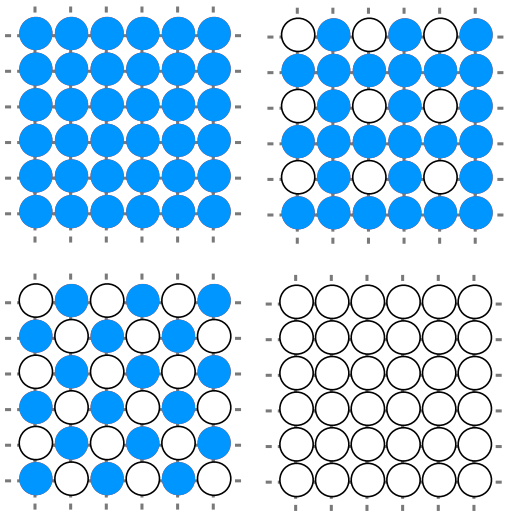


Figure 3: Left: All symmetrically-distinct binary configurations (with periodicity $n = 4$) for a one-dimensional lattice. Right: Six distinct clusters to use in the first problem. Note that the top cluster, the so-called "empty cluster," will always have value 1. Like $x^0 = 1$ in an expansion in powers of x, this term of the cluster expansion is just a constant.



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

Figure 4: 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.



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

Figure 5: 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 figure 4.

2D CE by hand

For this next exercise, we first consider only four structures (see Fig. 5). In this example, we use two hypothetical elements "white" and "blue" that form alloys on a two-dimensional square lattice. The energies for the structures we use here are *completely fictional*.

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

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

Task: Following the same procedure outlined in the introduction to this tutorial, calculate the Π's for each of the three remaining structures in Fig. 5. The $c(2 \times 2)$ structure for which we calculated the Π's together (during the introductory remarks) is shown in Fig. 4. The Π's we computed in the example are shown in the figure as the 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

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:

$$\begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \end{pmatrix} = \begin{pmatrix} \Pi_1^{(0)} & \Pi_2^{(0)} & \Pi_3^{(0)} & \Pi_4^{(0)} \\ \Pi_1^{(1)} & \Pi_2^{(1)} & \Pi_3^{(1)} & \Pi_4^{(1)} \\ \Pi_1^{(2)} & \Pi_2^{(2)} & \Pi_3^{(2)} & \Pi_4^{(2)} \\ \Pi_1^{(3)} & \Pi_2^{(3)} & \Pi_3^{(3)} & \Pi_4^{(3)} \end{pmatrix} \begin{pmatrix} J_0 \\ J_1 \\ J_2 \\ J_3 \end{pmatrix} \tag{8}$$

$$\Downarrow$$

$$\begin{pmatrix} J_0 \\ J_1 \\ J_2 \\ J_3 \end{pmatrix} = \begin{pmatrix} \Pi_1^{(0)} & \Pi_2^{(0)} & \Pi_3^{(0)} & \Pi_4^{(0)} \\ \Pi_1^{(1)} & \Pi_2^{(1)} & \Pi_3^{(1)} & \Pi_4^{(1)} \\ \Pi_1^{(2)} & \Pi_2^{(2)} & \Pi_3^{(2)} & \Pi_4^{(2)} \\ \Pi_1^{(3)} & \Pi_2^{(3)} & \Pi_3^{(3)} & \Pi_4^{(3)} \end{pmatrix}^{-1} \begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \end{pmatrix} \tag{9}$$

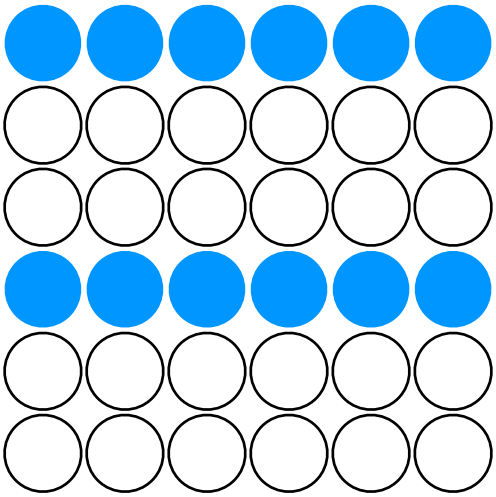


Figure 6: A new structure to use in predicting with your cluster expansion.

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, use the built-in tool in the following link. `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. 6. 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.

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 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 Alloy Theoretic Automated 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). For our first example using UNCLE this file will contain the structural information of the four structures for which we computed the correlations by hand.
3. `CEfitting.in` parameters for the fitting
4. `clusters.out` contains the clusters (i.e., figures) to be 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 beginning with `#` are comments) but the input is not keyword-based—the inputs have to be given in a particular 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 in the master tutorial directory ($HANDSON/hands-on-tutorial/tutorial_8/prepared_input/). 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 (but they are well-commented if you are interested). We will run UNCLE using "mode 12" which directs UNCLE to perform a fit.

```
uncle.x 12
```

to do a simple fit. Much of the information UNCLE reads in is echoed 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 (to avoid line wrapping) and look at fourth column of the file `fittingErrors.1.out` and you'll see that UNCLE found an exact fit (no errors beyond numerical roundoff). This file lists input energies as well as formation enthalpies, ΔH_f (called `DHf_DFT` and `DHf_CE` in the output file).

> Definition: Often, the cluster expansion equations (4) and (5) are not applied to straight total energies (which are large values), but rather to formation enthalpies (which tend to range from a few to a few hundred meV/atom)
>
> $$\Delta H_f(\boldsymbol{\sigma}) = E_{\rm tot}(\boldsymbol{\sigma}) - x\, E_{\rm tot}(\text{pure A}) - (1-x) E_{\rm tot}(\text{pure B}) \quad . \quad (10)$$
>
> $\mathrm{A}_x\mathrm{B}_{1-x}$ 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.

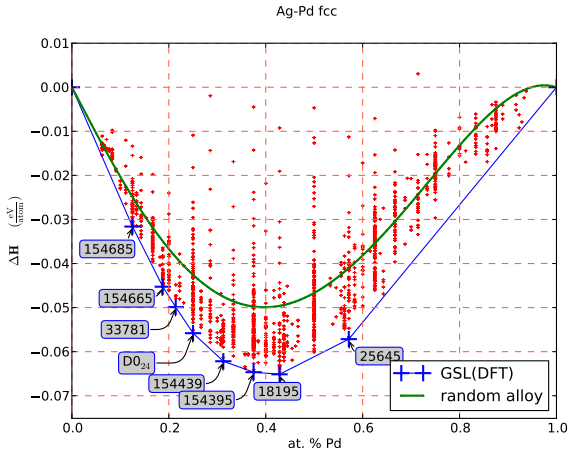


Figure 7: Convex hull of first principles enthalpies for Ag-Pd. The red (small) +'s are the formation enthalpies for different configurations. The straight-line segments form the *convex hull* for all the enthalpies. The configurations marked with a labelled + are configurations that are thermodynamically stable (at $T = 0\,\mathrm{K}$). The smooth line is the energy of a completely random configuration, as predicted by the cluster expansion.

The first two columns of the file (after the structure numbers column) are the concentrations of "white" and "blue". The column labeled `E_DFT` contains the input energies. (DFT is an acronym for Density Functional Theory—it is synonymous with "first principles" or "ab initio".) The next column contains the fitting errors, that is, the differences between the DFT input energies and the CE fitted values (practically zero here). The column labeled `E_CE` contains the energies predicted by the cluster expansion. The next two columns contain the DFT and CE *formation enthalpies*.

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 (which J) pertains to which actual figure by drawing the vertices. In this 2D case, the clusters lie in the y-z plane (so ignore their x-coordinates.)

Another interesting outcome from a cluster expansion are the formation enthalpies as a function of composition x ($0 \leq x \leq 1$). Figure 7 shows an example from a cluster expansion for Ag-Pd. Plotting the enthalpies versus concentration, one can immediately see 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, the *convex hull* construction (blue line) identifies those structures (blue crosses) that are thermodynamically stable at $T = 0\,\mathrm{K}$. 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.

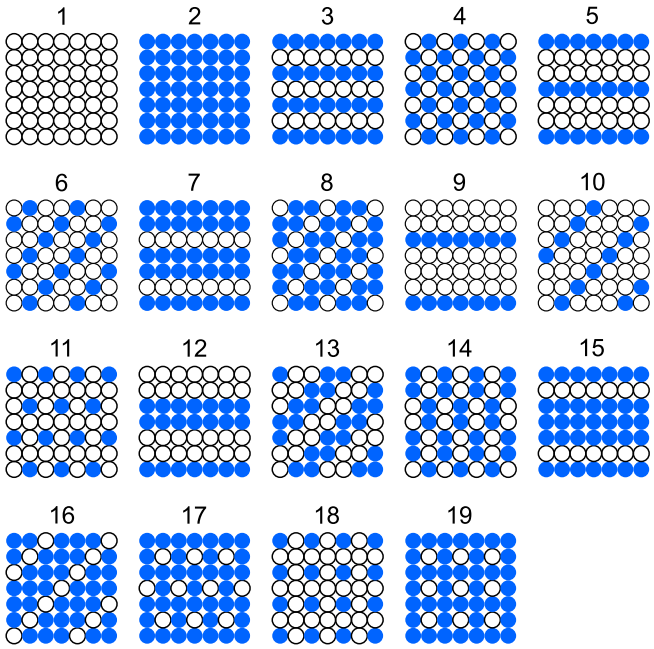


Figure 8: All possible binary configurations for unit cells of 4 atoms/cell or less.

You can plot the convex hull of the input structures using `xmgrace` as you did in all the other tutorials, or you can use our gnuplot scripts to generate pdf figure files: `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 `acroread` or `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. 8. We can predict the energy for each one of these structures using UNCLE (thankfully, UNCLE will compute all the $\bar{\Pi}$'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 21"). 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 3-atom/cell structure you predicted in the first part of the activity (shown in Fig. 6) is number 5 in the enumerated list (top row, last entry in Fig. 8). 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 `prepared_input/problems_I_and_II`. 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. These new structures are more stable than our original input structures, and this is the kind of thing we look for in these models. In practice, we would calculate the energies of these predictions with our DFT code and verify our predictions... but since this is just a toy problem, let's go on to the next part of the exercise.

3 Problem III: Ni-Al on a square lattice

We will now substitute the made-up example of the previous section with real energies computed using first principles (DFT). 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].

If we had time, we could use `FHI-aims`, VASP, QUANTUMESPRESSO, or some other first principles code, to go through the exercise of calculating the formation enthalpies for this exercise ourselves. *All kinds of considerations arise related to the convergence of our computed formation enthalpy: basis set convergence (i.e., energy cutoff), k-point density, choice of pseudopotential, etc.* That kind of exercise is instructive but outside of our scope today. So I have the first-principles formation enthalpies, computed with `FHI-aims` for the 4 configurations of Problem III for you. (The original 2011 version of this tutorial included a section about calculating enthalpies with `FHI-aims`. That part is now included as an appendix at the end of this tutorial so that you can try the calculations yourself if you want to.)

With these formation enthalpies of four structures (Fig. 5) for the 2D Ni-Al system. We can use the results to create the same simple cluster expansion (except with "real" enthalpies), just as in Problem II, and explore the results.

Structure	Total energy [eV/atom]	ΔH_{DFT} [eV/atom]
Ni	−41495.7023	0.0
Al	−6588.3014	0.0
c-(2×2)-NiAl	−24042.8048	−0.8030
p-(2×2)-Ni_3Al	−32769.2366	−0.3845

3.1 Cluster expansion

Task: Repeat the cluster expansion (UNCLE's mode 12) using only the four structures of Fig. 5. 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 I gave you just above. (What was blue should become Ni, white 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 (J's) do you get for each interaction type?

3.2 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 "convex hull and ground state search": 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)?

3.3 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. 8, numbers 5, 6, 7, 8 in the `gss.out` list.

Where are these structures in the ground state search plot?

Note: Ideally, you would compute the DFT-LDA enthalpies of formation of these four structures, which are called (in surface science notation):

- $p(3\times1)$, Ni_2Al and $NiAl_2$, nos. 5 and 7
- $c(3\times3)$diag, Ni_2Al and $NiAl_2$, nos. 6 and 8

but we don't have time for that today. So I've already done it for you and put the enthalpies and structures in a `structures.in_8` file for you.
(Look in the `prepared_input/problem_III/section_3_4/` directory.)

(The appendix includes some discussion of calculating these enthalpies yourself if you were inclined to try.)

DFT-LDA results: Relaxed structures

Number	Structure	Energy [eV]	ΔH_{DFT} [eV/atom]
5	p-(3×1)-$NiAl_2$	−18224.4614	−0.359
6	c-(3×3)diag-$NiAl_2$	−18224.7630	−0.6613
7	p-(3×1)-Ni_2Al	−29860.2041	−0.302
8	c-(3×3)diag-Ni_2Al	−29860.4178	−0.5158

3.4 New ground state predictions?

Task: Update the UNCLE input files to use these 8 input structures and refit (using more than just 4 clusters as in the previous exercises), and do a new ground state search.

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` that you can copy over if you are too lazy to edit it yourself.) Next, you will run a new fit using a larger cluster pool, but first we need to talk about more sophisticated fitting methods.

Up to this point we have been doing fitting by exact inversion. We could continue that approach by adding four more clusters to our `clusters.out` file. In general, *exact inversion is not a reliable strategy.* Exactly-inverted solutions typically have poor predictive capability. In practice (for decades) cluster expansion *has not* been done this way. Instead, the fitting has been done for an *over-determined* problem. That is, the number of input structures is larger than the number of clusters used in the fitting. The fitting is then done to minimize the root-mean-square error via singular value decomposition or similiar techniques. Generally, this leads to much more reliable predictions because the models are constrained by the "extra" data.

The drawback of such an approach (and it is a significant drawback) is that the number of clusters used in the fitting must be only a fraction of the number of input structures (which are typically calculated via DFT). This is a common challenge in many fitting problems, not just the cluster expansion. The shortcoming is obvious—only a few clusters can be included, but it is difficult to know *which* clusters to use because there are *so many* clusters, even of short range, that could be included. Discrete optimization and genetic algorithms have been used to more fully explore the combinatorial cluster space but the former is time-consuming (maybe requiring millions or billions of cpu hours) and the latter requires hand-tuning of the parameters (mutation rate, gene length, population size, etc.) and may still run for days or even a week.

The genetic algorithm approach is included in UNCLE (and was state-of-the-art

until early 2013), so you are welcome to try it, but it has been superseded by a compressive-sensing-based approach which allows us to solve an *under-determined* problem where the number of clusters considered to build our model is far greater than the number of input structures used. No details on compressive sensing will be given here, just an example of using uncle to make a fit with it.

In order to do a fit using compressive sensing, you need the following *additional* files.

1. `structures.holdout` (Just make a copy of the `structures.in_8` in the `problem_III/section_3_4/` directory. For this exercise, the two files are identical.)
2. `CS.in` (Just copy from `CS.in_8`)
3. A new `clusters.out` file that has a lot more clusters included. UNCLE can generate cluster lists (mode 10), but there is already copy (`clusters.out_8`)of a clusters file with 50 clusters (10 each of pairs, triplets, 4-vertex, and 5-vertex clusters)

Then just run mode 15 (`uncle.x 15`). Even though you have 50 basis functions (clusters) to fit to and only 8 data points, the BCS will find a fit. Look at the `J.1.summary.out`.

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 (remove the `struct_enum.out` and 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 do a GSS for structures up to cell sizes of 12, 16, or 20. Try that too if you want. Just edit the first line in `groundstatesearch.in` after removing `gss.out` and `struct_enum.out`.)

4 Problem IV: Compare your results to DFT-LDA

You will see in your GSS plot that the CE of problem III, with more terms, predicts new ground states on both sides of 50%. At 25%, there is one structure that is below the original tie line by just a little bit. The following somewhat arcane unixism will list all the structures at that concentration in the order of their formation enthalpies.
`grep "0.25000␣␣0.75000" gss.out | sort -k 8`.
(Take care to include *2* spaces between the two columns.) Note that at the bottom of the list (low energy structures) there may be some that are degenerate.

We could next compute the enthalpies of the structures that appeared below the original convex hull, by direct DFT, and see how well your model predicted them. 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. 8. Let's make a new fit using all 19 structures as input and then compare the new fit to the predictions of the preceding problem. How much will the model be improved?

Task: We'll answer the question by doing a groundstate search using a model constructed from a larger data set (19 structures) that has more paramaters. So first we need to fit to a larger data set. Use `structures.in_19` in the `prepared_input/problem_IV` directory that contains all 19 of the structures from Fig. 8 and the relaxed enthalpies from the table below. Fit again (mode 15, you'll need a `structures.holdout` file—just use a copy of `structures.in_19`). If you want to see the errors, run mode 13 to create a `fitted_energies.out` then plot the errors. If you do you'll notice that the crosses and the squares don't line up anymore. The crosses show the DFT calculated formation enthalpies while the squares show the formation enthalpies predicted by the cluster expansion (`gnuplot errors_plot.gp`). Once you have the better fit, repeat the groundstate search (mode 21). Compare to the case when you only used 8 structures to construct the fit.

Formation enthalpies of 19 gss structures from Fig. 8

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

5 Problem V: Order-disorder transitions

With the last exercise completed, we are ready to do some thermodynamics. We 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 have started with a few hundred input structures (the compressed sensing formalism gives us a prescription for choosing structures) and we would not have to iteratively refine the fit—it would be a one-shot exercise—but our approach in the tutorial helps you see the underlying ideas.

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) with 19 input structures.

5.1 Nearest-Neighbor-only CE

To get some intuition for Monte Carlo, we'll start with a simple, nearest-neighbor-pair-only cluster expansion, created 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 a single, nearest-neighbor-interaction-only fit 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. You need to change the number of clusters used in the fit in `CEfitting.in`. 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 to 2 (*not* to 1—UNCLE always includes the constant term as well [UNCLE will set it to zero in this example, though, because there is no constant shift when we use formation enthalpies.]).

Task: Run UNCLE with mode 12 again to create a fit. The errors will be large. How large? (mode 13 and then `gnuplot errors_plot.gp` and look in the `fittingErrors.1.out` file.) You can't expect too much from a one-parameter model.

Now, do a Monte Carlo (MC) simulation by using mode 30 of UNCLE. The case we will try is fixed-concentration at $x = 50\%$ to find the temperature where the 50:50 ground state structure becomes more stable than a disordered phase. 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`. The output you want to look out is called `Tc.pdf`.

5.2 "Best we can do" CE

Task: Starting from the same set of input structures, this time create a cluster expansion using the set of 50 clusters, as you did in problem IV, (mode 15). **Task:** What does the expansion look like? By how much does our fitting error change? (Make a plot—mode 13 and `gnuplot errors_plot_bcs.gp`.) Predict ground states up to 12 or 16 atoms/cell (modify `groundstatesearch.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 in the `clusters.out`, but that's beyond the scope of this problem.

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 (in CEfitting.in change the number of clusters used from 2 back to 10).

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.

Appendix

5.3 Answers to Problem I

$$\bar{\Pi} = \begin{pmatrix} 1 & -1 & 1 & 1 & -1 & 1 \\ 1 & -0.5 & 0 & 0 & 0.5 & -1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 1 & 0 & 1 \\ 1 & 0.5 & 0 & 0 & -0.5 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

5.4 Answers to Problem 1.1-1.3

```
1.0  1.0  1.0  1.0
1.0  0.5  0.0 -0.5
1.0  0.0 -1.0  0.0
1.0 -1.0  1.0 -1.0
```

$\Pi^{-1} =$

```
0.250  0.000  0.500  0.250
0.000  1.000 -0.500 -0.500
0.250  0.000 -0.500  0.250
0.500 -1.000  0.500  0.000
```

$J = \{-0.04, -0.0075, 0.025, 0.0125\}$

$\Pi_{p(3\times1)} = \{1, -1/3, 1/3, -1/3\}$

$\Pi_{p(3\times1)} \times J = -0.0333$ eV/atom

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/∼avdw/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).

6 Appendix

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 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. (10)] 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. 5. (Blue corresponds to Ni and white 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?

6.1 4.3