Cluster Expansion Journal

Ryan Botts

1 6/20/07-7/1/07 Background of the Cluster Expansion Problem

I began this project by reading many related articles getting a feel for the current approaches to approximating properties of crystalline solids. Here is what I found in regards to the current methods.

Many properties of crystalline solids are determined by their lattice structures, however, other properties depend not only on the structure and the proportions of each substance, but where they are located in the solid. We also know that crystalline solids have stable states which minimize certain properties such as enthalpy of formation. In order to compute these quantities one of the first intuitive ideas to come to mind would be to predetermine some maximum unit cell size and then, insert the constituents into the sites in this cell. Once you have a possible configuration you could then compute the desired quantity based on the interactions of the individual particles. Minimize the function over the possible configurations and you are done. Since we know that it is highly plausible that there is a reasonable maximum unit cell size it is possible to enumerate all possible configurations and based on fitting a few bits of test data, predict the structures of new solids. This leads to the methods of cluster algebra.

2 7/1/07-7/9/07 The Technicalities

During this time period I wanted to become more familiar with the terminology, so I wrote out the methods currently used in describing the problem, and that description is below. This was a great benefit in understanding the

problem and the notation. Spending more time with material and meticulously working through it is one of the best ways to become "fluent" with it.

We begin with an M-component system, although we will usually consider the case of M=2 or M=3. Since we are assuming that there is some maximum unit cell size we have at most N lattice points to consider. We will let σ_p represent the component occupying site p and call these the spin operators. Let $\overrightarrow{\sigma} = (\sigma_1, \sigma_2, ...\sigma_N)$ be the configuration vector. If we take $\{\Pi_\alpha\}$ to be a complete orthonormal set of interaction functions, then any function $f$which depends only upon the configuration can be written as

$$f(\overrightarrow{\sigma}) = J_0 + \sum_i J_i \Pi_i(\overrightarrow{\sigma}) + \sum_{i,j} J_{i,j} \Pi_{i,j}(\overrightarrow{\sigma}) + \sum_{i,j,k} J_{i,j,k} \Pi_{i,j,k}(\overrightarrow{\sigma}) + ... \quad (1)$$

Where the $J's$ can be found by taking some sort of appropriate inner product and the i.j and k represent lattice sites.

One of the first encountered difficulties is how to find the $\Pi's$. This can be done using intuition about the types of multi-body interactions. Intuitively we have a good feeling for what types of interactions would be most significant (i.e. nearest neighbors). So if we know $\overrightarrow{\sigma}$ and $f(\overrightarrow{\sigma})$ for a few structures then we can compute the $J's$ and use these to compute $f(\overrightarrow{\sigma})$ for new $\overrightarrow{\sigma}$. Data has shown that this assumption is fairly accurate, however based on the number of of possible configurations this problem remains to computationally costly using current means of computation. Another difficulty is that with these methods it seems as though the theoretical data still shows a large amount of variation for the test data. Even if we could get more accuracy here there is yet the problem that we need to be able to do these computations for larger component systems and for multiple different lattice structures, leading us to believe that perhaps there we should try another way.

3 7/10/08-7/17/08: The New Approach

In order to make these computations faster we wish to remove the restriction that the functions used in these approximations represent actual interactions. We are going to construct functions which operate on the various lattice sites using a least squares fitting technique. We will require that our functions are symmetric with respect to the lattice structure, that they are invariant

under any translations or rotations of the structure, and that they preserve the periodicity of the lattice.

The first difficulty we encounter is making sense of our functions and what it means to evaluate them for our data. Our data, D, is presented to us as a list of locations, given as vectors, and the components at these locations. To make sense of this we will first make a sufficiently large copy of the unit cell and then for given compositions, we will assign the components locations in the lattice. It will be necessary to make better sense of a sufficiently large copy, but this will have to do with the number of active sites involved in evaluating our functions and the periodicity of the lattice. It will only be necessary to store this structure once, at the beginning of any computation, hence it will not effect the speed of the calculations later.

Consider a function

$$f(\overrightarrow{\sigma}) = \prod_{i=1}^{n} f_i(\sigma_i). \tag{2}$$

Now, to ensure that f is invariant under the group operations on the lattice we define

$$F(\overrightarrow{\sigma}) = \frac{1}{|G/G_f|} \sum_{g \in G/G_f} f(g\overrightarrow{\sigma}), \tag{3}$$

where G/G_f represents the group G, of translations and rotations which act on the lattice, and the group $G_f = \{g \in G | f(g\overrightarrow{\sigma}) = f(\overrightarrow{\sigma}) \forall \overrightarrow{\sigma}\}$. Our function F now has the desired property that it is invariant under G. At first we believed that f must also be periodic. I have proofs that f and F will both have the same period (this is actually quite obvious as under the definition of periodicity each f evaluated at any given data point and that same point shifted in some multiple of the period, say $k\alpha$, then each evaluation of f involved in computing F will be the same so F will be the same. Further, if we consider a second function f^* where f^* and f are identical, only we naively take f^* to have twice the period of f, then F and F^* are identical. This can be shown by considering $|G/G_f|, |G/G_{f^*}|$and $|G_f/G_{f^*}|$. the properties of algebra tell us that, although G is infinite, since $|G/G_f|$ is finite $|G : G_{f^*}| = |G : G_f||G_f : G_{f^*}|$ and since $|G_f : G_{f^*}| = 2$, as there are only the two obvious cosets,

$$F^*(\overrightarrow{\sigma}) = \frac{1}{|G/G_{f^*}|} \sum_{g \in G/G_{f^*}} f^*(g\overrightarrow{\sigma}) \tag{4}$$

$$= \frac{1}{|G/G_{f^*}|}\Big[\sum_{g\in eG_f} f^*(g\overrightarrow{\sigma}) + \sum_{g\in \alpha G_f} f^*(g\overrightarrow{\sigma})\Big] \quad (5)$$

$$= \frac{1}{|G/G_{f^*}|}\Big[\sum_{g\in G/G_f} f(g\overrightarrow{\sigma}) + \sum_{g\in \alpha G_f} f(g\overrightarrow{\sigma})\Big] \quad (6)$$

$$= \frac{1}{|G/G_{f^*}|}\Big[\sum_{g\in G/G_f} f(g\overrightarrow{\sigma}) + \sum_{\alpha g\in G_f} f(\alpha g\overrightarrow{\sigma})\Big] \quad (7)$$

$$= \frac{1}{|G/G_{f^*}|} * 2 \sum_{g\in G/G_f} f(g\overrightarrow{\sigma}) \quad (8)$$

$$= \frac{1}{|G/G_f|} \sum_{g\in G/G_f} f(g\overrightarrow{\sigma}) \quad (9)$$

$$= F(\overrightarrow{\sigma}) \quad (10)$$

Finally, when we define an inner product between two functions f and h as

$$< f, h >_D = \sum_{\overrightarrow{\sigma}\in D} \frac{1}{|G/G_D|} \sum_{g\in G/G_D} F(g\overrightarrow{\sigma})H(g\overrightarrow{\sigma}), \quad (11)$$

that $< f, \phi >_D = < f^*, \phi >_D$.

After working with a few examples, I found that the assumption that f is periodic was not used, and that it is much simpler to assume the condition that f acts on a finite number of active sites (this is simpler than assuming an equivalent condition that f is evaluated on an infinite number of lattice sites, where only a finite number of the f_i are not 1). In order to make sense of G_f it will also be necessary to assume that the active sites on which f operates is periodic. This assumption does not actually make any limitations on f as we can extend f to a larger number of active sites by simply assuming that $f_i(\sigma_i) = 1$ as we would in the infinite lattice case. I have shown that no matter how we extend the domain of f, all inner products with f will still be unchanged.

4 Week 7/18/07 -7/25/07: Abstracting the Problem

The previous section's work turned out to be of much use, but there were too many difficulties in hashing out the problem. I found that I needed to

start from the beginning with the setup and that one thing that would make the problem much easier would be to remove the context of problem and try to abstract it into an entirely mathematical setting. I found that many of the previous proofs were adaptable to the new setting, so it was much help to save all of the old work and that the ways that didn't work provided much understanding of the structure of the data. For example I began to understand the group symmetries and how the group acting on the data would leave some crystal structures fixed. It is already apparent that one of the great difficulties in this problem will be taking the data and finding a useful way to evaluate our function over it.

This week I rewrote the Cluster Expansion notes. I reformulated the problem so that it would be more independent of the cluster expansion setting. As per Dr. Mohlenkamp's suggestion, I am taking our lattice to be a countably infinite vector $\overrightarrow{\sigma}$ and our set G to be an infinite set of permutations on $\overrightarrow{\sigma}$. I convinced myself that $G_\sigma = \{g \in G | \sigma_i = \sigma_{gi} \forall \sigma\}$. was a definition of G_σ consistent with what we were already using and proved that F would have the properties we desired. See the Cluster Expansion Notes.

5 7/25/07 - 7/30/08: Using the Alternating Least Squares Method With Our New Inner Product

During this period of time I worked out a good copy of the Cluster Expansion notes. I found that $G_{\vec{\sigma}}$ does, in fact, depend upon the $\vec{\sigma}$, and is not defined on the entire data set. It appears that this will be the most difficult thing to implement in the code. We are going to have to find a way to figure out which translations and rotations leave each data set fixed. I worked out the ALS procedure for the single separable function taking into account that it was only used to construct a more sophisticated symmetry invariant function with which to approximate our crystal data. One natural extension here would be to extend to a linear combination of separable functions to use in our approximations. I also assumed that we had a linearly independent basis of functions, $\{\phi_k\}_{k=1}^M$, from which to approximate the f_i and spent time developing a procedure to find the coefficients, which would construct f, minimizing the least squares error, using conjugate gradient techniques. I had to get reacquainted with the conjugate gradient method, and also spend

some time working with matrices in higher dimensions. I still need to hone the latter skill. I also finished finding the operation counts up to this point. I also worked out most of the details for the case where a pre-determined basis set for f is not given. This may not be useful in implementation, but i will keep it around. It was a much easier problem than using a pre-determined basis for f.

From this point I am going to begin looking into Python code which could be used to test this procedure. I began reading "Dive into Python." Some light bathroom reading. Until Next week...

6 7/30/07-8/07/08: Using a Predetermined Basis

After tidying up the details on using an unknown basis for our separable functions, I learned that Dr. Mohlenkamp had removed this method from the previous paper as it did not provide good results. So although, not that good in practice, it was a much easier setting to work out the ALS procedure. Overall, it made it much easier to understand and work out the notation in pre-determined basis setting. I also learned that we are going to extend this to a sum of separable functions.

During this time I also completed writing the procedure we had up to this point. I found a few errors in the operation counts from before and fixed those. Next week I will spend most of the time learning Python.

7 8/8/07-8/15/07: Learning A New Tool and Some Background on the Chemistry

I tried to research the significance of the reciprocal vector and only found that it is a projection of the translation vectors into 1-D. I found that many mechanical properties of crystals can be ascertained by their reciprocal vectors, however for our purposes I did,'t find anything that seemed that useful, as they do not represent any physical properties of a structure. I had been told that these vectors could be found using the inverse matrix of the translation vectors or something of this sort, but I did not find anything of this sort in the literature. I will have to confirm that this is consistent with the

definition of the reciprocal vector used in our computing.

I read "Dive into Python" and wrote several trivial classes using dictionaries, lists and tuples. Lists are a very useful tool that is not found in any of the other languages I know. Lists can be lengthened or shortened at any time using the .append and .remove command. In general Python conveniently does not require variable declaration, which seems to be a very nice way to fully utilize an object oriented language. Methods can be defined for undeclared variable types and can be used in any setting. One difficulty I did find with this is that as I read previously written code it was often difficult to trace variable types, hence it was difficult to know which explicit functions were available. I found this to be an especially difficult problem as I tried insert code accounting for the group symmetries in the inner products. I eventually learned the "type" command in Python, allowing you to determine what variable type you have during a run. This is the first time I have ever worked on code written by someone else and any code of this length. I have found that it can be very difficult to read someone else's code and to understand how it works. I believe that during the next week I will find that much time is spent just trying to understand what code is already in place, what needs to be modified and what is already in place.

Some other useful things to note in Python are the use of indices ranging over multiple variables. One other thing is the use of a command to insert variables of different variable types into strings. It makes the code to output information much shorter to write when formatting output.

These weeks did not result in many tangible products. It appears that there will be many times in researching when it will be necessary to learn things not directly related to the task at hand. I believe this is going to be a very valuable tool in the future.

Finally, with this newfound knowledge, I began to break apart the previously written code in an attempt to understand what is already there and while trying to find where the methods, and classes are which need to be modified in order to account for the group symmetry.

8 8/16/07-8/23/07: Implementing the Newly Learned Tools, and Handling Data

Here I attempted to break into the old code and write several modules which would adapt what was already in place to accomodate the group structure. This proved to be much more difficult than I anticipated as several of the previously written classes did not work. I encountered problems with missing modules on Turing and on my desktop and laptop. I attempted to set Python up, again, on my laptop, but I could not get Scipy and Numpy to work. These prebuilt modules were difficult to install, and in the end I had no success. I should have spent the time just working on turing. I did learn a lot about the Numpy and Scipy packages, and what components are available to the mathematician. There are many tools here that I had not seen before. These would be very valuable tools for use later on. One of the most difficult problems that I have found is how to handle our input data. This data consists of vectors describing the locations of components, the components and translation vectors. It is not easy to see how these can be easily turned into active sites especially as our active sites will have to be rotated and translated under the group symmetries. In order to develop fully functional code it will be necessary to convert these into a list of active sites.

9 8/24/08-9/1/08: Data and Data Types

I have so far had much difficulty figuring out how to process our data. As noted before it consists of a few vectors describing locations and their components along with translation vectors according to the periodicity of the data. As it will be necessary to find all possible translations and rotations of the data in order to construct the group symmetry used to compute the inner products, I had to understand how the data is processed better. It appears that the data is read in and converted to a list of locations and components. Process imports files containing BCC data. this data is first read in as a list of PointValues which are then converted to a BCCStructure. This structure is passed the location vectors, and the translation vectors among other things. The processing here takes the list of inputs, finds all possible rotations and locations for the origin and constructs new BCCStructure's with these in mind, if any of these group symmetries result in repeated structures a weight is adjusted to account for this and avoid storing the repeats. Note

that these BCCStructure's will have to be copied to cover all active sites.

At first i did not realize that previous work had found the group of symmetries, so I attempted to construct this. Aweful task, never do something which has already been done. The rotations were easy to find, however the possible origins, I did not realize were actually given to us. I thought that we would have to construct several unit cells, and from this find the possible origins. After some time on this I discovered that the only possible origins were limited to the locations of the original components used to construct the crystal. This made life much easier.

Now, I am still working on making modifcations to allow for the group symmetries, however the work is much simpler as the groups are already computed for us. up to this point I have not been able to get BCC to run, if I can do this I will be able to see a test of the entire procedure on some 2-D data.

I am also going to work on revising the notes I have prepared on this subject. The revisions should help take what I have, make it more readable, and further increase my understanding of the topic.