龙讯教程

ATAT Interface

Contributed by Long Xun

The Alloy-Theoretic Automated Toolkit (ATAT)

The alloy-Theoretic Automated Toolkit (ATAT) is a generic name that refers to a collection of alloy theory tools developed by Axel van de Walle, in collaboration with various research groups:

- Codes to construct cluster expersions from first-priciples (maps and mmpas).
- Codes to perform Monte Carlo simulation (emc2 and memc2) of lattice models in order to compute thermodynamic properties of alloys, starting from a cluster expansion.
- Codes to perform lattice dynamics calculations (fitfc, fitsvsl, svsl)
- Codes to calculate electronic and magnetic free energy contributions (felec, fmag, fempmag) using simple physical or semiempirical models.
- Utilities to combine all of the above to generate free energies that include configurational, vibrational and electronic contributions (mkteci).
- Codes to generate Special Quasirandom structures (SQS), to model disordered solid solutions (mcsqs, gensqs) and to enumerate structures (genstr).

The Alloy-Theoretic Automated Toolkit (ATAT)

- A large library of pre-computed SQS and structure prototypes.
- Extension of the two above tools that allow the contribution of so-called reciprocal-space cluster expansion, which are useful to model the energetics of alloy exhibiting a large atomic size mismatch.
- Tensorial cluster expansions (gce).
- Elastic constant calculations (calcelas).
- Structure conversion utilities (subcells, supercells, coordinate system changes, file format, etc.) (cellcvrt, wycked, etc)
- Scripts to automate tasks (foreachfile, sspp, getvalue, getlines, etc)
- Codes to generate CALPHAD databases (sqs2tdb)
- Utilities to interface the above tools with first-principles codes, such as PWmat (runstruct_pwmat, runstruct_vasp, runstruct_abinit, runstruct_gulp, etc.).
- Job control utilities that enable the efficient use of a cluster of workstations to run the first-principles codes that provide the input to the above codes (pollmach).

Download:

Download ATAT package from: https://www.brown.edu/Departments/Engineering/Labs/avdw/atat/, and PWmat interface from: http://www.pwmat.com/module-download , then upload them to the server

ATAT Installation:

If you have an earlier version of ATAT installed, please delete or rename the former atat directory before proceeding, e.g.,

```
$ mv atat atatold
```

Then, type

```
$ gunzip atatX_XX.tar.gz
$ tar –xvf atatX_XX.tar.gz
```

ATAT Installation:

where X_XX is the current version number. These commands create a directory called atat in the current directory. It contains the whole package. For future reference, I`ll call the whole access path to this directory atat.

Type

```
$ cd atat
```

and open the file makefile with a text editor and look for the line BINDIR=$(HOME)/bin/. Changes $(HOME)/bin/ to point wherever you want to put the executables. Type

```
$ make
```

If no error message appears, proceed with the next steps, otherwise consult ATAT manual Chapters 8.

```
$ make install
```

rehash (not necessary with bash shell)

PWmat interface Installation:

First you should make sure the PWmat interface ‘pwmat_atat_interface.1.0.tar.gz’ and ATAT installation packages under the same directory. If you have an earlier version of PWmat interface installed, please delete or rename the former pwmat directory before proceeding e.g,

```
 $ mv pwmat pwmatold
```

Then type

```
$ tar –zxvf pwmat_atat_interface.1.0.tar.gz
$ mv pwmat atat/glue/
$ cd atat/glue/pwmat
```

and open the file makefile with a text editor and look for the line BINDIR=$(HOME)/bin/. Changes $(HOME)/bin/ to be same as ATAT makefile. Type

```
$ make install
```

Command Reference:

1.runstruct_pwmat

```
runstruct_pwmat [-w file] [-d depth] [-p] [-lu] [-ng] [-nr] [-ex] cmdprefix
  where file is an optional alternate wrap file (default: pwmat.wrap)
    If the wrap file is not found in the current directory,
    it searches in the parent directories, up to number specified by depth (default is 5).
  -p means preserve pwmat output file
  -ng means do not generate a pwmat.in file, use the existing one.
  -nr means do not run pwmat, just generate input files
  -ex means do not generate pwmat.in, do not run pwmat, but extract info from pwmat output file
  cmdprefix is the prefix needed for pwmat to run on a remote machine,
          such as "node -s node2"
```

Command Reference:

2.ezpwmat

-c: Cell patch. This option will convert the cell you entered into a cell that has the symmetry needed for pwmat to work properly. The atom positions are altered correspondingly.

-s: Static run. After the relaxations are completed, this option makes pwmat start again.

-n: do Not run pwmat. If you want to create the input files now and run them later, perhaps on another machine.

-p: path of the PWmat ("PWmat" by default). You can include addition prefix such as:

```
ezpwmat -p PWmat  pwmat.in
```

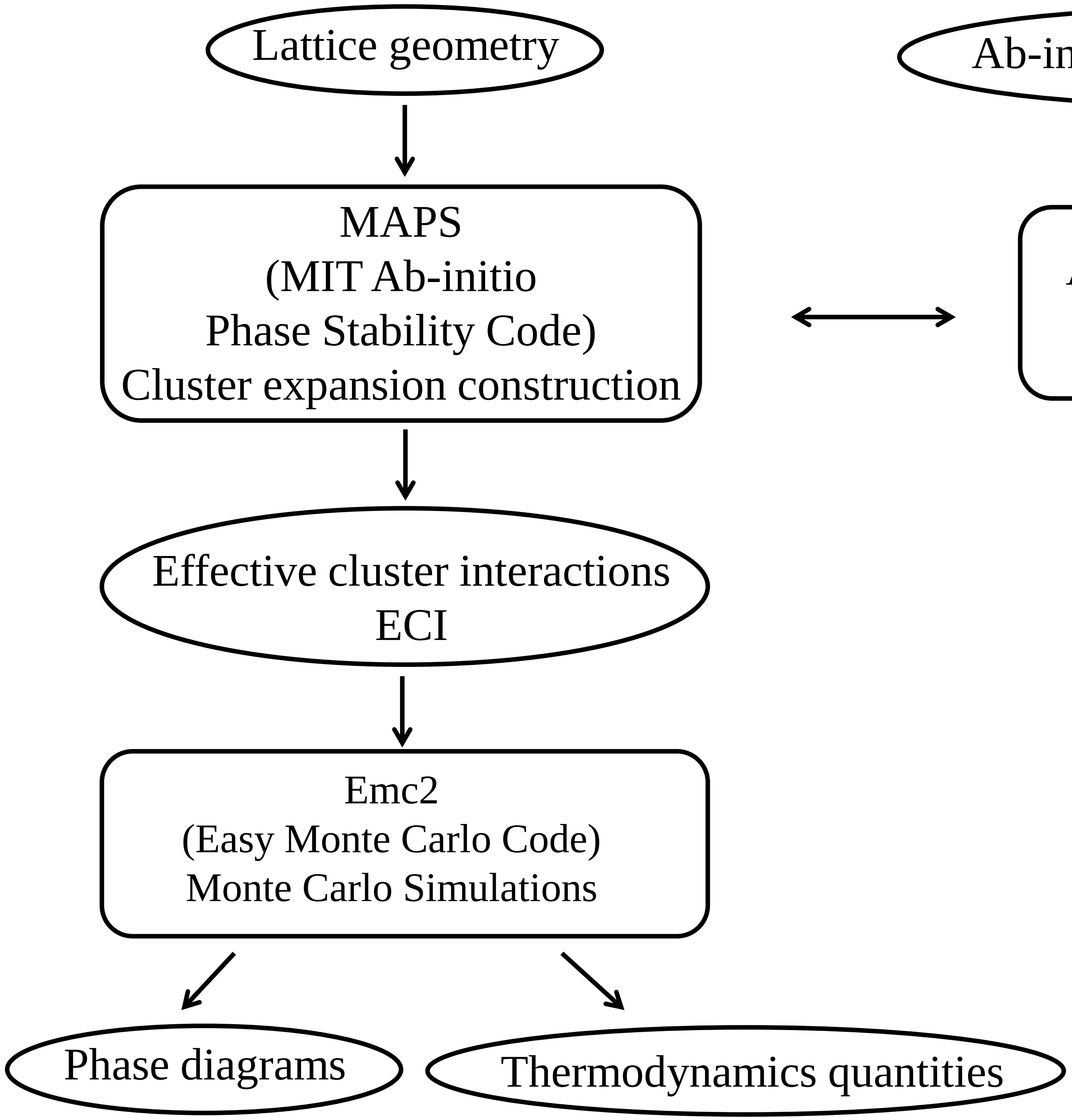
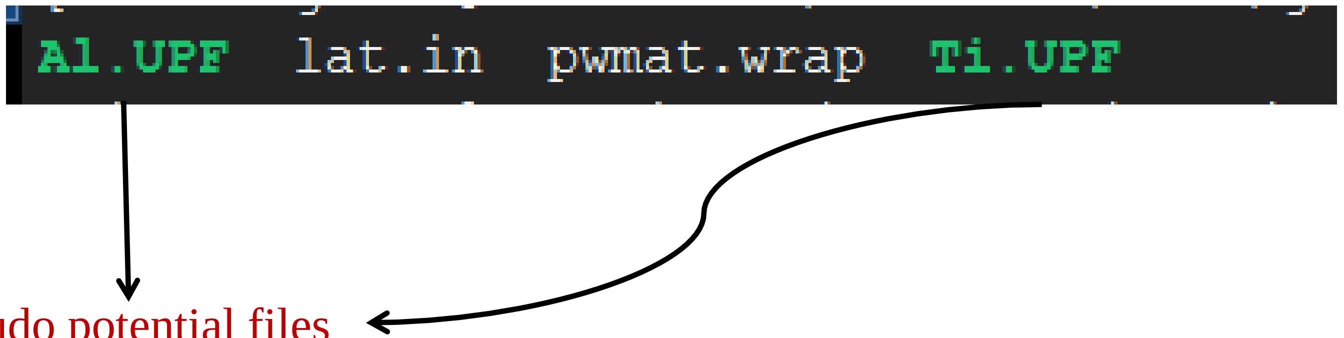


Diagram 1.1 Methodology implemented in atat for the computation of thermodynamic properties from first principles. The automated construction of the cluster expansion is performed by the maps code. Whenever needed, maps requests the calculation of the formation energy of various atomic configurations by a first-principles code (such as PWmat). The necessary input files are created and the resulting output files are parsed without requiring user intervention. The output of maps is a set of effective cluster interactions that define a computationally efficient Hamiltonian that can be used to perform Monte Carlo simulations with the emc2 code. These simulations provide thermodynamic properties and phase diagrams that can be used to create thermodynamic databases or supplement existing ones.

Cluster expansion construction using the MAPS and PWmat

1.Input files

The maps code needs two input files: one that specifies the geometry of the parent lattice (**lat.in**) and one that provides the parameters of the first-principles calculations (pwmat.wrap). Table 1.1 gives two annotated examples of a lattice geometry input file. The package includes ready-made lattice files for the common lattice types (e.g. bcc, fcc, hcp). It also includes an utility that automatically constructs multiple lattice geometry input files for common lattices. In addition, pseudo potential files need provided in the calculation directory and the potential files name format must be atomtype.UPF (for example Al.UPF Ti.UPF). For instance:



```
Al.UPF  lat.in  pwmat.wrap  Ti.UPF
```

Cluster expansion construction using the MAPS and PWmat

Table 5.1: Examples of lattice geometry input file lat.in. Typically, the coordinate system entry is used to define the conventional unit cell so that all other entries can be specified in the normalized coordinates that are the most natural for the symmetry of the lattice. The input lattice parameters do not need to be exact, as the first-principles code will optimize them.

Example 1: hcp Ti-Al system

3.1 3.1 5.062 90 90 120	(Coordinate system: a b c α β γ notation)
1 0 0	(Primitive unit cell: one vector per line
0 1 0	expressed in multiples of the above coordinate
0 0 1	system vectors)
0 0 0 Al,Ti	(Atoms in the lattice)
0.6666666 0.3333333 0.5 Al,Ti	

Cluster expansion construction using the MAPS and PWmat

Example 2: rocksalt CaO-MgO pseudobinary system

```
4.1 4.1 4.1 90 90 90        (Coordinate system: a b c α β γ notation)
0 0.5 0.5                   (Primitive unit cell: one vector per line
0.5 0 0.5                   expressed in multiples of the above coordinate
0.5 0.5 0                   system vectors)
0 0 0 Ca,Mg                 (Atoms in the lattice)
0.5 0.5 0.5 Al,Ti
```

Cluster expansion construction using the MAPS and PWmat

The first-principles input file is usually less than 10 lines long, thanks to the dramatic improvements in the user-friendliness of most modern first-principles codes. For instance, in the case of the PWmat code [13, 12], a typical input file is given in Table 1.2. Examples of such input files are provided with the package. Note that atat contains a utility that enables the automatic construction of k-point meshes from a single parameter defining the desired target k-point density, the number of k-point per reciprocal atom (KPPRA).

Table 1.2: Example of first-principles code input file (example given for the PWmat code). It is especially import to verify that the KPPRA parameter is

[etot.input]	
1 4	
IN.ATOM = atom.config	
JOB = RELAX	
RELAX_DETAIL = 1 500 0.01 1 0.01	
Ecut = 70	
Ecut2 = 280	(See PWmat manual for a description of the above 6 parameters)
KPPRA = 1000	(Sets the k-point density (K Point Per Reciprocal Atom))
DOSTATIC	(Performs a ‘”static run” –see pwmat manual)

Cluster expansion construction using the MAPS and PWmat

2.Running the code

The maps code is started using the command

```
$ maps -d &
```

Where the option -d indicates that all default values of the input parameters should be used, which is what most users will ever need. (The optional parameters can be displayed by typing maps by itself and further is help is available via the command maps –h.) The trailing & character cause the command to execute in "background" mode. In this fashion, maps can continuously be on the lookout, responding to various "signals", while the user performs other tasks. (The ongoing discussion assumes that the code is run under a UNIX environment within a shell such as sh, csh, tcsh or bash.)

The process of constructing a cluster expansion from first-principles calculations can be summarized as follows.

Cluster expansion construction using the MAPS and PWmat

The process of constructing a cluster expansion from first-principles calculations can be summarized as follows:

① Determine the parameters of the first-principles code that provide the desired accuracy.

② Let maps refine the cluster expansion

③ Decide when the cluster expansion is sufficiently accurate.

Typically, one calibrates the accuracy of the first-principles calculations using the “pure” structures of the alloy system of interest. To generate the two “pure” structures, type

$ touch ready

This creates a file called ready which tells maps that you are ready to calculate the energy of a structure. Within 10 seconds, maps relies with

Finding best structure…

done!

Cluster expansion construction using the MAPS and PWmat

maps has just created a directory called 0 and, within it, a file called str.out that contains the geometry of one of the two “pure” structures. If you type touch ready once more, the other “pure” structure is written to 1/str.out. You now need to launch the first-principles code to calculate the energy of each structure. Type

```
$ cd 0
```

to go into the directory of the first structure, type

```
$ runstruct_pwmat &
```

After this command has successfully terminated, display the energy of that structure and go back to the initial directory

```
$ cat energy
$ cd  ..
```

and edit the file defining the first-principles code parameters, for example,

```
$ vim runstruct_pwmat
```

Cluster expansion construction using the MAPS and PWmat

Then you rerun the calculations to check by how much the calculated energy has changed:

$ cd 0

$ runstruct_pwmat &

$ cat energy (After the calculations are completed)

$ cd ..

This process is repeated until the user is satisfied with the precision of the calculation (that is, if the energy has become insensitive to changes in the input parameters within the desired accuracy). A similar study should also be performed for the other “pure” structure (labeled structure 1) and, if one is really concerned with precision, for a few structures with intermediate concentrations.

Cluster expansion construction using the MAPS and PWmat

Once the appropriate ab initio code parameters have been determined, the fully automated process can begin. From within the directory where maps was started, type

```
$ pollmach runstruct_pwmat
```

to start the job manager that will monitor the load on your local network of workstations and ask maps to generate new structures (i.e. atomic arrangements) whenever a processor becomes available. Note that the first time the command is run, instructions will appear on screen that explain how to configure the job dispatching system in accordance to your local computing environment. Once this configuration is complete, the above command should be invoked in the background by appending a “&” to it.

Cluster expansion construction using the MAPS and PWmat

3.Output of MAPS

While the calculations are running, you can check on the status of the best cluster expansion obtained so far. The file log.out contains a brief description of the status of the calculations, such as the accuracy of the cluster expansion and various warning messages. Most of the messages pertains to the accurate prediction of the so-called ground states of the alloy system. The ground states, which are the structures that have the lowest energy for each given concentration, are extremely important to predict accurately because they determine which phases will appear on the phase diagram. For more information, please refer to the ATAT manual.

Cluster expansion construction using the MAPS and PWmat

4.Check the results:

You can view MAPS output directly, or use the command “mapsrep”, it can display the output of maps in graphical form.

$ mapsrep (the detail output of mapsrep, please refer to ATAT manual)

This command displays, in turn

- The log.out file described earlier.
- The formation energy of all structures whose energy is known from first-principles calculations, as well as the predicted energy of all structures maps has in memory. The convex hull of the ground states among structures of known energy is overlaid while the new predicted ground states (if any) are marked by an “×”. (Note that this ground state line is only meaningful if the log.out file contains “Among structures of known energy, true and predicted ground states agree.”)
- The formation energy of all structures calculated from first principles and associated ground state line.

Cluster expansion construction using the MAPS and PWmat

- A plot of the magnitude of the Effective Cluster Interactions (ECI) as a function of the diameter of their associated cluster (defined as the maximum distance between any two sites in the cluster). Pairs, triplets, etc. are plotted consecutively. This plot is useful to assess the convergence of the cluster expansion. When the magnitude of the ECI for the larger clusters has clearly decayed to a negligible value (relative to the nearest-neighbor pair ECI), this is indicative of a well-converged cluster expansion.
- A plot of the residuals of the fit (i.e. the fitting error) for each structure. This information is useful to locate potential problems in the first-principles calculations. Indeed, when first-principles calculations exhibit numerical problems, this typically results in calculated energies that are poorly reproduced by the cluster expansion.

Cluster expansion construction using the MAPS and PWmat

5. Stop MAPS

When the user is satisfied with the results (which are constantly updated), maps can be stopped by creating a file called stop in the current directory using the command:

```
$ touch stop
```

while the job dispatching system can be stopped by typing

```
$ touch stoppoll
```

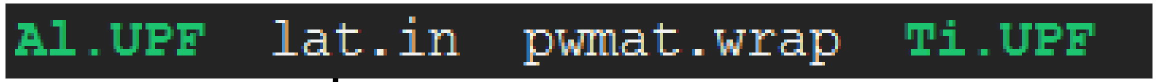
A cluster expansion can be considered satisfactory when

- All ground states are correctly reproduced and no new ground states are predicted. (The log.out file would then indicate that Among structures of known energy, true and predicted ground states agree. No other ground states of n atoms/unit cell or less exist.)
- The crossvalidation score, as given in the log.out file, is small (typically less than 0.025 eV).
- Optionally, it is instructive to verify that the magnitude of the ECI decays as a function of the diameter of the
- corresponding cluster and as a function of the number of sites it contains.

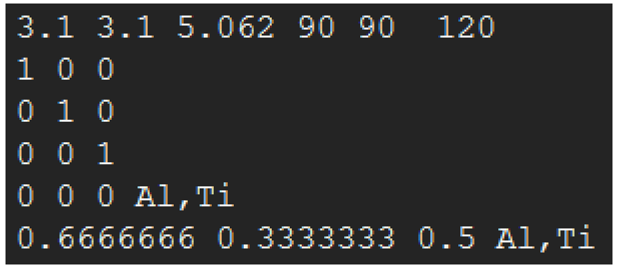
Cluster expansion construction using the MAPS and PWmat

4.Example

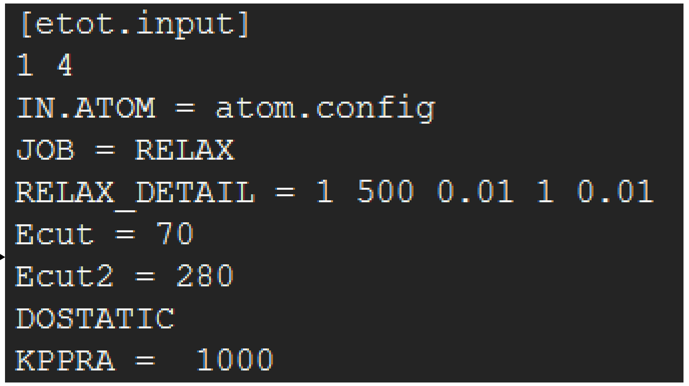
Cluster expansion construction for Ti-Al system, the input files as following:



```
Al.UPF   lat.in   pwmat.wrap   Ti.UPF
```



```
3.1 3.1 5.062 90 90  120
1 0 0
0 1 0
0 0 1
0 0 0 Al,Ti
0.6666666 0.3333333 0.5 Al,Ti
```



```
[etot.input]
1 4
IN.ATOM = atom.config
JOB = RELAX
RELAX_DETAIL = 1 500 0.01 1 0.01
Ecut = 70
Ecut2 = 280
DOSTATIC
KPPRA =  1000
```

For this pwmat.wrap file, most systems are applicable.

Run this example:

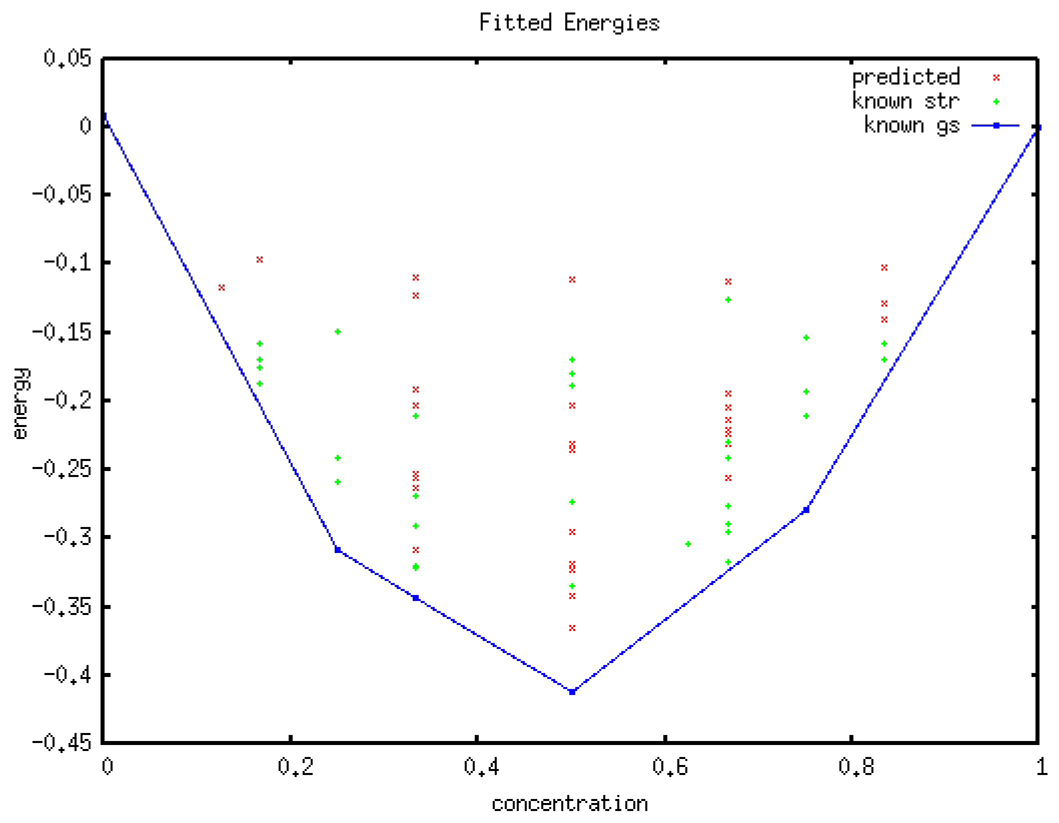
$ maps -d &

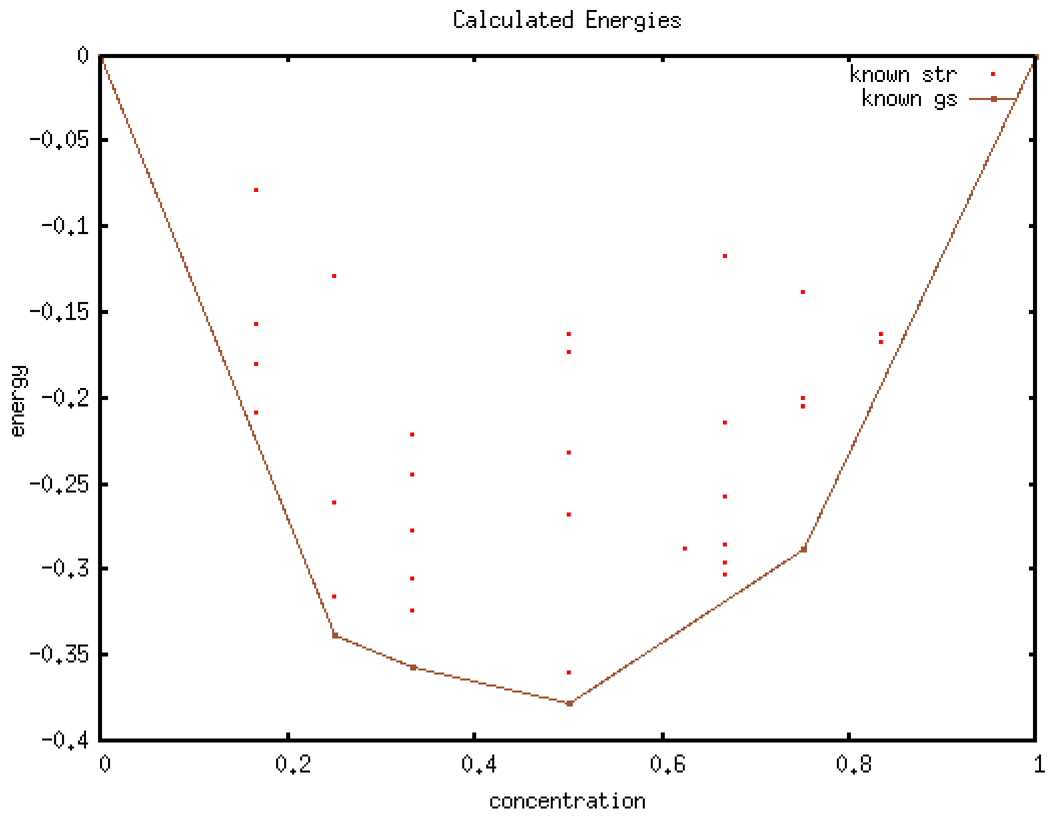
$ pollmach runstruct_pwmat &

Check the results:

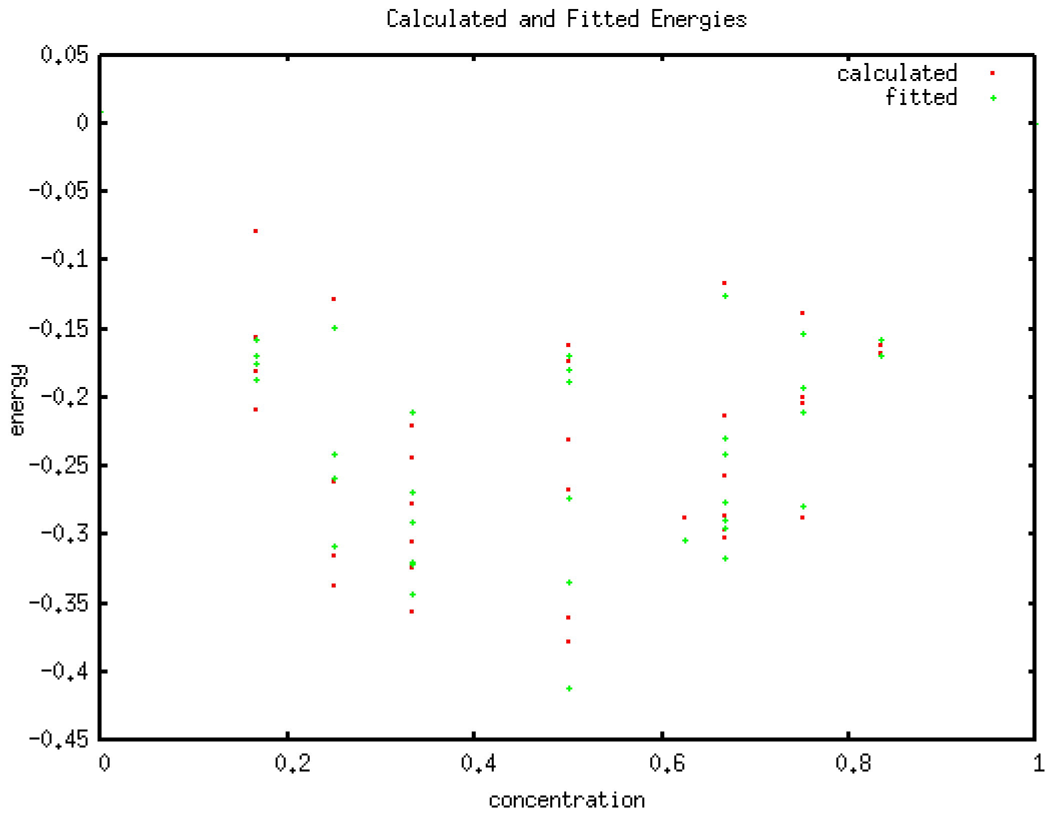
$ mapsrep (the output of mapsrep, please refer to ATAT manual)

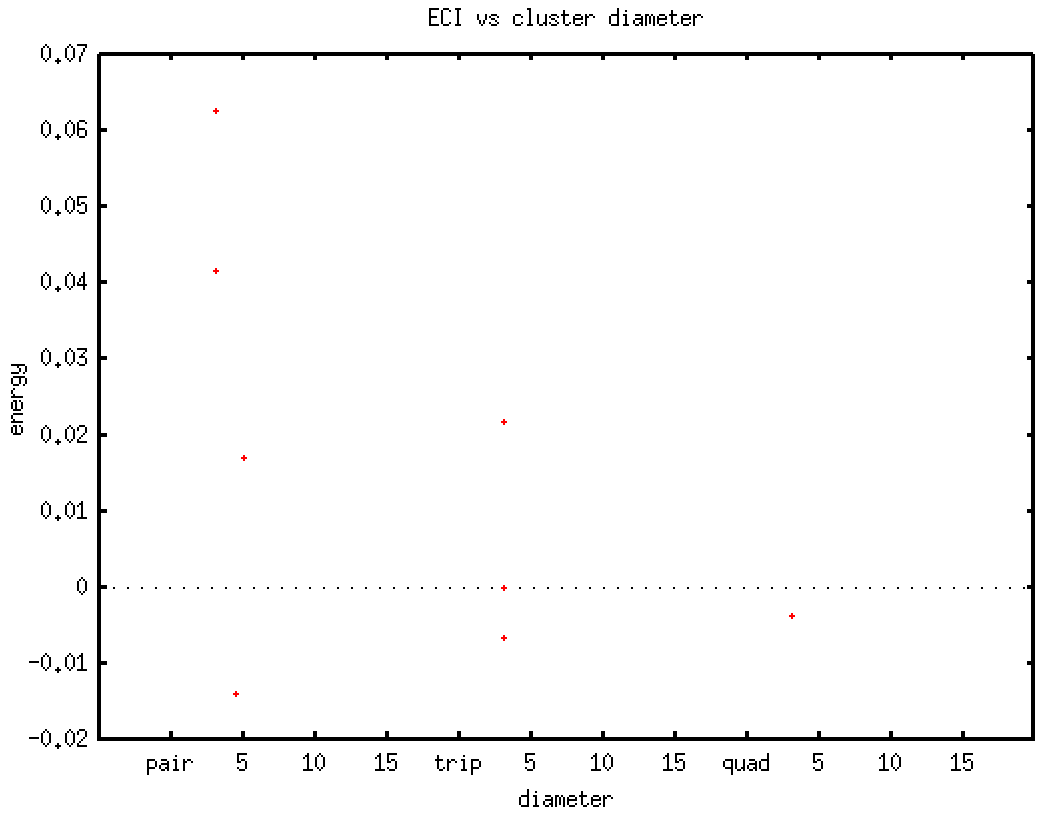
Cluster expansion construction using the MAPS and PWmat



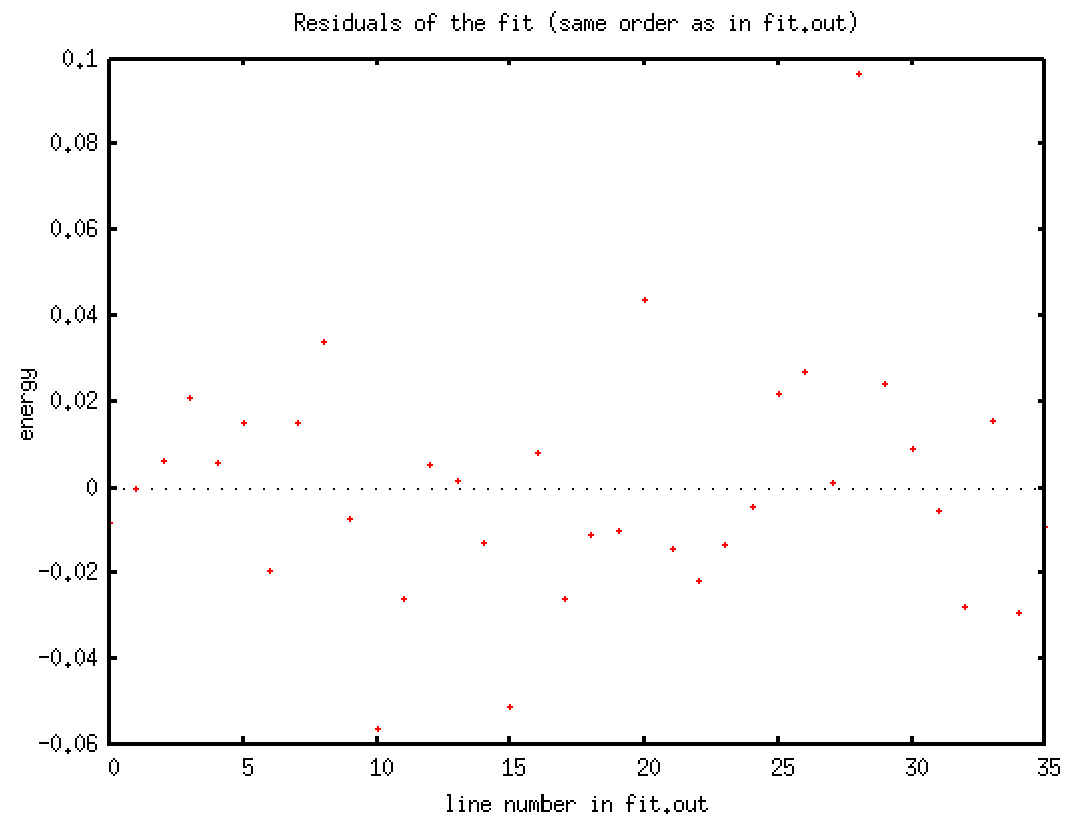


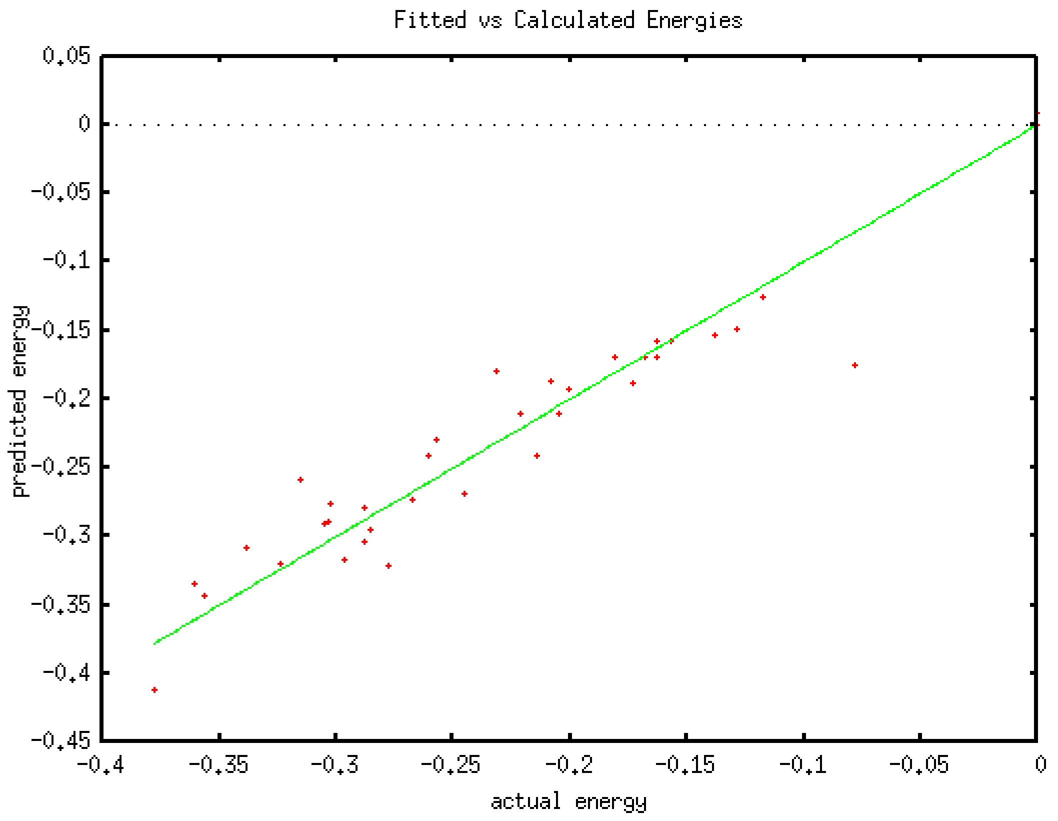
Cluster expansion construction using the MAPS and PWmat





Cluster expansion construction using the MAPS and PWmat





Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

mcsqs uses a Monte Carlo algorithm to find a Special Quasirandom Structure (SQS).

1)This code requires a input files:

A file defining the random state (by default rndstr.in) in a format similar to the lat.in that is needed for maps or corrdump but with partial occupation of the sites (see below).

-> Format of the input file defining the lattice (specified by the –l option)

First, the coordinate system a,b,c is specified, either as

[a] [b] [c] [alpha] [beta] [gamma]

or as

[ax] [ay] [az]

[bx] [by] [bz]

[cx] [cy] [cz]

Then the lattice vectors u,v,w are listed, expressed in the coordinate system just defined:

Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

[ua] [ub] [uc]

[va] [vb] [vc]

[wa] [wb] [wc]

Finally, atom positions and types are given, expressed in the same coordinate system

as the lattice vectors:

[atom1a] [atom1b] [atom1c] [atomtype11]=[occupation11],[atomtype12]=[occupatioin12], …

[atom2a] [atom2b] [atom2c] [atomtype21]=[occupation21],[atomtype22]=[occupatioin22], …

etc.

Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

The fcc lattice of the Cu-Au system; request for an SQS at composition 0.5:

```
3.8 3.8 3.8 90 90 90
0   0.5 0.5
0.5 0   0.5
0.5 0.5 0
0 0 0 Cu=0.5,Au=0.5
```

A lattice for the Li_x Co_y Al_(1-y) O_2 system; request for an SQS with different compostion on each sublattice:

```
0.707 0.707 6.928 90 90 120
0.3333 0.6667 0.3333
-0.6667 -0.3333 0.3333
0.3333 0.3333 0.3333
0          0         0          Li=0.75,Vac=0.25
0.3333  0.6667  0.0833 O
0.6667  0.3333 0.1667  Co=0.25,Ni=0.25,Al=0.5
0          0         0           O
```

Please note that sites that are equivalent by symmetry must have the same occupation. The file rndstrgrp.out can help you figure out which sites are equivalent. If you want override this, you can simply use slightly different species labels (e.g. Fe_a,Fe_b) on the sites whose occupation must differ, even though these sites are a priori equivalent. Using different labels forces the code to consider the sites as inequivalent.

Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

mcsqs uses a Monte Carlo algorithm to find a Special Quasirandom Structure (SQS).

2) A cluster file (by default clusters.out), generated, for instance, with the command line:

mcsqs -2=… -3=… etc.

Where -2=… -3=… indicate the range of pairs, triplets etc.

-> output files:

Clusters.out, sym.out

3) Generate SQS, one typically needs to specify the –n=[number of atoms per cell] parameter on the command line:

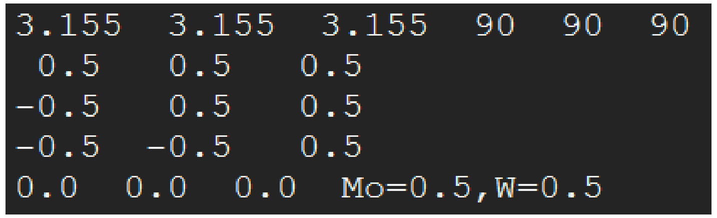
mcsqs –n=[number of atoms per cell]

Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

Example

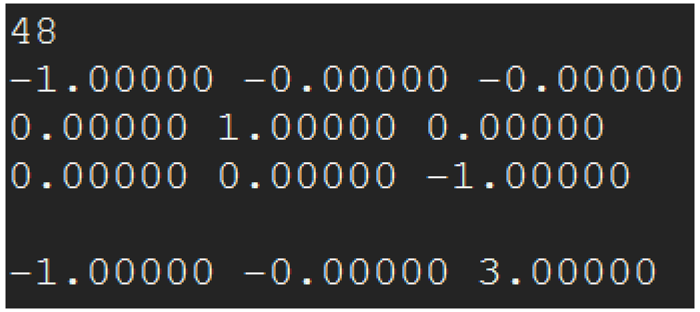
The lattice of the Mo-W system, request for an SQS at composition 0.5:

1).prepare the input file rndstr.in



```
3.155  3.155  3.155  90  90  90
 0.5   0.5   0.5
-0.5   0.5   0.5
-0.5  -0.5   0.5
0.0  0.0  0.0  Mo=0.5,W=0.5
```

rndstr.in



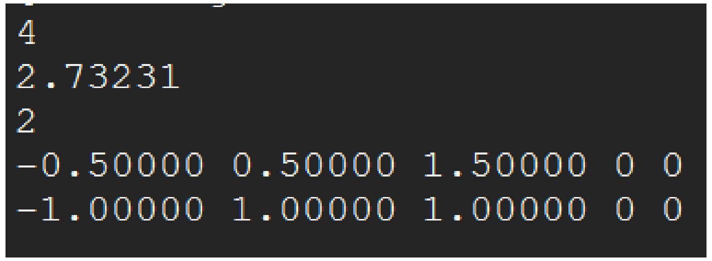
```
48
-1.00000 -0.00000 -0.00000
0.00000 1.00000 0.00000
0.00000 0.00000 -1.00000

-1.00000 -0.00000 3.00000
```

2).generate cluster file (clusters.out, sym.out)

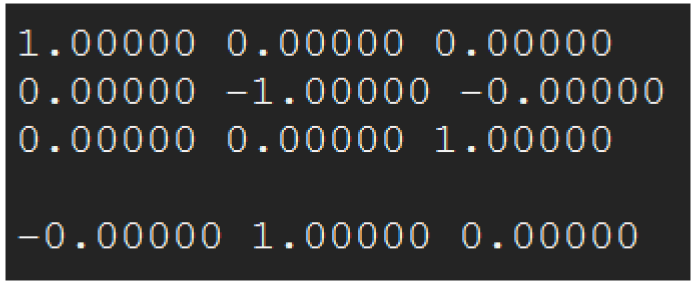
mcsqs –n=3.0 -3=2.4 -4=2.1

…



```
4
2.73231
2
-0.50000 0.50000 1.50000 0 0
-1.00000 1.00000 1.00000 0 0
```

clusters.out



```
1.00000 0.00000 0.00000
0.00000 -1.00000 -0.00000
0.00000 0.00000 1.00000

-0.00000 1.00000 0.00000
```

sym.out

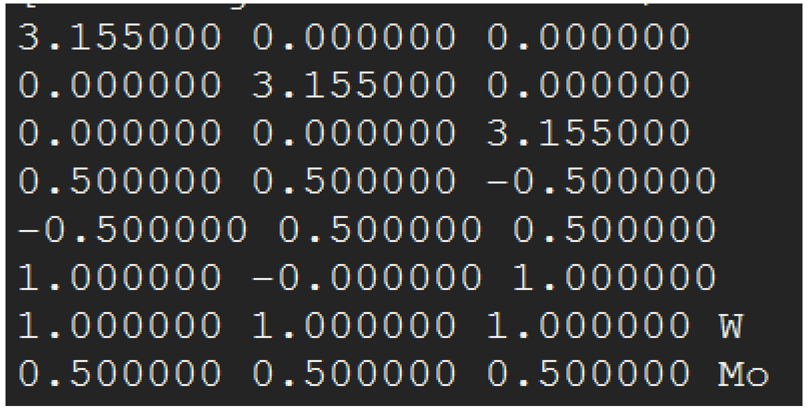
Monte Carlo algorithm to find a Special Quasirandom Structure: mcsqs

Example

The lattice of the Mo-W system, request for an SQS at composition 0.5:

3).generate SQS

mcsqs –n=2



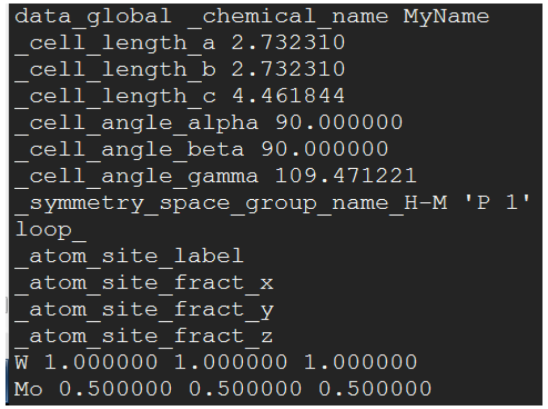
```
3.155000 0.000000 0.000000
0.000000 3.155000 0.000000
0.000000 0.000000 3.155000
0.500000 0.500000 -0.500000
-0.500000 0.500000 0.500000
1.000000 -0.000000 1.000000
1.000000 1.000000 1.000000 W
0.500000 0.500000 0.500000 Mo
```

bestsqs.out

4).converts an ATAT structure file to the CIF format

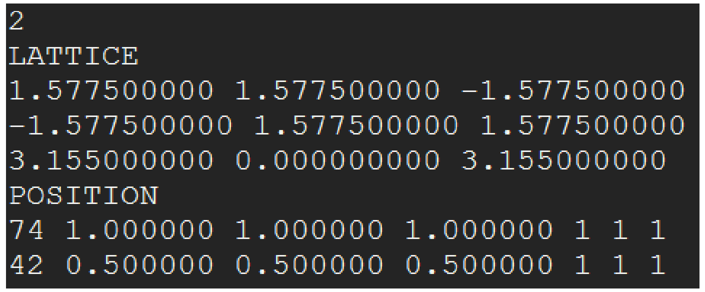
or PWmat format atom.config

str2cif < bestsqs.out > str.cif



```
data_global _chemical_name MyName
_cell_length_a 2.732310
_cell_length_b 2.732310
_cell_length_c 4.461844
_cell_angle_alpha 90.000000
_cell_angle_beta 90.000000
_cell_angle_gamma 109.471221
_symmetry_space_group_name_H-M 'P 1'
loop_
_atom_site_label
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
W 1.000000 1.000000 1.000000
Mo 0.500000 0.500000 0.500000
```

str2config bestsqs.out > atom.config



```
2
LATTICE
1.577500000 1.577500000 -1.577500000
-1.577500000 1.577500000 1.577500000
3.155000000 0.000000000 3.155000000
POSITION
74 1.000000 1.000000 1.000000 1 1 1
42 0.500000 0.500000 0.500000 1 1 1
```