supercell

—

A Scientific Tool

Kirill Okhotnikov, Sylvian Cadars

May 22, 2021
Version 2.0

Abstract

supercell is a tool to convert a crystallographic structure with partial occupancy and/or vacancies to ordinary supercell structure suitable for calculations.

1 Synopsis

supercell **-h, --help**

supercell [**OPTIONS**] **-i** *input-file*

supercell [**OPTIONS**] **--input=***input-file*

2 Description

Disordered compounds are very important now for fundamental science and industrial applications, but most of the methods of exploring solid state material properties require ideal periodicity. In opposition on many reported research, which imply periodicity to disordered system "by hand" and ad-hoc the program *supercell* allows to apply the approximation systematically with all-in-one algorithms implemented for structure manipulation, supercell generation, permutations of atoms and vacancies, charge-balancing, detecting symmetry equivalent structures, Coulomb energy calculations and sampling output data. The advantages of the program are fast-algorithms implementation and availability under GPL license.

3 Options

[-h], [--help] Print help message and exit the program.

[-v *level*], [--verbose=*level*] Change verbosity level from default 1. Level 0 - quiet mode, only error output. Level 1 is suggested for regular users. Higher levels are suggested for developers, bug tracking and long program execution times.

-i *input-file*, --input=*input-file* Required option. Input file in CIF format. Only one file can be specified.

[-d], [--dry-run] The option is highly recommended for the first run of the program with new input specified. With the option the program will do everything but write the files. Be careful if you switched **-m**, **--merge-symmetric** option: the program will go through all crystal structures, so even the dry-run mode can take time.

[-s *cell-size*], [--cell-size=*cell-size*] The option specifies the size of the supercell. The format is AxBxC, where A, B, C are positive integer multipliers of **a**, **b** and **c** unit cell vectors of input system. Default is "1x1x1".

[-c *balance-type*], [--charge-balance=*balance-type*] The option helps to balance charges of the structure. Be careful, charge balancing with wrong input charges will make output system composition far from desired or it can even freeze the program. Charges (i.e. oxydation states) may be specified with the **-p**, **--labels-properties** option (see below) or directly in the CIF file ("_atom_type_oxidation_number"). The possible arguments are:

- no — No charge balancing. The option will set all charges to zero.
- try — Default. Try to charge balance system, if initial system is not charged.
- yes — Charge balance the system.

[-p *labels-properties*], [--property=*labels-properties*] The option will allow you to manually specify some properties of atoms with specific labels. You may use the simple or the extended syntax:

- Simple: <label>:<property_name>[=<property_value>].
- Extended: "<OPT>(<labels>):<properties>" (Do not forget to put extended syntax in quotes).

where <label> is the label of the crystallographic site, the properties of which you want to change. <labels> is a set of space-separated labels. <OPT> is the type of string processing which is used to set the labels. The possible values of <OPT> are:

- p — Treat the labels in parenthesis like a plain string.
- w, "" — Default for simple syntax. Treat the labels in parenthesis like a wild card. For example, N* means all labels, starting with N: N1, N12, but also Na3. You can use "*" (any numbers of any symbol) and "?" (any symbol: exactly one).
- r — Treat the labels in parenthesis like a Perl Regex. Syntax description can be found at `http://en.wikipedia.org/wiki/Regular_expression`.

The <property_name> is the name of the property you want to set for all specified crystallographic sites. Some properties can have values, which should be set by using an equal symbol. The properties can be:

- c[harge] — Set the charge of atoms with specified label(s). Floating-point value in elementary charge units.
- p[opulation] — Number of atoms with specific label(s) in output supercell structure.
- [not]fixed — Exclude crystallographic sites from the combinatorial analysis. The output supercell file will contain partial occupancies for the corresponding sites (if there were partial occupancies in the original structures).

Some fancy examples can be found below.

[-t *tolerance*], [--tolerance=*tolerance*] The argument of the option specifies the maximum distance (in Angstroms) between sites that should be contained within the same group (meaning that the corresponding positions in the supercell cannot be occupied simultaneously). Check carefully the minimal distance between 2 atoms assigned to different groups, which is written in the output of *supercell* before changing this parameter. Default *tolerance*=0.75 A.

[-m], [--merge-symmetric] The option enables the symmetry check algorithm on output structures. Structures that can be transformed to each other using crystallographic symmetry operations will be merged and stored as one structure. For cases with more than 10^4 total combinations it is recommended to use verbosity level 2 or higher to trace program execution.

[-n *selection*], [--store-structures=*selection*] The option allows you to generate subsets of structures (rather than all) according to certain criteria (listed below). The option is very useful to both save disk space and facilitate navigation among files for large numbers combinations (typically more than 10^5).The argument of the option has special structure <sampling type><count>. The <count> is a positive integer value or zero. <sampling type> is a letter which sets the sampling (selection) mode:

r : select <count> random structures.

f : select <count> first structures.

a : select <count> last structures.

l : select <count> low energy structures (**--coulomb-energy** option required).

h : select <count> high energy structures (**--coulomb-energy** option required).

Example: l100 will store first 100 structures with low Coulomb energy.

Another extra option available to find symmetric structures:

w : selects all structures with multiplicity (weight) less or equal <multiplicity>.

Example: w32 will store all structures with multiplicity, for example 32, 24, 16, 8, 1, but skip structures with multiplicity 48.

Multiple declaration of the option is allowed. If the option is enabled, an extra prefix (the letter <sampling type> will be added to output file name. Be careful, enabling the option requires extra memory, proportionally to the number of structures to store. This is specially important for "w" option.

[-q], [--coulomb-energy] The option enable Coulomb energy calculations. The result energies for each structure are stored in file <output-prefix>_coulomb_energy.txt. The energies can help to consider the relevant regular structures for following calculations. Ewald summation algorithm is used for the calculations. Be careful, in case of charged cell the energy values can be meanless. See also **--charge-balance** option.

[-g], [--coulomb-store] Use the option to create a file with electrostatic energy for all processed structures.

[-o *output-prefix*], [--output-prefix=*output-prefix*] The options specify output file name prefix. The prefix can contain folder name but the folder should be created before run the program. For example, --output-prefix=myfolder/myfiles. The output files will be created according to templates. For non merging run the template will be <output-prefix>_i<sampling type><index>.cif. For merging run the template will be <output-prefix>_i<sampling type><index>_w<weight>.cif, where <weight> - number of the structures merged to the structure. Be careful, all existing files with mask "<output-prefix>*.cif" will be deleted during not dry-run. The <output-prefix>_coulomb_energy.txt file will be overwritten only in case of **--coulomb-energy** option enabled.

[--random-seed=*number*] The option specifies a random seed for structures random sampling.

[-a *archive-file*], [--archive=*archive-file*] The option specifies a target archive file name for the output files, except energy file for all permuted configurations. If archive-file string is empty (default) no packing will be performed and all files will be stored directly to disk. If the file with the already name exists it will be overwritten. This optional feature requires *libarchive* `http://www.libarchive.org/` to be available in the system. Otherwise the option will be disabled. Check the status by **-h** option. The extension of the file should be set according to desired archive type: ".zip", ".tar", ".tgz", ".tar.gz", ".tar.bz2", ".tar.xz".

4 Files

5 Examples

The examples are based on file `supercell/data/examples/Ca2Al2SiO7`.

- Dry run. Obtain information about the structure: group assignment, cell size, etc. It is a good practice to start a new structure processing with this command. This run doesn't produce any output files.

  supercell -d -i Ca2Al2SiO7.cif

- Dry run. Obtain information about the structure.

  supercell -d -i Ca2Al2SiO7.cif -s 1x1x2

- Dry run. Obtain information about the structure.

  supercell -d -i Ca2Al2SiO7.cif -s 1x1x2

Some advances examples, with *supercell* embedded to bash scripts you can find in `supercell/data/examples` folder.

6 Requirements

Compiled version of *supercell* from site `https://orex.github.io/supercell` is standard alone program for both Linux and MacOS systems, which requires neither installation nor third-party libraries to install. Please check `supercell/INSTALL` file, if you would like to compile program by yourself.

7 Bugs and limitations

Supercell size limitation. The maximum number of permutations should not exceed limit of 10^{16}. If more, the program will return an error. The value is far beyond a reasonable limit due to calculation time. The average program performance is about 10^{12}–10^{13} structures per day with average workstation.

Filesystem limitation. Although, *supercell* can produce millions of files, most of the filesystems can't manage more than one thousand files in one folder with acceptable performance. Be careful, if you need to process a lot of combinations, sample them or use **-a** option.

Symmetry information handling in input file. Be careful with low symmetric structures. For example, in monoclinic cells structures often have an inconsistency beetwen symmetry operations atoms positions and unit cell configuration. Supercell program does the best to solve the problems, but it is always a good idea to verify CIF file with external tools like *enCIFer*. The problem can be fully solved by converting input file to P1 structure (see non-diagonal supercell expansion).

Non-diagonal supercell expansion. The program has it's own general algorithm for symmetry search. Input structures are converted internally to P1 structures and builtin algorithms search for symmetry in P1 structures. It gives a huge freedom for user, manipulating the input structure. For example, although, *supercell* cannot generate non-diagonal supercells directly, it can use such cells, generated with other software. You need to generate any cell and store it as P1 cell. *Supercell* is fully compatible with *VESTA* (highly recommended to use) output CIF files, but it is not compatible with *Material studio* output for example.

8 Version

Version: 2.0 of May 22, 2021.

9 Version history

Version 2.0 (20-05-2021) A new major release of the program. A performance increase of almost all algorithms: enumerating(x4-x10), electrostatic calculation (x2-x5), symmetry search (x2). Shared memory multithreading in during enumerating of structures. Better support of CIF input files. New command line options: **-n** *w*, **--random-seed**, (see above). OpenBabel dependency removed. The code is upgraded to C++14 standard.

Version 1.2 (23-05-2019) Four times performance increase.

Version 1.1 (24-01-2019) The version of supercell program can process up to 10^{16} total structures. The value is far beyond a reasonable limit due to calculation time. The average program performance is about 10-100 billion structures per day with standard desktop processor.

Version 1.0 (31-03-2016) Initial release.

10 License and Copyright

Misc The actual version of *supercell* may be found on my homepage `https://orex.github.io/supercell`. Please, use github site `https://github.com/orex/supercell` to download the source code and submit a bug of the program.

11 Author

Kirill Okhotnikov e-mail: `kirill.okhotnikov@gmail.com`

Dr. Sylvian Cadars e-mail: `sylvian.cadars@cnrs-imn.fr`
Institut des Materiaux Jean Rouxel (IMN) - UMR6502
2 rue de la Houssiniere, BP32229
44322 Nantes cdx3, France
Tel: +33 (0)2 40 37 39 34
Fax: +33 (0)2 40 37 39 95