Nanocut - Usermanual

Florian Uekermann & Sebastian Fiedler

March 17, 2010

Contents

1 What is nanocut?

The purpose of nanocut is to provide an easy way to cut certain shapes out of arbitrary crystals.

It is available under the terms of the 3-clause license ("New BSD License").

2 Basics

2.1 A very basic setup

Every information on the material and structures you want to cut out must be stored in an INI file. The listing below shows an example of a very basic setup defining a sphere made up of Natriumchloride:

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0

basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[sphere: somename]
radius_vector = 2 0 0
```

Every input file producing any output consists at least of two sections. The geometry section, containing everything needed to specify the crystals structure, and at least one body section (opened by `[sphere: somename]` in this case), defining the body to be cut out.

2.2 Usage

The simplest valid command is `nanocut INFILE`. INFILE must be a valid INI file like the one shown in the example above. The result will be written to the standard output by default. The output format is XYZ. Options can be added to the commandline at arbitrary positions. Possible Options are:

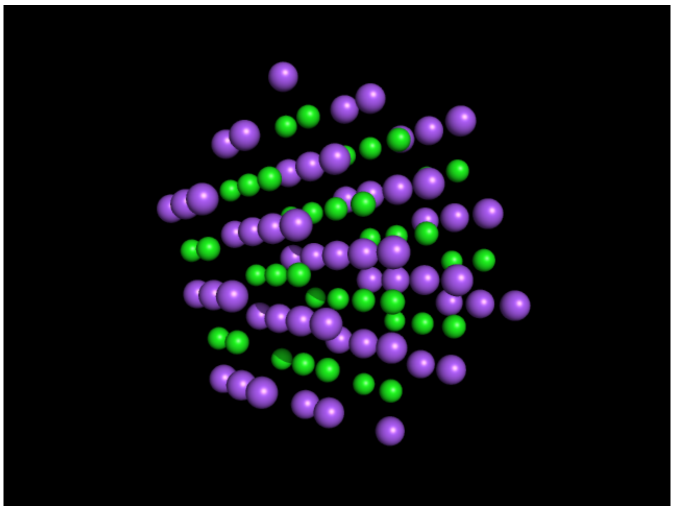
-w FILE This writes the output to the file specified by FILE. In case FILE doesn't exist it will be created, otherwise the existing file will be overwritten without further questions.

-a FILE This merges the result with the content of an existing INI file specified by FILE. FILE must exist.

-h Prints helptext.

–help Prints helptext.

Usually the command looks like this: `nanocut basic.ini -w basic.xyz`. With basic.ini containing the configuration specified in the listing above the output looks like this:



3 Geometry - defining unit cells

The [geometry] section contains the information required to create the crystal structure. Only one geometry section in every INI file is allowed and in case of multiple entries only the last one will be used. It requires always two parameters and only a third one is enabled.

Parameters The [geometry] section requires the two entries lattice_vectors and basis_coordsys and features the possibility to define the basis in either cartesian or lattice vectors.

lattice_vectors unit vectors of a unit cell of the crystal in cartesian coordinates. Of course three and only three entries are allowed which may not be linear dependant upon one another or empty. The vectors are to be given one vector after another.

basis all atoms included in a unit cell. The atoms are each to be noted beginning with their chemical symbol followed by their coordinate in the unit cell.

basis_coordsys determines the coordinate system the vectors in basis are considered to be in. Options are either cartesian for values in cartesian coordinate system or lattice for fractions of lattice vectors.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0
basis: Na 0   0   0
       Cl 0.5 0.5 0.5
```

```

[sphere: round]
radius_vector: 0 0 5.55
```

4 Periodicity - defining periodic structures

Structures with periodicities in one or two dimensions require an additional section defining the type of periodicity and the axis or axes alongside which the supercell is periodic. It is opened by `[periodicity]` and must contain the following parameters:

Parameters

`period_type` Defines the number of directions in which the structure is periodic. Possible Values are `1D`, `2D` and `0D`. Specifying he latter is equal to leaving out the whole section.

`axis` Defines the axis alongside which the supercell is periodic in lattice coordinates. The number of elements supplied must be 3 times the number in `period_type`. This parameter will not be evaluated if `period_type` is `0D`. Non-integer parts of the numbers given are disregarded

Bodies not matching the type of periodicity specified in `period_type` are ignored.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0
basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[periodicity]
period_type= 1D
axis= 1 1 1

[periodic_1D_cylinder: somename]
radius=5
```

5 Body - defining Bodies

An INI file can contain an arbitrary number of sections defining bodies. Each body section is opened by `[BODY: NAME]` where `BODY:` defines the bodies type and `NAME` is an unique name to tell different bodies with equal types apart. The `order` and the shift_vector parameters are supported by all bodies but not mandatory. They will be explained in the Advanced section.

Important: All vectors inside a body section are specified in lattice coordinates by default. For every vector specified in cartesian coordinates the additional parameter someparameter_coordsys must be added. Its value is either lattice or cartesian.

5.1 Non-periodic bodies

5.1.1 Sphere

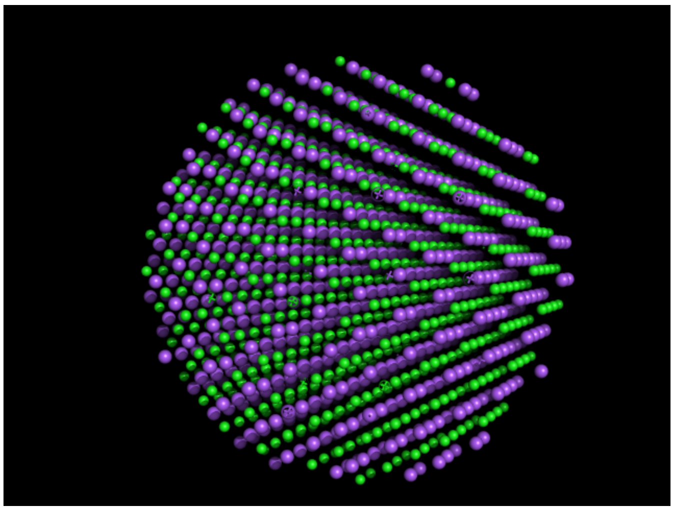
The simplest body one can specify is a sphere. The body's section is opened by [sphere: NAME]. Its size is determined by a vector pointing from the center of the sphere (at the origin of the coordinate system) to an arbitrary point on its surface.

Parameters

radius_vector Vector whose length specifies the radius of the sphere.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0

basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[sphere: mysphere]
radius_vector = 5 0 0
```



5.1.2 Convex polyhedron

`[convex_polyhedron]` enables cutting of bodies surrounded by an arbitrary amount of planes. The planes might be determined in different ways. Also the combination of planes defined differently is possible but only one kind is required. If the input is insufficient the program will exit if no proper body can be calculated or calculate the shape it can derive from the given INI. The body's section is opened by `[convex_polyhedron: NAME]`.

Parameters `[convex_polyhedron]` requires at least one item of the following. Each item may contain an arbitrary amount of planes each determined by four integer or float values.

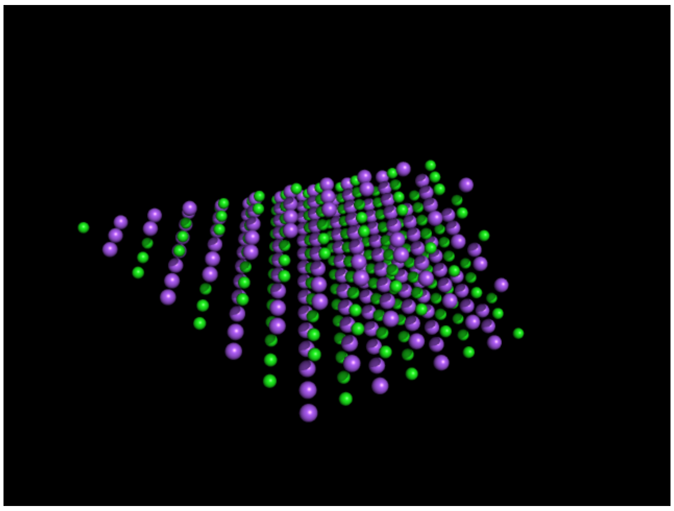
`planes_miller` planes defined by miller indizes. Each plane is given by 4 integers or floats with the first three being the miller indices and the last one the plane's minimum distance from the origin (the length of the smallest possible cartesian vector between the origin and the plane)

`planes_normal` planes defined by a vector orthogonal to the plane and its minimum distance from the origin as in `planes_miller`. The vector does not need to be normalized.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0
basis: Na 0   0   0
       Cl 0.5 0.5 0.5
```

```

[convex_polyhedron: example]

planes_normal: 0   0 1 0
               1   0 1 30
              -1   0 1 30
               0   1 1 30
               0  -1 1 30
```



5.1.3 Cylinder

In this context a cylinder is a body with circular base and top areas which are orthogonal to the difference vector of their centers. The edges of base and top area are connected by the smallest lateral area possible. The body's section is opened by `[cylinder: NAME]`.

The name "cylinder" is a bit misleading, since cylinders and truncated cones are contrivable.

Parameters

`point_1` Position vector to the center of the first circular area.

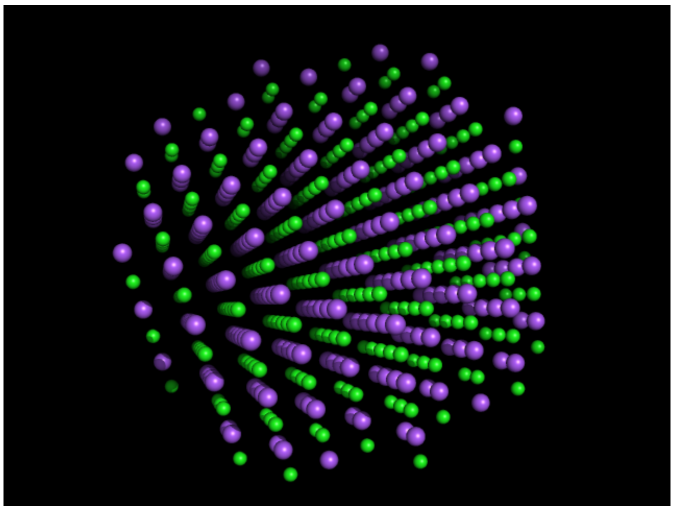
`radius_1` Radius of the first circular area.

`point_2` Position vector to the center of the second circular area.

`radius_2` Radius of the second circular area.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0
basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[cylinder: mycylinder]
point_1: 0 0 -1
point_1_coordsys: cartesian

point_2: 0 0 10
point_2_coordsys: cartesian

radius_1: 10
radius_2: 10
```



5.2 1D-periodic bodies

Apart from the periodicity section being a mandatory part of the configuration there is no fundamental difference between creating 1D-periodic structures and non-periodic ones. The resulting supercell is rotated around the origin so that the axis vector defined in the periodicity section is parallel to the z-Axis.

5.2.1 Periodic Cylinder

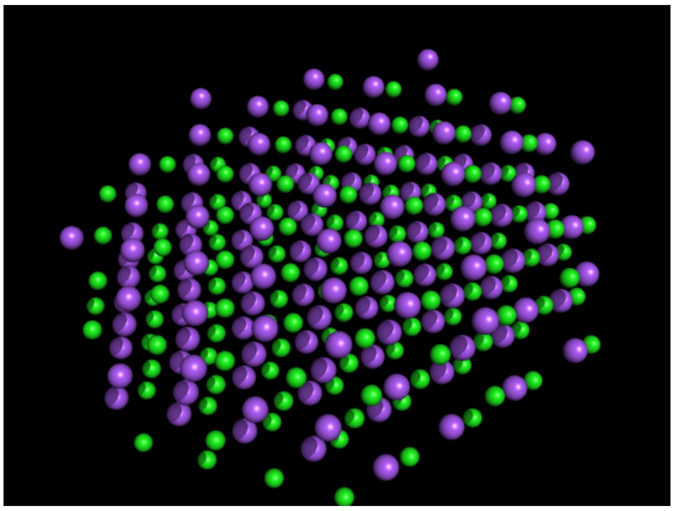
The periodic cylinder is the supercell of an infinitely long cylinder with a circular base area. The base area's center is the origin and its normal vector is parallel to the axis specified in the periodicity section. The body's section is opened by `[periodic_1D_cylinder: NAME]`.

Parameters

`radius` The cylinder's radius.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0

basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[periodicity]
period_type= 1D
axis= 1 1 1

[periodic_1D_cylinder: asdasd]
radius=10
```



5.2.2 Prism shaped wires

`[periodic_1D_convex_prism: NAME]` enables cutting a convex prism which can be appended unto itself to create an infinte wire from the given crystal structure. The prism is determined by its lateral planes. These planes can be determined in different ways and the combination of planes defined differently is possible. Please note the planes should not cross with the axis but be parallel to it or they will be projected accordingly changing the created body. Of course at least three planes are needed to create a proper body. If no prism can be calculated from the given planes the programm will indicate

so and exit.
The body's section is opened by `[periodic_1D_convex_prism: NAME]`.

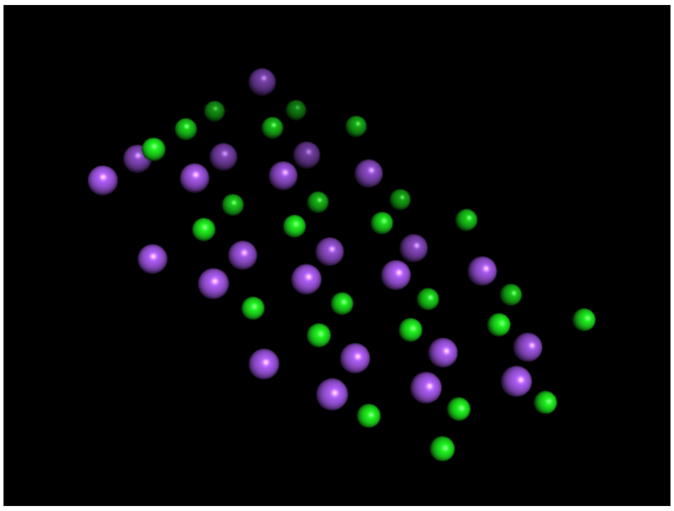
Parameters `[periodic_1D_convex_prism]` requires at least one item of the following. Each item may contain an arbitrary amount of planes each determined by four integer or float values.

`planes_miller` planes defined by miller indizes. Each plane is given by 4 integers or floats with the first three being the miller indices and the last one the plane's minimum distance from the origin (the length of the smallest possible cartesian vector between the origin and the plane)

`planes_normal` planes defined by a vector orthogonal to the plane and its minimum distance from the origin as in `planes_miller`. The vector does not need to be normalized.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0
basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[periodicity]
period_type: 1D
axis: 0 0 1

[periodic_1D_convex_prism: example]

planes_miller:  -1.3 0 0 0
                2 0 0 10

planes_normal:  0 5.1 0 5
                0 -3 0 10
```



5.3 2D-periodic bodies

Apart from the periodicity section being a mandatory part of the configuration there is no fundamental difference between creating 1D-periodic structures and non-periodic ones. The resulting supercell is rotated around the origin so that the z-Axis is orthogonal to both axis vectors defined in the periodicity section.

5.3.1 Periodic plane

The periodic plane is the supercell of a plane infinitely extended in two dimensions. The upper and lower plane limiting the body in the third dimension are equidistant from the origin.

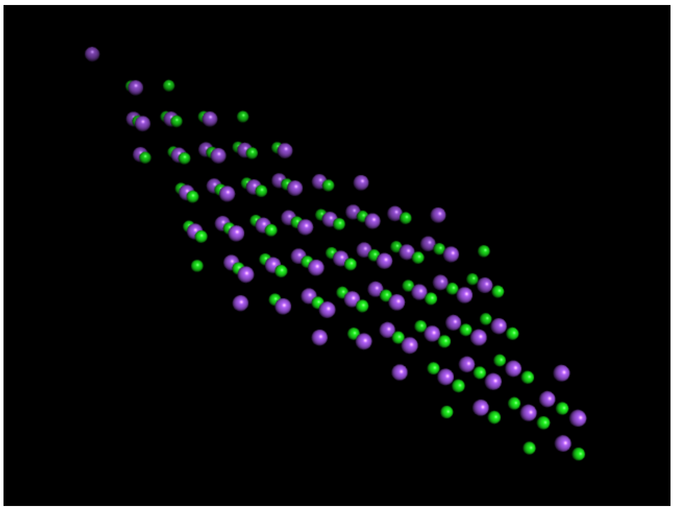
Parameters

`thickness` The planes thickness.

Example

```
[geometry]
lattice_vectors: 0 2 2
                 2 0 2
                 2 2 0

basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[periodicity]
period_type= 2D
```

```
axis: 1 1 5
      5 5 1

[periodic_2D_plane: myplane]

thickness=5
```



6 Advanced Usage

6.1 The order parameter - adding and substracting bodies

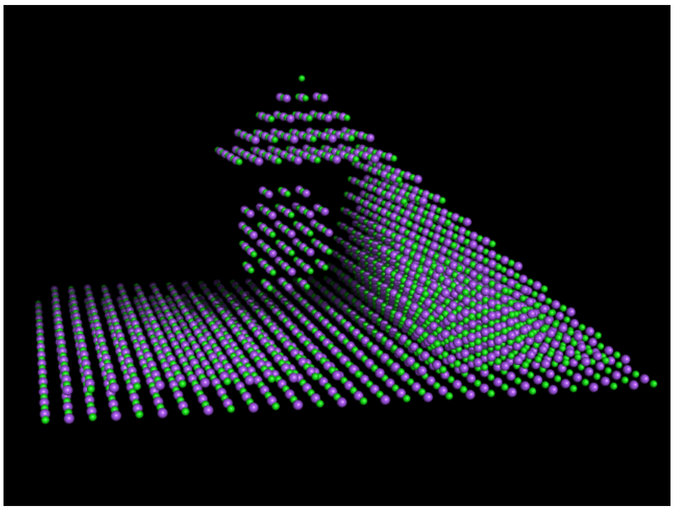
Additional to simply creating bodies nanocut features the possibility to substract and add bodies. The `order` parameter is an integer which can be specified for every body. It determines the order in which the bodies are substracted or added. Beginnig with the bodies with the smallest order greater than zero the bodies are processed in ascending order. Therefore bodies with orders beiing lower than one are ignored. An uneven order causes the body to be added. Those with uneven orders are substracted. The default order is 1.

Multiple bodies having the same order is allowed and common practice since there is no difference in result for processing a set of bodies in any order as long as they are all substracted or added.

Example

```
[geometry]

lattice_vectors: 0     2 2
                 2 0     2
                 2     2    0
```

```
basis: Na 0   0   0
       Cl 0.5 0.5 0.5

[convex_polyhedron: pyramid]

order: 1

planes_normal: 0  0 1 -15
               1  0 1 15
              -1  0 1 15
               0  1 1 15
               0 -1 1 15
planes_normal_coordsys: cartesian

[convex_polyhedron: substrac_cuboid]

order: 2

planes_normal: 1  0 0 -30
               1  0 0  30
               0  1 0 -7
               0  1 0  30
               0  0 1 -11
               0  0 1  5
planes_normal_coordsys: cartesian

[sphere: added_sphere]

order:3

radius_vector: 6 0 0
radius_vector_coordsys: cartesian
shift_vector: 0 2 -3
shift_vector_coordsys: cartesian
```



6.2 Shift vectors

The `shift_vector` parameter enables the possibility to shift a body to a certain position. The value for `shift_vector` is the vector defining the translation.

This is particularly usefull when using the order parameter to create specific shapes.

6.2.1 Periodic bodies

Using a shift vector with periodic can lead to unexpected results at first glance, for two reasons.

1. Every atom is moved into the first supercell after being cut out. This undoes the effect of the shift vector's components in the direction(s) of the axis or both axes.

2. The automatic rotation of periodic bodies is applied last.

In combination this means:

A body's translation in 1D-periodic structures is visible in the result as a translation inside the x-y-plane which is lacking the component in direction of axis.

In 2D-periodic structures the translation is visible in z-direction by the shift vector's component orthogonal to both axes.