

# Nanocut - Usermanual

Florian Uekermann & Sebastian Fiedler

March 31, 2010

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>What is Nanocut?</b>                                       | <b>1</b>  |
| <b>2</b> | <b>Basics</b>   | <b>2</b>  |
| 2.1      | A very basic setup . . . . .                                  | 2         |
| 2.2      | Usage . . . . .   | 2         |
| <b>3</b> | <b>Geometry - defining unit cells</b>                         | <b>3</b>  |
| <b>4</b> | <b>Periodicity - defining periodic structures</b>             | <b>4</b>  |
| <b>5</b> | <b>Body - defining Bodies</b>                                 | <b>4</b>  |
| 5.1      | Non-periodic bodies . . . . .                                 | 5         |
| 5.1.1    | Sphere . . . . .  | 5         |
| 5.1.2    | Convex polyhedron . . . . .                                   | 5         |
| 5.1.3    | Cylinder . . . . .  | 6         |
| 5.2      | 1D-periodic bodies . . . . .                                  | 8         |
| 5.2.1    | Periodic cylinder . . . . .                                   | 8         |
| 5.2.2    | Periodic convex prism . . . . .                               | 9         |
| 5.3      | 2D-periodic bodies . . . . .                                  | 10        |
| 5.3.1    | Periodic plane . . . . .                                      | 10        |
| <b>6</b> | <b>Advanced Usage</b>   | <b>11</b> |
| 6.1      | The order parameter - adding and subtracting bodies . . . . . | 11        |
| 6.2      | Shift vectors . . . . .                                       | 13        |
| 6.2.1    | Periodic bodies . . . . .                                     | 13        |

## 1 What is Nanocut?

The purpose of nanocut is to provide an easy way to cut certain shapes out of periodic structures.

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

## 2 Basics

### 2.1 A very basic setup

Every information on the material and structures to be 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:

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                 2 0 2
4                 2 2 0
5
6 basis: Ni 0 0 0
7        Tl 0.5 0.5 0.5
8
9 [sphere: somename]
10 radius_vector = 2 0 0
```

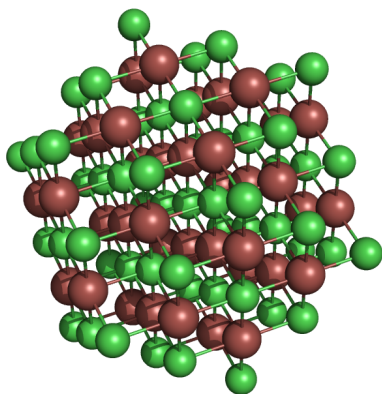
Every input file producing any output consists of at least two sections. The geometry section, containing everything needed to specify the crystal's structure, and at least one body section (opened by `[sphere: somename]` in the example above case), defining the body to be cut out.

### 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** Writes the output into 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 all information regarding crystal structure. It is opened by `[geometry]`.

#### Parameters

`lattice_vectors` Defines the three lattice vectors of the crystal structure in cartesian coordinates.

`basis` Chemical symbol of each basis atom followed by its basis vector.

`basis_coordsys` Defines the coordinate system of the basis. `cartesian` and `lattice` are the possible values. This parameter is optional, by default the basis is specified in lattice coordinates.

#### Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5 basis: Ni 0 0 0
6        Tl 0.5 0.5 0.5
7
8 [sphere: round]
9 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 the 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 discarded.

Bodies not matching the type of periodicity specified in **period\_type** are ignored.

### Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                 2 0 2
4                 2 2 0
5 basis: Ni 0 0 0
6        Tl 0.5 0.5 0.5
7
8 [periodicity]
9 period_type= 1D
10 axis= 1 1 1
11
12 [periodic_1D_cylinder: somename]
13 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 body's 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 are not mandatory. They will be explained in section 6.

**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

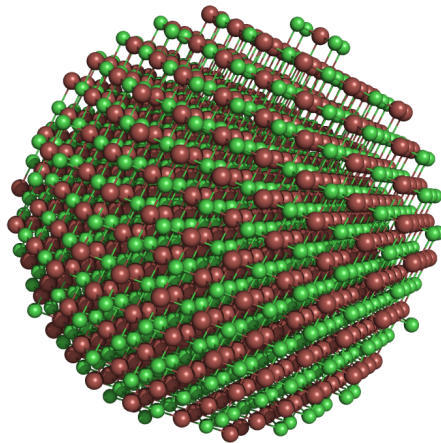
The simplest body one can specify is a sphere. The body's section is opened by `[sphere: NAME]`. Its size is determined by a vector pointing from the sphere's center (at the origin of the coordinate system) to an arbitrary point on its surface.

#### Parameters

`radius_vector` Vector whose length specifies the sphere's radius.

#### Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5
6 basis: Ni 0 0 0
7        Tl 0.5 0.5 0.5
8
9 [sphere: mysphere]
10 radius_vector = 5 0 0
```



### 5.1.2 Convex polyhedron

A convex polyhedron is defined by the planes delimiting the body. A plane can be defined using miller indices or the normal vector. The body's section is opened by `[convex_polyhedron: NAME]`.

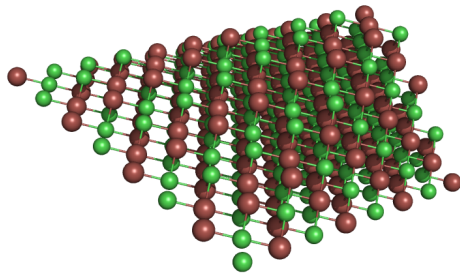
## Parameters

**planes\_miller** Miller indices of each plane (except those defined using normal vectors) followed by its distance from the origin.

**planes\_normal** Vector orthogonal to each plane (except those defined using miller indices) followed by its distance from the origin. The vectors do not need to be normalized.

## Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5 basis: Ni 0 0 0
6         Tl 0.5 0.5 0.5
7
8 [convex_polyhedron: example]
9
10 planes_normal: 0 0 1 0
11               1 0 1 30
12              -1 0 1 30
13               0 1 1 30
14               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.

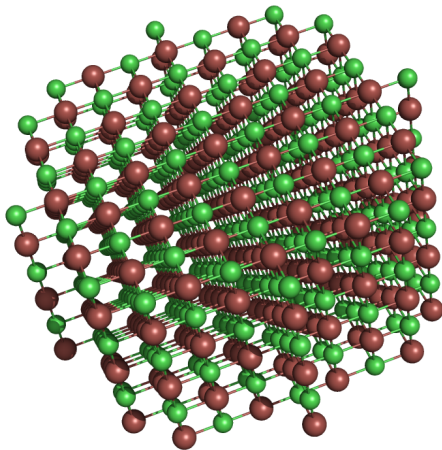
`radius_1` Radius of the first circular area.

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

`radius_2` Radius of the second circular area.

### Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5 basis: Ni 0 0 0
6         Tl 0.5 0.5 0.5
7
8 [cylinder: mycylinder]
9 point_1: 0 0 -1
10 point_1_coordsys: cartesian
11
12 point_2: 0 0 10
13 point_2_coordsys: cartesian
14
15 radius_1: 10
16 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 such 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]`.

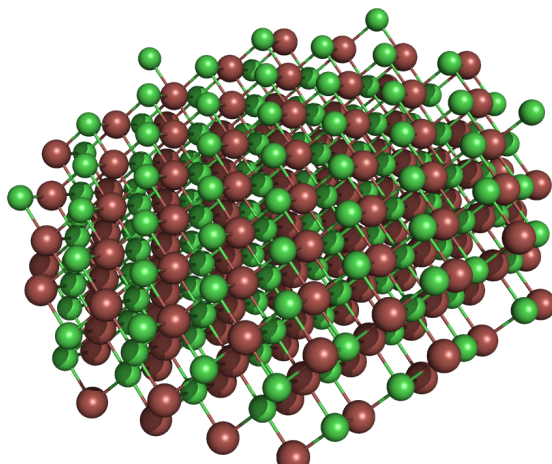
#### Parameters

`radius` The cylinder's radius.

#### Example

```
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5
6 basis: Ni 0 0 0
7        Tl 0.5 0.5 0.5
8
9 [periodicity]
10 period_type= 1D
11 axis= 1 1 1
12
13 [periodic_1D_cylinder: asdasd]
14 radius=10
```





### 5.2.2 Periodic convex prism

The periodic convex prism is an infinitely long cylinder with a convex polygon as base area. The prism is defined by its lateral planes. A plane can be defined using its miller indices or its normal vector. The body's section is opened by `[convex_polyhedron: NAME]`.

If a plane is not parallel to the periodicity axis the plane is rotated until it is parallel. The rotation axis is orthogonal to the normal vector of the plane and the periodicity axis and intersects with the origin.

#### Parameters

`planes_miller` Miller indices of each plane (except those defined using normal vectors) followed by its distance from the origin.

`planes_normal` Vector orthogonal to each plane (except those defined using miller indices) followed by its distance from the origin. The vectors do not need to be normalized.

#### Example

```

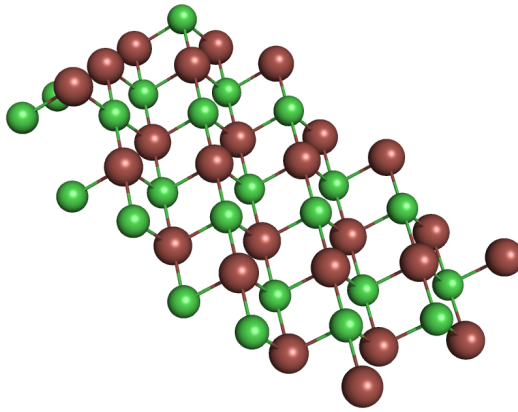
1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5 basis: Ni 0 0 0
6         Tl 0.5 0.5 0.5
7
8 [periodicity]
9 period_type: 1D

```

```

10 axis: 0 0 1
11
12 [periodic_1D_convex_prism: example]
13
14 planes_miller:  -1.3 0 0 0
15                  2 0 0 10
16
17 planes_normal:   0 5.1 0 5
18                  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 2D-periodic structures and 1D-periodic or non-periodic ones. The resulting supercell is rotated around the origin such 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

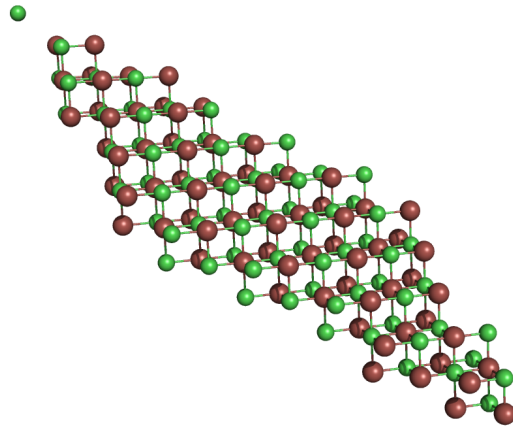
`thickness` The plane's thickness.

##### Example

```

1 [geometry]
2 lattice_vectors: 0 2 2
3                  2 0 2
4                  2 2 0
5
6 basis: Ni 0 0 0
7        Tl 0.5 0.5 0.5
8
9 [periodicity]
10 period_type= 2D
11 axis: 1 1 5
12       5 5 1
13
14 [periodic_2D_plane: myplane]
15
16 thickness=5

```



## 6 Advanced Usage

### 6.1 The order parameter - adding and subtracting bodies

In addition to simply creating structures of the shapes mentioned in the previous section, Nanocut features the possibility to subtract and add those to build more complex shapes.

The `order` parameter is an integer which can be specified for every body. It determines the order in which the bodies are subtracted or added. The bodies are processed in ascending order, beginning with the bodies with the smallest value for `order` greater than zero. Therefore bodies with orders lower than one are ignored. An uneven value

for `order` causes a body to be added. Those with even values for `order` are subtracted. The default value for `order` is 1.

It is allowed and common practice to define multiple bodies with the same value for `order`, since there is no difference in the result for processing a set of bodies in any order as long as they are all subtracted or added.

### Example

```

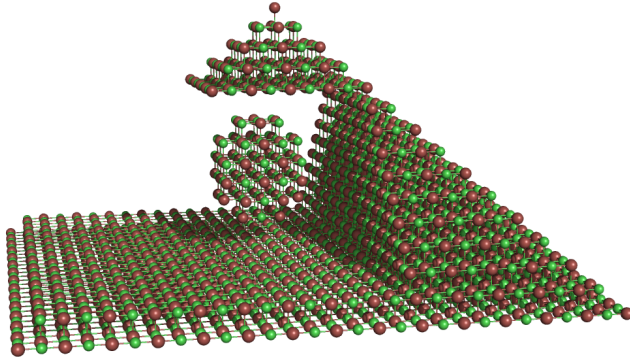
1  [geometry]
2
3  lattice_vectors: 0      2 2
4                  2 0      2
5                  2      2    0
6  basis: Ni 0      0      0
7          Tl 0.5 0.5 0.5
8
9  [convex_polyhedron: pyramid]
10
11 order: 1
12
13 planes_normal: 0  0 1 -15
14                1  0 1 15
15               -1  0 1 15
16                0  1 1 15
17                0 -1 1 15
18 planes_normal_coordsys: cartesian
19
20 [convex_polyhedron: substrac_cuboid]
21
22 order: 2
23
24 planes_normal: 1  0 0 -30
25                1  0 0 30
26                0  1 0 -7
27                0  1 0 30
28                0  0 1 -11
29                0  0 1 5
30 planes_normal_coordsys: cartesian
31
32 [sphere: added_sphere]
33
34 order:3
35
36 radius_vector: 6 0 0

```

```

37 radius_vector_coordsys: cartesian
38 shift_vector: 0 2 -3
39 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 useful when the order parameter is used to create specific shapes.

### 6.2.1 Periodic bodies

Using a shift vector with periodic bodies 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 after the body is shifted.

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 the periodicity axis.

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