PBCTools Plugin User's Guide

Jerome Henin   Olaf Lenz   Cameron Mura   Jan Saam

Version 2.5

The VMD plugin "PBCTools" provides procedures to handle periodic boundary conditions, *i.e.* to set and get the unitcell parameters, to wrap atoms into the central image, to unwrap atoms when they have been wrapped, and to draw the unit cell vectors. All of the procedures are able to handle non-orthorhombic periodic boundary conditions and changes of the unitcell parameters over time (as for example in constant pressure simulations).

Contents

1 Installation

Since VMD version 1.8.6, the PBCTools plugin is part of the official distribution of VMD[1], and all commands can be used within VMD without further preparation.

In the case that you are using an older version of VMD, or that you want to use a more recent version of PBCTools than what came with the VMD distribution, you can activate the PBCTools plugin as follows:

1. Download the PBCTools plugin from its homepage[2] (either as a package, or from Subversion).
2. Unpack the archive to an arbitrary installation directory (*e.g.* `/usr/local/lib/vmd/plugins-local/pbctools/`).
3. Add the following lines to your VMD startup file (`~/.vmdrc` on Unix or `vmd.rc` on Windows)[3]:

```
set dir installation-directory
source $dir/pkgIndex.tcl
package require pbctools
```

2 Basic usage and command summary

All of the plugin's functions can be accessed via the Tcl text command

```
pbc subcommand [options]...
```

that you can write in a VMD-Tcl-script or interactively enter in the VMD console window or the VMD TkConsole (accessible via VMD Main Menu → Extensions → Tk Console). When no *subcommand* is provided, a short help message will be printed.

The subcommands that can be used are summarized in table 1.

[1]`http://www.ks.uiuc.edu/Research/vmd/`
[2]`http://www.espresso-pp.de/projects/pbctools/`
[3]For more details on the startup files, see chapter "Startup Files" in the VMD User's Guide.

Subcommand	**Description**	**p.**
`set` *cell* [*options...*]	Set the VMD unit cell properties.	4
`readxst` *xstfile* [*options...*]	Read the VMD unit cell properties from an XST file.	5
`get` [*options...*]	Get the VMD unit cell properties.	6
`wrap` [*options...*]	Wrap atoms into a single unit cell.	7
`unwrap` [*options...*]	Unwrap frames of a trajectory, so that no atoms are wrapped between adjacent frames.	8
`join` *compound* [*options...*]	Joins compounds (residues, chains, segments, or fragments) that have been split due to wrapping around the unit cell boundaries, so that they are not split anymore.	9
`box` [*options...*]	(Re)Draws a box that shows the boundaries of the unit cell. The box will automatically adapt to changes in the unit cell parameters in the course of a trajectory.	10
`box_draw` [*options...*]	Draws a static box that shows the boundaries of the unit cell, but will not adapt to changes in the unitcell properties.	11

Table 1: Subcommands of the VMD-Tcl-command `pbc`.

3 `set` and `readxst` – Setting the unitcell parameters

To be able to work correctly, all other procedures of the PBCTools plugin require the VMD unitcell parameters to be set. Some file formats and their readers provide the necessary information (*e.g.* the DCD, VTF and Amber crdbox formats). When the format does not provide the information, the parameters can either be set with help of the command `pbc set` (see section 3.1), or it can be read in from a file in XST format via the procedure `pbc readxst` (see section 3.2).

3.1 `set`

Syntax
`pbc set` *cell* [*options...*]

Description
Sets the VMD unit cell properties to *cell* in the specified frames. *cell* must either contain a single set of unit cell parameters that will be used in all frames of the molecule, or it must contain a parameter set for every frame.

Example

```
# set the unit cell side length to 10 in all frames
pbc set 10.0 10.0 10.0 -all
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`).
`-first` *frame*|`first`|`now`	The first frame to use (default: `now`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `now`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-namd`|`-vmd`	Format of the unit cell parameters *cell*. When `-vmd` is used, a parameter set must be a list of the VMD unitcell parameters *a*, *b*, *c* (*i.e.* the side lengths of the unit cell) and optionally *alpha*, *beta* and *gamma* (the angles of the unit cell) for non-orthorhombic unitcells. When `-namd` is used, a parameter set must contain the three unit cell vectors *A*, *B* and *C* (the 3D-vectors of the unitcell sides) (default:`-vmd`).
-[no]`alignx`	If the option `-namd` is used and the unit cell vector *A* is not parallel to the x-axis, `-alignx` will rotate the system so that it is. If `-noalignx` is used, the function will return with a warning when *A* ist not aligned with the x-axis.

3.2 `readxst`

Syntax

`pbc readxst` *xstfile* [*options...*]

Description

Read the unit cell information from an XST or XSC file.

Example

```
# read the unit cell parameters from system.xst
pbc readxst system.xst
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`).
`-first` *frame*|`first`|`now`	The first frame to use (default: `first`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `last`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-stride` *stride*	Read only every *stride*-th timestep from the file (default: 1).
`-`[`no`]`skipfirst`	Whether to skip the first line of the file, or not (default: `-skipfirst` for XST files, `-noskipfirst` for XSC files)
`-step2frame` *num*	Conversion factor between step *num* in XST file and frame *num* in DCDs. This is useful when loading multiple XSTs and want to avoid over-writing info of earlier frames by having a unique mapping between step and frame.
`-`[`no`]`alignx`	If the unit cell vector A is not parallel to the x-axis, `-alignx` will rotate the system so that it is. If `-noalignx` is used, the function will return with a warning when A ist not aligned with the x-axis.
`-log` *logfile*	Log file used for debugging information.

4 `get` – Getting the unitcell parameters

Syntax

`pbc get` [*options...*]

Description

Gets the VMD unit cell properties from the specified frames. Returns a list of one parameter set for each frame or an empty list when an error occured.

Example

```
# get the unit cell parameters of the current frame
set cell [pbc get -now]
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`)
`-first` *frame*|`first`|`now`	The first frame to use (default: `now`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `now`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-namd`|`-vmd`	Format of the unit cell parameters. When `-vmd` is used, a parameter set will contains the VMD unitcell parameters *a*, *b*, *c*, *alpha*, *beta*, *gamma*. When `-namd` is used, a parameter set contains the three 3D unit cell vectors *A*, *B* and *C* (default: `-vmd`).
`-`[`no`]`check`	Check whether the unit cell parameters seem reasonable, *i.e.* whether the side lengths are not too small and the angles are not very small or very large (default: `-nocheck`).

5 `wrap` – Wrapping atoms

Syntax

`pbc wrap` [*options...*]

Description

Wrap atoms into a single unitcell.

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`)
`-first` *frame*|`first`|`now`	The first frame to use (default: `now`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `now`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-parallelepiped` |`-orthorhombic`	Wrap the atoms into the unitcell parallelepiped or the corresponding orthorhombic box with the same volume and center as the (non-orthrhombic) unitcell. The unitcell displacement vectors are not changed (default: `-parallelepiped`).
`-sel` *sel*	The selection of atoms to be wrapped (default: `"all"`). Use this if you don't want to wrap all atoms.
`-nocompound` |`-compound` `res`[`id`[`ue`]]|`seg`[`id`]|`chain`	Defines, which atom compounds should be kept together, *i.e.* which atoms will not be wrapped if a compound would be split by the wrapping: residues, segments or chains (default: `-nocompound`).
`-nocompoundref` |`-compoundref` *refsel*	When compounds have been defined via the `-compound` option, this defines a reference selection of atoms. After the wrapping, at least one of the atoms in this selection will be in the central image. This can be useful, for example, when water molecules should be wrapped such that the oxygen atom ends up in the central image (default: `-nocompoundref`).
`-center origin`|`unitcell` |`com`|`centerofmass` |`bb`|`boundingbox`	Specify the center of the wrapping cell. The center can be set to the origin (`origin`), to the center of the unit cell (`unitcell`), to the center of mass (`com` or `centerofmass`) of the selection specified by the option `-centersel`, or to the center of the bounding box (`bb` or `boundingbox`) of the selection specified by the option `-centersel` (default: `unitcell`).
`-centersel` *sel*	Specify the selection *sel* that defines the center of the wrapping cell in the option `-center` (default: `"all"`).
`-shiftcenter` *shift*	Shift the center of the box by *shift*. *shift* has to be a list of three numerical values. (default: `{0 0 0}`)
`-shiftcenterrel` *shift*	Shift the center of the box by *shift* (in units of the unit cell vectors). *shift* has to be a list of three numerical values. (default: `{0 0 0}`)
`-`[`no`]`verbose`	Turn on/off verbosity of the function (for debugging) (default: `-noverbose`).
`-`[`no`]`draw`	Draw some test vectors (for debugging) (default: `-nodraw`).

Example

```
# wrap the system into the orthorhombic box
# shifted by one box length in X-dir
pbc wrap -orthorhombic -shiftcenterrel 1 0 0
```

6 `unwrap` – Unwrapping atoms

Syntax

`pbc unwrap` [*options...*]

Description

If a simulation only saves the central image coordinates of a system, atoms are wrapped around when they reach the boundaries. This leads to big jumps in the coordinates of the atoms, and to bonds that stretch the whole box length. This procedure will reverse these jumps and make the movement of the atoms continuous over a series of frames. This process is not necessarily unique, so this procedure can *not* exactly reverse the effects of the command `pbc wrap`.

In the case of a simulation trajectory, the following process most probably gives the best result:

1. Go to the first frame.
2. Shape the unitcell of the frame for the best visualization by using the commands `pbc join -now` and `pbc wrap -now` with appropriate options.
3. Unwrap the trajectory, starting from the current frame, by using `pbc unwrap -first now`.
4. Visually check the result. If the system gets smeared out too fast because the diffusion is too high, repeat the procedure with successively later frames.

Example

```
# unwrap all protein atoms
pbc unwrap -sel "protein"
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`)
`-first` *frame*|`first`|`now`	The first frame to use (default: `now`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `now`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-sel` *sel*	The selection of atoms to be unwrapped (default: `"all"`). Use this if you don't want to unwrap all atoms.
`-`[no]`verbose`	Turn on/off verbosity of the function (for debugging) (default: `-noverbose`).

7 `join` – Joining residues, chains, segments and fragments

Syntax

`pbc join` *compound* [*options...*]

Description

Joins compounds of type *compound* of atoms that have been split due to wrapping around the unit cell boundaries, so that they are not split anymore. *compound* must be one of the values `res[id[ue]]`, `chain`, `seg[id]` or `fragment`.

This procedure can help to remove bonds that stretch the whole box. Note, however, that `join` is relatively slow and is required only in a very few cases. If you have a simulation trajectory that contains frames with overstretched bonds, it is usually enough to apply `join` only to the first frame and then the much faster procedure `unwrap` to all of the frames:

```
  pbc join compound -first 0 -last 0
  pbc unwrap
```

Example

```
# join all residues such that the Carbon alpha atom
# is in the central image
pbc join res -ref "name CA"
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`)
`-first` *frame*|`first`|`now`	The first frame to use (default: `now`).
`-last` *frame*|`last`|`now`	The last frame to use (default: `now`).
`-all`[`frames`]	Equivalent to `-first first -last last`.
`-now`	Equivalent to `-first now -last now`.
`-sel` *sel*	The selection of atoms to be joined (default: `"all"`). Use this if you don't want to join all atoms.
`-noref`|`-ref` *refsel*	This defines a reference selection of atoms. When joining compounds, the first atom matching the selection in each compound will be chosen, and all atoms will be wrapped into a unit cell around this atom. If `noref` is used, the first atom in the compound is the reference atom (default: `-noref`).
`-`[`no`]`verbose`	Turn on/off verbosity of the function (for debugging) (default: `-noverbose`).

8 `box` and `box_draw` – Drawing the unit cell boundaries

8.1 `box` – Automatically updateing box

Syntax

`pbc box` [*options…*]

Description

(Re)Draws a box that shows the boundaries of the unit cell. The box will automatically adapt to changes in the unit cell parameters in the course of a trajectory, as for example for simulations at constant pressure. Only a single automatically updated box can exist at a time.

Example

```
# draw a box, centered on the origin
pbc box -center origin
```

Options

`-molid` *molid*|`top`	Which molecule to use (default: `top`)
`-on`|`-off`|`-toggle`	Turn the box on, off, or toggle whether it is on or off. (default: `-on`)
`-parallelepiped` |`-orthorhombic`	Draw the box as a parallelpiped, or as the corresponding orthorhombic box. (default: `-parallelepiped`).
`-color` *color*	Draw the box in color *color*. (default: `blue`)
`-style` `lines`|`dashed`|`arrows`|`tubes`	Choose the style of the box (default: `lines`).
`-width` *width*	Define the *width* of the lines/arrows/tubes (default: `3`).
`-resolution` *res*	Use *resolution* faces for the tube style (default: `8`).
`-center` `origin`|`unitcell` |`com`|`centerofmass` |`bb`|`boundingbox`	Specify the center of the box. The center can be set to the origin (`origin`), to the center of the unit cell (`unitcell`), to the center of mass (`com` or `centerofmass`) of the selection specified by the option `-centersel`, or to the center of the bounding box (`bb` or `boundingbox`) of the selection specified by the option `-centersel` (default: `unitcell`).
`-centersel` *sel*	Specify the selection *sel* that defines the center of the wrapping cell in the option `-center` (default: `"all"`).
`-shiftcenter` *shift*	Shift the center of the box by *shift*. *shift* has to be a list of three numerical values. (default: `{0 0 0}`)
`-shiftcenterrel` *shift*	Shift the center of the box by *shift* (in units of the unit cell vectors). *shift* has to be a list of three umerical values. (default: `{0 0 0}`)

8.2 `box_draw` – Drawing a static box

Syntax

`pbc box_draw` [*options...*]

Description

Draws a static box that shows the boundaries of the unit cell, but will not adapt to changes in the unitcell properties. This might be useful when you want to draw more than one box at a time (*e.g.* to show periodic images of a box), or to show the initial box in a simulation with fluctuating box unit cell geometry.

Options

`pbc box_draw` uses the same options as the command `pbc box`, with the exception of the options `-on`|`-off`|`-toggle` and `-color`, which can not be used. To set the color of the box, use the `graphics color` command.

Example

```
# draw a box around the central image
set box0 [pbc box_draw -shiftcenterrel  0 0 0 ]
# draw a box around the central image shifted by
# the unit cell vector C
set box1 [pbc box_draw -shiftcenterrel  0 0 1 ]
```

9 Credits

The PBCTools plugin has been written by (in alphabetical order)

- Jerome Henin `<jhenin _at_ cmm.upenn.edu>`
- Olaf Lenz `<lenzo _at_ mpip-mainz.mpg.de>` (maintainer)
- Cameron Mura `<cmura _at_ mccammon.ucsd.edu>`
- Jan Saam `<saam _at_ charite.de>`

The `pbcbox` procedure copies a lot of the ideas of Axel Kohlmeyer's script `vmd_draw_unitcell`. Please submit your bug reports, comments and feature requests on the PBCTools homepage[4].

[4] `http://www.espresso-pp.de/projects/pbctools/`