PyContact - A Tool for Analysis of Non-Covalent Interactions in MD Trajectories

Tutorial by M. Scheurer and P. Rodenkirch

Main Developers: Maximilian Scheurer and Peter Rodenkirch

University of Illinois at Urbana-Champaign, Theoretical and Computational Biophysics Group

University of Heidelberg, Biochemistry Center

June 4, 2017

Version: 1.0

Tutorial Version: 1.0

Code: `https://github.com/maxscheurer/pycontact`

Project Website: `https://pycontact.github.io`

1 Introduction

Non-covalent interactions of biomolecules are known to be the cornerstones for biochemical processes: They govern molecular recognition, induce conformational changes in proteins and exhibit a plethora of other key functions in the cell. For example, cellular signaling is just working as a result of specifically evolved interactions of biomolecules.
As atomic interactions through electrostatics, hydrogen bonds or hydrophobic properties of a biomolecule, are invisible to the common microscopes, they are visible through the computational microscope: Molecular dynamics (MD) simulations aim at describing the aforementioned interactions at an atomic level of detail, thereby also yielding dynamics of proteins, DNA, ligands, membranes or other small molecules. Thus, MD is the gateway to study non-bonding interactions with high spatial and time resolution. The results of a plain MD simulation are the positions of every atom at every timestep, called the trajectory. Depending on the system and simulation time, this corresponds to a lot of data that needs to be analyzed. Furthermore, the different types of interactions have to be distinguished and the level of detail that could be studied reaches from interactions between individual atoms to complete protein chains. It depends on the scientific task or question how specific the ”resolution” of the analysis ought to be.
To target these tasks, we provide a novel tool, PyContact, that is capable of non-covalent interaction (or contact) analysis from MD simulation trajectories. Thereby, it offers high flexibility and can be used without any programming experience, as it is a GUI (graphical user interface) application in the first place. In the following sections, we will examine interactions of two key players in proteasome-guided protein degradation, i.e. Ubiquitin (Ub) and Rpn11, a metallo-protease residing in the lid of the 26S proteasome.
We already provide a short sample trajectory file. If you plan to start learning how to perform MD simulations yourself, QwikMD `http://www.ks.uiuc.edu/Research/vmd/plugins/qwikmd` is a good way to go.

User Contributions

PyContact is a very new tool, hence it is under constant development. The source code is publicly available on GitHub (`https://github.com/maxscheurer/pycontact`). Your ideas, contributions to novel features and bug reports are very much appreciated. If you want to contribute to the project, opening Pull Requests or Issues directly on GitHub is the most convenient way. Subscribing the project on GitHub, you can also follow the development progress and get to know the development of PyContact in a very transparent manner.

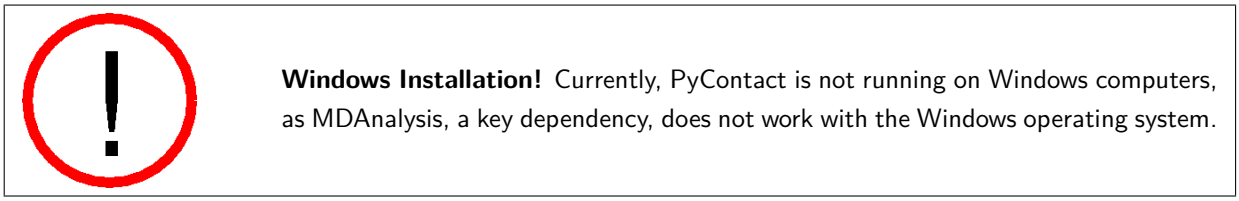
How to cite

Available soon.

2 Installation and required software

PyContact is written Python and thus requires a working Python version installed on your system. Currently, a dependency (*MDAnalysis*) does not completely support Python3, therefore PyContact also only supports Python2.7.
However, as soon as *MDAnalysis* will support Python3, PyContact will as well!
The only difficulty is the installation of *PyQt5*, which is not available as binary for Python2.7. This step will vanish once MDAnalysis supports Python3 completely.

- setuptools or pip2
- For built:
  - Cython (`pip install Cython`)
  - `working gcc compiler`
  - `python-dev`



2.1 Installation with pip

First, we need to build PyQt5, which is not available on pip.

1 Getting Qt5
To build PyQt5, a working installation of Qt5 has to be present.
Linux: Use the OS package manager to install Qt5 (and the Qt5 developer tools)
macOS: User macports or brew to install Qt5.

In both cases, make sure that `qmake` is in your `PATH` variable! (Test by executing `which qmake in the bash.`)

2 Building PyQt5
Follow the installation instructions ("Building and Installing from Source") on the PyQt5 website: `http://pyqt.sourceforge.net/Docs/PyQt5/installation.html`
Make sure you build everything using **Python2.7**! Otherwise, you will not be able to run PyContact correctly!

Test your PyQt5 installation by running `python2` in your terminal and trying the following command:

```
>>> import PyQt5
```

If the package cannot be found, make sure you have added PyQt5 to your `PYTHONPATH` environment variable.

3 Installing PyContact
PyContact as such is available on `pip`. To install it, just run

```
pip install ---upgrade pycontact
```

in the terminal. This will download and install the other dependencies, which are available on pip.

3 Basic analysis of protein interactions

Before taking off, open the PyContact tool by executing `pycontact` in your terminal.

1 Loading an MD trajectory

First, we need to load a trajectory (i.e. the coordinates for every simulation timestep) together with the topology (information about bonds etc.) into the tool. To do so, click on File → Load Trajectory Data (or hit Ctrl+I / ⌘+I). Then, click on the Topology button and select the `rpn11_ubq.psf` file in the tutorial folder. Afterwards, click on Trajectory and select the trajectory file `rpn11_ubq.dcd`, also residing in the tutorial folder. As we want to elucidate the interactions between Rpn11 ("segid RN11") and Ubiquitin ("segid UBQ"), we will put those in the input selections 1 and 2, respectively. See Tab. (1) for detailed description of the input fields.

Protein-internal contacts. If you want to scan for protein- internal interactions, type "self" into the selection 2 text field. PyContact will then find contacts between amino acids in selection 1 that are more than 4 residues apart.

Finally, click on OK to load the trajectory into the program and run the atom-atom contact analysis.

When the task is accomplished, the Status field should say "50 frames loaded".

Table 1: Description of input arguments for trajectory loading

Input	Description	Tutorial Value
distance cutoff	maximal atom-atom distance (in Å) for contact scoring	5.0
angle cutoff	cutoff angle for hydrogen bonds (in deg)	120.0
acc-h cutoff	distance cutoff between the hydrogen bond acceptor and the H-atom	2.5
selection 1	atom selection text for the first selection (contacts are mapped between first and second selection) [A]	segid RN11
selection 2	atom selection text for the second selection	segid UBQ

[A] MDAnalysis selection language is used:
`https://pythonhosted.org/MDAnalysis/documentation_pages/selections.html`

2 Visualizing interactions between amino acids

Now that we have the initial analysis of atomic contacts run, we need to accumulate their scores to the corresponding amino acids. Please click on Accumulate Scores and check resid and resname for both selections. Thus, PyContact will accumulate the individual atom-atom interaction scores to the respective amino acids, i.e. the residue name and the residue ID. Click on OK to finally run this task. After the process is finished (indicated by the progress bar), you should see some colorful stuff in the timeline. You do? Great, so let's discuss what all these colors mean... First, have a look at the first column of the timeline: You ought to see a title of the contact, such as "ASN133-THR9". On the right hand side of the title, you'll see small boxes with different colors and color intensities: The individual boxes correspond to the frames in the trajectory. The color intensity shows the contact score, and the color as such points out wether the interaction is established by sidechain or backbone interactions (helpful for protein and DNA interactions) [1]. Play around with the Accumulate Scores menu, try different selections for the accumulation and see what you get.

3 Basic contact filtering

Table 2: Filtering Options in PyContact

Input	Description	Comment
Frame Stride	Stride of the frames being displayed in the timeline	
Timeline Range	Timeline frames will be displayed only in the given range	If Filter Range box is checked, only the displayed frames of the contacts are taken into account for further filtering.
Total Time	Total contact lifetime [B]	
Mean/Median/HB %	Mean/Median contact score or hydrogen bond percentage (HB %) must be greater/smaller than the given value [B]	
Sort by: mean, median, bb/sc type, contact type, total time, mean lifetime, median lifetime	Contacts are sorted *ascending* or *descending* according to the selected property [B]	
Show only: hbonds, hydrophobic, saltbridges, unknown	Show contacts with the selected type only [B]	

[B] Only applied if the corresponding checkbox is checked.

[1] Green: sidechain-sidechain interaction
Yellow: backbone-sidechain interaction
Blue: backbone-backbone interaction

4 Exporting Contact Data

For various contact data exportation tasks, we provide a tool, accessible in the menu under Tools → Export Contact Data or by hitting Ctrl+E / ⌘+E. The first tab available offers to save the current contact view from the main window to a file. The format can be chosen in the dropdown menu on the left, where the common png, as well as a svg vector graphics format are disposable.
The next tab brings us to the histogram tool, which allows a fast and clear visualization of useful properties like Mean Score, Mean Lifetime or Hbond percentage (selectable on the right hand side of the widget). Two major histogram options are available, General Histogram and Bin per Contact. The former one groups the corresponding property values in numerical bins. If you wish to use the analyzed contacts for the bin selection, you may choose Bin per Contact.
To update your choice and draw the histogram, simply click on Show Preview. Using the Bin per Contact option, you may want to adjust the font size of the bin labeling, which can be easily achieved with the bin per contact font size text field. To save the histogram, pick a suitable file format from menu on the right and press Save Histogram to choose the file location.
Another option of visualization of our data can be found in the third tab, called Contact Map. Here the same properties, which are also provided in the histogram tool can be plotted grayscale matrix, using the two selections on the two axes respectively. Similar to the tools before, the view can be updated via the Show Preview button and be saved by clicking on Save Map.
In order to view the trajectory data with VMD, go to the VMD tab and let the tool create a suitable tcl script for you. If you wish to differentiate between single contacts, you may tick the Split selections for each contact checkbox. Additional selection texts can be given in the two text fields below.
Finally, for further data analysis purposes we provide you with a raw data export functionality in the last tab, titled Plain Text. Pick your preferred combination of properties you want to export with the checkboxes on left and click on Export to text, which file open the common file dialogue. The data will be written in a tab separated based ASCII text file.

4 Contact area calculation

To decipher the contact area between Rpn11 and Ubiquitin in a time-dependent manner, we can use the built-in SASA widget in PyContact. Open the widget by clicking on Tools → Contact Area Calculation or hit Ctrl+A / ⌘+A.

1 Selecting the trajectory
First, click on Load Data and select the topology and trajectory files of the Rpn11-Ubq simulation as described in section 3.

2 Atom selections
To specify the atom selections for contact area calculations, one need to understand the underlying principle of solvent-accessible surface area (SASA) calculations to some extent. (...)
Let's say we are interested in the contact area of Rpn11 to Ubq. Then we specify "segid RN11" in the Selection text field. We want to shrink the selection to the Ubq interface, so we only select those residues in 5Åproximity by typing "segid RN11 and around 5 segid UBQ" in the Restriction text field. Finally, the program will need to subtract the SASA of protein residues at the outside of the interface. Type "protein" in the Selection 2 text field and check the contact checkbox.

3 Running the calculation Choose the number of cores to run the calculation on. Then click on Calculate to run the SASA/contact area calculations. The results will be plotted directly in the GUI and are available for export. See Fig. 1 for an example output.

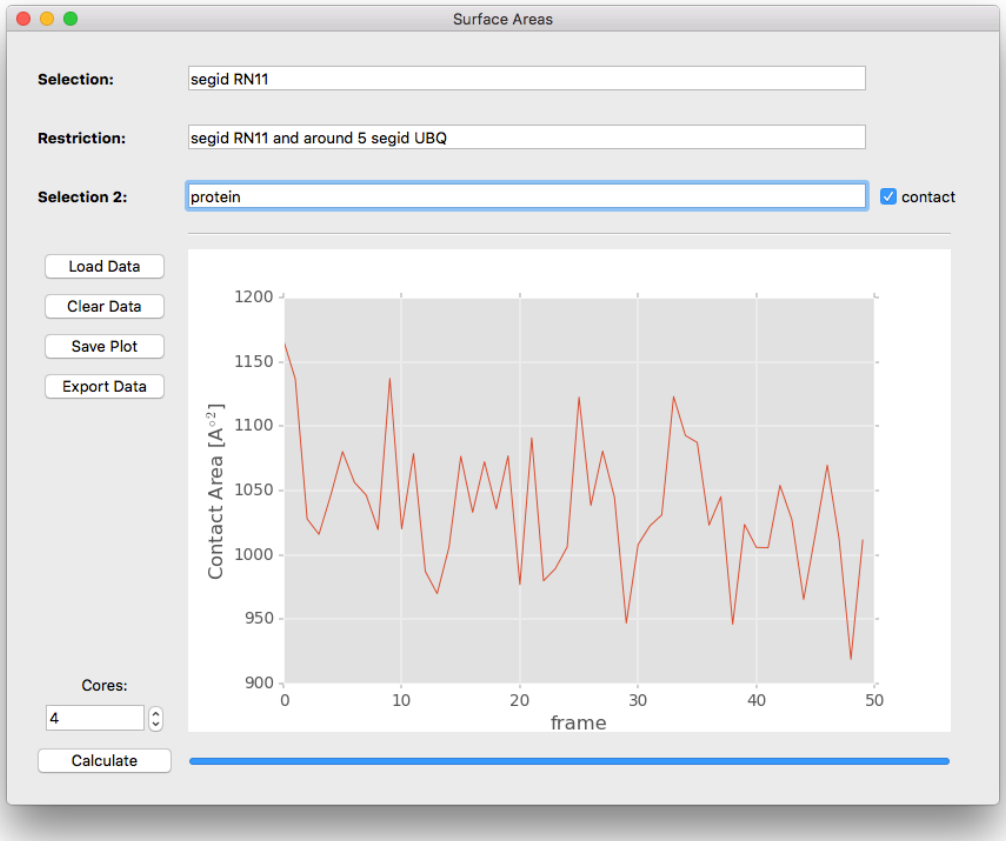


Figure 1: Contact area calculation example. As explained in section 4, we calculated the time-dependent evolution of the contact area between Rpn11 and Ubiquitin.

5 Python scripting for job automation

For users that want to automate analysis of their trajectories with PyContact, we provide an easy to use interface for the most important features. The underlying workflow is suggested to be as follows:

1. Writing the Python script:
    - job configuration for loading the trajectory and atom-atom contact scoring are defined
    - subsequent score accumulation parameters
    - number of cores can also be set
    - job results are to be saved as a .session file
2. Load the .session file into PyContact and proceed as usual

This scripting capability allows faster contact analyses for trajectories with identical parameters for example. Furthermore, it shall motivate users to understand how PyContact works, dig into the code and probably come up with own feature ideas and code.

```
from PyContact.core.Scripting import PyContactJob, JobConfig

# define input files and parameters
job = PyContactJob("/path/to/topology", "/path/to/trajectory", "title",
↪  JobConfig(5.0, 2.5, 120, [0,0,1,1,0], [0,0,1,1,0], "segid A", "segid B"))
# running the job on 4 cores
job.runJob(4)
# writing the session to file
job.writeSessionToFile()
```