SMDAgui: A Graphical User Interface for Molecular Dynamic analysis

Thibault Tubiana1 ,Melaine Kuenemann1

*1Institut de Recherche Servier, 125 chemin de Ronde, 78290 Croissy*

# Summary

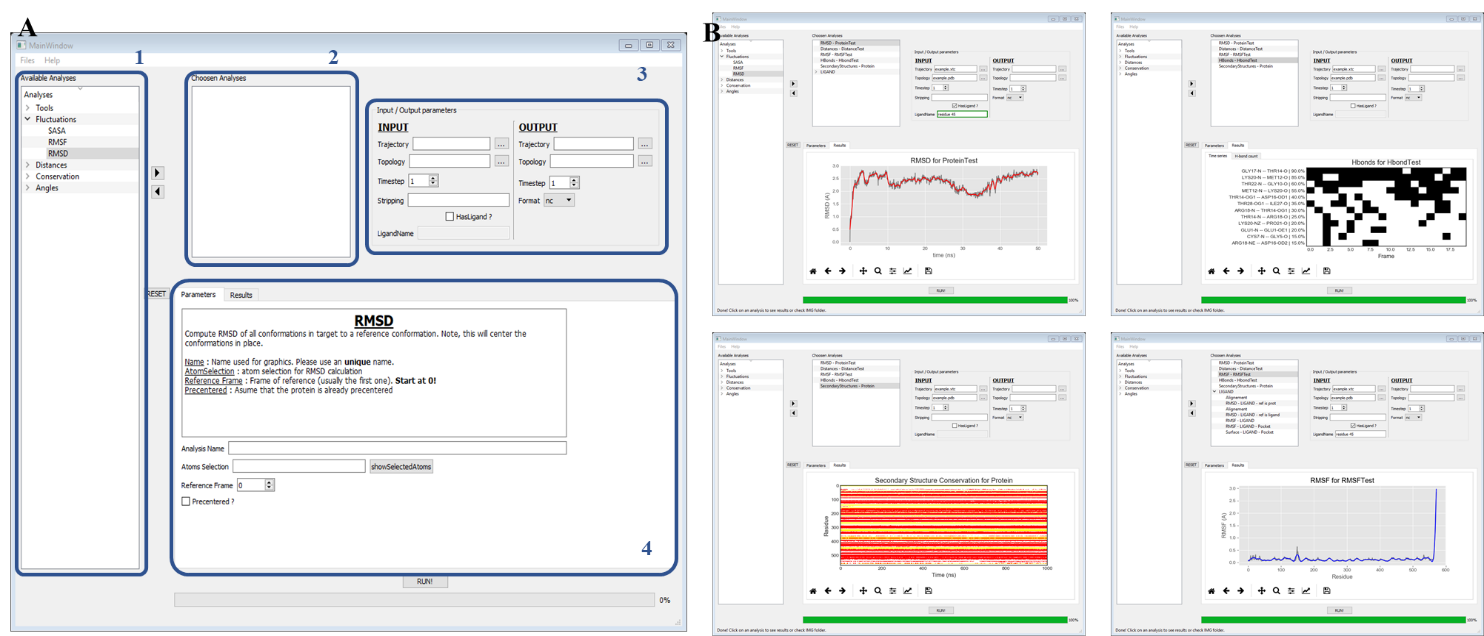
Molecular dynamic simulation (MD) consists of observing the evolution of a biomolecular system over time. Such simulations can produce significatively amount of data called trajectories. They are essentially composed of each 3D atom’s coordinates in a succession of matrices called frames. Generally, modelers do the same analysis and generate the same graphs to check if the MD was correct if the system has reached equilibrium or to analyze some components of the MD. While most of the MD packages, such as Amber1 or Gromacs2 have their own analysis tools, they are not always straightforward to use and might need a learning time before using it. GROMACS analysis tools can generate simple graphics files that can only be read with XmGrace4 but it also has almost a hundred analysis software that requires reading the trajectory every time a new call is made. Conversely, Amber uses CPPTRAJ3 that can queue the analysis to save time and memory but produce raw data only and it is often necessary to implement or improve the graphics. Finally, other major MD software such as Charmm5 or NAMD6 relies on the use of very complete and complex tools for neophytes, like VMD7, or tools required programming skills (Bio3D8, MMTK9, ptraj3, MDAnalysis10, MDTRAJ11). SMDAgui is a Simple Molecular Dynamics Analysis tool with a Graphical User Interface (GUI) targeting neophyte MD users and modelers who do not want to spend too much time generating the same graphics repeatedly.

# Execution

The SMDAgui core is developed in Python12, the GUI in PyQT5 and the MD analysis engine chosen is MDTRAJ which has the advantage of being very flexible, performant, and multi-platform. Analyses data are stored and saved with Pandas and graphics are generated with Matplotlib. Most of these Python packages are installed by default with the Anaconda 3 packages, except for MDTRAJ which can be easily installed with condos.

As seen in Fig 1.A, the main windows are simplified to its maximum: 4 frames are only available for (i) input and output MD files (Fig 1.3), Choose an analysis from the available analysis list (Fig 1.1) and put it in the analyses queue (Fig 1.2) and set up analysis parameters (Fig 1.3). One of the tricky tasks during analysis parameterization is to adapt to the selection syntax and be sure that we select the right atoms. Here MDTRAJ uses a very simple selection syntax close to the human language (see <http://mdtraj.org/1.9.3/atom_selection.html>) and to help the user, a real-time selection syntax checking function was implemented in SMDAgui and the possibility to view, in Dataframe, windows, the selected atoms. Moreover, a keyword was added to give the opportunity to the user to select all atoms close to another (e.g.: “within 3 of resname ATP” will select all atoms less than 3 angstroms of the ATP molecule). This keyword is not implemented yet in MDTRAJ and is very useful for ligands or hydrogen bonding analysis.

Once all analyses made, graphics are saved in PNG format and plotted in a matplotlib widget in the “results” tab of an “analysis” object. The user selects an analysis of the queue (Fig 1.2) and the figure will be automatically updated. It is fully interactive, and users can change the zooming level, range, colors (as a regular matplotlib window). If several plots are available for one analysis type (such as hydrogen bonding), the other ones will be plotted in a second tab within the results tab. All graphics are made with matplotlib from Pandas DataFrame. The Dataframe is saved in CSV format and allows the user to generate its own graphics if he wishes.



*Fig 1 – (A)* *SMDAGui Main windows with some output examples. (1) List of implemented analyses (2) Analysis queue (3) Frame for Input / Output parameters (4) Frame for analysis parameterization and results display. Data was obtained from a backbone trajectory of the Norovirus capsid dimer* 21,22 *and from the NMR conformation of growth-blocking peptide (2eqq*23*)*

If the user adds several trajectories and check the “replica” checkbox, each trajectory is considered as a replica and all analyses in the queue will be executed on all replicas. The results will then be displayed in the new tabs 'Results' and a final tab will show all the results of the replicas between them.

Finally, the session can be saved in JSON format and be reopened afterward. SMDA saves queued analyses with their parameters, load CSV Dataframes and regenerate graphics from data without reading the trajectory and making the calculation again (which can take time for big trajectories).

# Implemented analyses

Since SMDA uses MDTRAJ engine to generate analysis and graphical output, most of the mandatory analysis are implemented:

* Root Mean Squared Deviation (RMSD)
* Root Mean Square Fluctuation (RMSF)
* Accessible solvent Surface Area (SASA)
* Distances
* Secondary Structures
* Hydrogen bonding
* Radius of gyration
* Dihedral angle
* Classical angle
* trajectories operations (joining, slicing, stripping, merging, Imaging, converting…)

Thanks to SMDA modularity, this list can be easily extended in future developments to add new analysis like co-insertable hydrophobic protrusions for peripheral membrane protein binding site detection13, clustering14, Cation-pi conservation 15 or hydrophobic contacts.

# Deployment and Dependencies

This application is freely available on [GITHUBLINK] under the [XXX] license and requires Python 3.3+, MDTraj 1.9.0+, PyQT5, Pandas16, Scipy17, Numpy18, Matplotlib19, and seaborn20. SMDAgui can be installed using the command [PYPI COMMAND]

# Installation and execution

SMDA can be installed through PyPI with the command “pip install smda” or through conda with “conda install -c tubiana -c conda-forge smda”

# Execution

SMDA can be executed with the “smda” command if the software was installed through PyPI or Conda.   
If the sources were cloned from GitHub, SMDA will be through a python interpreter with “python smda.py”.

# Acknowledgment

I would like to thank Servier’s Molecular Modelling and Cheminformatics team for helping me test this software and to use it regularly for their analyses.

# Bibliography

1. The Amber biomolecular simulation programs. *J. Comput. Chem.* **26**, 1668–1688 (2005).

2. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* **1**–**2**, 19–25 (2015).

3. PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data. *J. Chem. Theory Comput.* **9**, 3084–3095 (2013).

4. Turner, P. J. XMGRACE, Version 5.1. 19. *Cent. Coast. Land-Margin Res. Oregon Grad. Inst. Sci. Technol. Beaverton, OR* (2005).

5. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J. Phys. Chem. B* **102**, 3586–3616 (1998).

6. Scalable Molecular Dynamics with NAMD. *J. Comput. Chem.* **26**, 1781–1802 (2005).

7. VMD: visual molecular dynamics. *J. Mol. Graph.* **14**, 33–38, 27–28 (1996).

8. B.J., G., A.P.C., R., K.M., E., J.A., M. & L.S.D., C. Bio3D: An R package for the comparative analysis of protein structures. *Bioinformatics* **22**, 2695–2696 (2006).

9. Hinsen, K. The molecular modeling toolkit: A new approach to molecular simulations. *J. Comput. Chem.* **21**, 79–85 (2000).

10. Michaud-Agrawal, N., Denning, E. J., Woolf, T. B. & Beckstein, O. MDAnalysis: A toolkit for the analysis of molecular dynamics simulations. *J. Comput. Chem.* **32**, 2319–2327 (2011).

11. McGibbon, R. T. *et al.* MDTraj: A Modern Open Library for the Analysis of Molecular Dynamics Trajectories. *Biophys. J.* **109**, 1528–1532 (2015).

12. *Python Reference Manual*. (1995).

13. Fuglebakk, E. & Reuter, N. A model for hydrophobic protrusions on peripheral membrane proteins. 1–27 (2018).

14. Tubiana, T., Carvaillo, J. C., Boulard, Y. & Bressanelli, S. TTClust: A Versatile Molecular Simulation Trajectory Clustering Program with Graphical Summaries. *J. Chem. Inf. Model.* **58**, 2178–2182 (2018).

15. Grauffel, C. *et al.* Cation-π interactions as lipid-specific anchors for phosphatidylinositol-specific phospholipase C. *J. Am. Chem. Soc.* **135**, 5740–5750 (2013).

16. Mckinney, W. *pandas: a Foundational Python Library for Data Analysis and Statistics*. *undefined* http://pandas.sf.net (2011).

17. *SciPy: Open source scientific tools for Python*. (2001).

18. Bressert, E. *SciPy and NumPy: an overview for developers*. (‘ O’Reilly Media, Inc.’, 2012).

19. Matplotlib: A 2D Graphics Environment. *Comput. Sci. Eng.* **9**, 90–95 (2007).

20. seaborn: statistical data visualization — seaborn 0.10.1 documentation. https://seaborn.pydata.org/.

21. Tubiana, T., Boulard, Y. & Bressanelli, S. Dynamics and asymmetry in the dimer of the norovirus major capsid protein. *PLoS One* **12**, e0182056 (2017).

22. Tubiana, T. Dynamique d’assemblage de la capside des norovirus. (2017).

23. Umetsu, Y. *et al.* C-terminal elongation of growth-blocking peptide enhances its biological activity and micelle binding affinity. *J. Biol. Chem.* **284**, 29625–29634 (2009).