Nanoparticle Modeller for Martini - *Martini-PyNP* - A Proposal

Sang Young Noh

November 5, 2021

1 Introduction

The introduction of software frameworks when constructing biomolecular systems and phase systems has revolutionized the pace of research in the molecular simulation field. This is due to the ease of entry with the python programming language, as well as the use of molecular template contruction software, such as moltemplate (for general use) [5], insane (for the use of Martini and bilayers/proteins) [7]. Nanoparticle structures (NPs), on the other hand, have seen a number of software that have seen use such as nanomodeler [3]. The case of nanomodeler and its use highlights the importance but also the difficulty of creating a general software that can manage the construction of structures. As highlighted by Franco-Ulloa *et al*, there are two main reasons for this:

- Construct complex 3D models of the inner metal core and outer layer of organic ligands.
- Correctly assign force-field parameters to these composite systems.

Other factors that inhibit the ease of construction of such software is that NPs can occur with several sizes and morphology, for example the carbon nanotube being a classical model which should be taken into account to create a general software.

Recent developments in existing software packaged have helped to make this goal closer towards being substatial. MDAnalysis is one of the de-factor molecular packages that is available for manipulating PDB/GRO/XYZ molecular topologies, as well as TRR/DCD/XTC binary trajectories from simulation files to analyze according to how one wants to look at the simulation. The most recent version, MDAnalysis-1.0.1, added new functionalities, of structural analysis but especially a cross-link compatibility with SMILES strings of molecules, which in turn allows it to utilize the RDKit library [1], which is already a very-well established chemiinformatics library. From the introduction of the RDKit conversion functionality in MDAnalysis (more information here - https://www.mdanalysis.org/2020/08/29/gsoc-report-cbouy/) the potential for simply inputting a smiles string to construct a ligand on the nanoparticle has been a potential. RDKit is convenient as it can detect what ligands is not suitable due to their unrealistic nature.

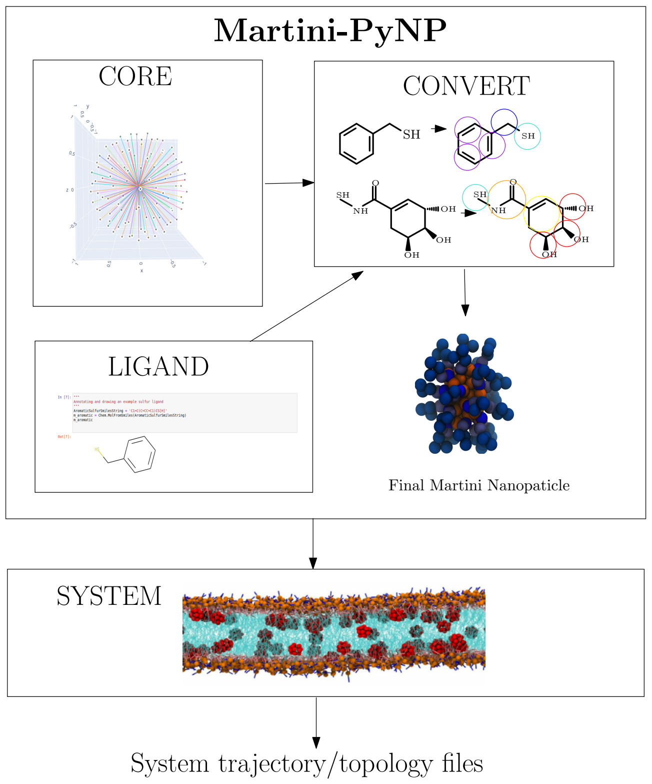
Another advantage with the RDkit infrastructure is the way aromatic atoms can be identified,. Following the procedures by Alessandri *et al* [2], we already know that with MARTINI3, aromatic and non-aromatic atomic moeities can have significantly different coarse-graining procedures, where instead of a normal 4-to-1 conversion from atomistic to coarse-grained, the mapping may require a 3-to-1 or even a 2-to-1 mapping. By utilizing the mentioned feature in RDkit, it would theoretically be a straightforward feature to identify and convert the aromatic features as a 2-to-1 conversion, while keeping the other parts as 3-to-1/4-to-1 conversions. This follows similar approaches from PyCGTOOL [4]. In the case of the NP core, previous approaches to NP construction include those that were done by Monticelli [6] which involved the formation of rigid bonds to the other beads within the sphere. In this case, we adopt this appraoch to create the central core of the NP, but as the NPs are larger in this case, the issue of both periodic boundary conditions and excessively long covalent bonds can create instabilities within the NP. Hence, here we have added a distance cutoff for the bond formation between the core atoms.

2 Proposed Code Workflow

The code workflow is primarily developed in Python, to leverage the wide range of molecular structure libraries already avaliable. Here, we have divided the primary construction code of Martini-PyNP into 4 main modules:

- CORE
- LIGAND
- CONNECT
- SYSTEM

The CORE module handles the central sphere/nanotube construction, the LIGAND module would handle the construction of the ligand(s), to remove/parse the structure so that unrealistic steric hindrances do not cause the breakdown of the simulation. The CONNECT module handles the attachment of the ligands onto the NP surface - where one can determine one wants to make the ligands into a striped/janus/random mix type, or a custom surface pattern that one can define. Also, the conversion of this system to the MARTINI 3 morphology would be done in this module. The SYSTEM deals with the construction of sample systems that the NP(s) can be placed, whether it be a lipid bilayer or a solvent system. This would follow similar protocol as the INSANE bilayer builder [7]. However, in this case the option for multiple-NP systems would be valid.



```
In [7]: """
        Annotating and drawing an example sulfur ligand
        """
        AromaticSulfurSmilesString = 'C1=C(C=CC=C1)CS[H]'
        m_aromatic = Chem.MolFromSmiles(AromaticSulfurSmilesString)
        m_aromatic
Out[7]:
```

Figure 1: General flowchart for how the NP construction takes place. The SYSTEM module has been omitted from the figure, but will be added later

3 Conclusion

This proposal highlights the project idea for creating an automated NP construction based on Python, which has been preliminarily named *Martini-PyNP*, following the observation that there is no unified approach to building a coarse-grained NP system (striped, tube or otherwise) that is easy to install and without having to go through the tedious process of having the design a separate in-house coding base. Based on the recent features of libraries such as MDAnalysis and RDkit, this proposal shows the possibility of making the library that enables this process.

4 Reference Links

- https://cedric.bouysset.net/blog/2020/06/18/gsoc-rdkit-to-universe - RDkit tutorial
- https://www.sciencedirect.com/science/article/pii/S0022283621000358 - Moltemplate literature
- https://pubs.acs.org/doi/10.1021/acs.jctc.8b01304 - Nanomodeler
- https://pubs.acs.org/doi/abs/10.1021/jacs.6b11717 - Heterojunction morphologies
- http://cgmartini.nl/index.php/martini-3-0 - Martini 3 parameters
- https://chemrxiv.org/engage/chemrxiv/article-details/60f3ea062b910135237380eb - Mapping of the ligands to specific beads of martini
- https://cedric.bouysset.net/blog/2020/08/07/rdkit-interoperability - Information on interoperatbility of the RDKit library and the MDAnalysis library
- http://www.cgmartini.nl/index.php/example-applications2 /35-downloads/exampleapplications/193 f16 - Fullerene models in Martini

References

[1] RDKit: Open-source cheminformatics. https://www.rdkit.org/.

[2] R. Alessandri, J. Barnoud, A. S. Gertsen, L. Patmanidis, A. H. de Vries, P. C. T. Souza, and S. J. Marrink. Martini 3 Coarse-Grained Force Field: Small Molecules. *ChemRxiv*, 2021.

[3] S. Franco-Ulloa, L. Riccardi, F. Rimembrana, M. Pini, and M. D. Vivo. NanoModeler: A Webserver for Molecular Simulations and Engineering of Nanoparticles. *Journal of Chemical Theory and Computation*, 15(3):2022–2032, 2019.

[4] J. A. Graham, J. W. Essex, and S. Khalid. PyCGTOOL: Automated Generation of Coarse-Grained Molecular Dynamics Models from Atomistic Trajectories. *Journal of Chemical Information and Modeling*, 57(4), 2017.

[5] A. I. Jewett. Moltemplate: A tool for coarse-grained modeling of complex biological matter and soft condensed matter physics. *Journal of Molecular Biology*, 433(11):166841, 2021.

[6] L. Monticelli. On Atomistic and Coarse-Grained Models for c60 Fullerene. *Journal of Chemical Theory and Computation*, 8(4), 2012.

[7] R. A. Wassenaar, H. I. Ingolfsson, R. A. Bockmann, D. P. Tielemann, and S. J. Marrink. Computational Lipidomics with insane: A Versatile Tool for Generating Custom Membranes for Molecular Simulations. *Journal of Chemical Theory and Computation*, 11:2144 – 2155, 2015.