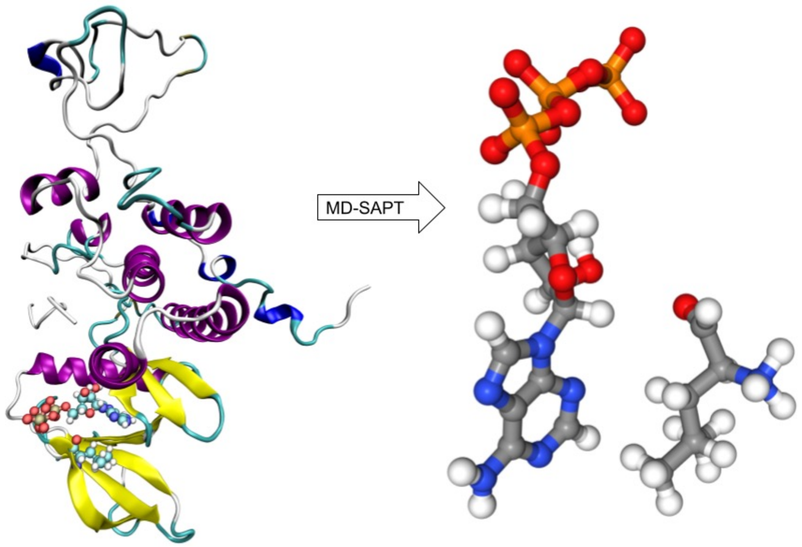
MD-SAPT: Python Based Toolkit for Running Symmetry Adapted Perturbation Theory Calculations on Molecular Dynamics Trajectories

Alia Lescoulie, Astrid Yu, and Ashley Ringer McDonald
Department of Chemistry and Biochemistry, College of Science and Mathematics, California Polytechnic State University, San Luis Obispo, CA, 93407



MEK1 with ATP and LEU-74 highlighted (left). ATP and LEU-74 visualized by MD-SAPT viewer after preparation for SAPT calculations (right).

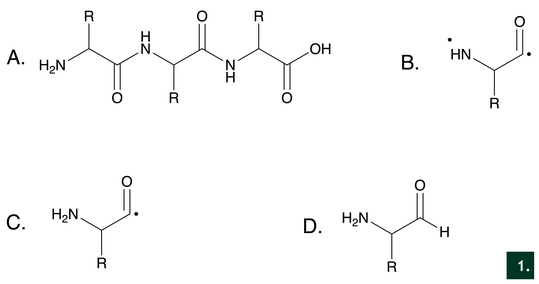
Introduction

We have developed a software package called Molecular Dynamics - Symmetry Adapted Perturbation Theory (MD-SAPT) to simplify the process of applying quantum chemical theory to intermolecular interactions in proteins whose structures generated by molecular dynamics (MD) simulations, which is a way to model how proteins move over time. We chose to apply symmetry-adapted perturbation theory (SAPT) calculations to breaks the interaction energy down into its physically relevant components.
SAPT calculations give the total non-covalent interactions between two molecules.[2] When attempting to run SAPT calculations of polypeptide residues from MD data several challenges emerged. Setting up SAPT calculations requires multiple steps:

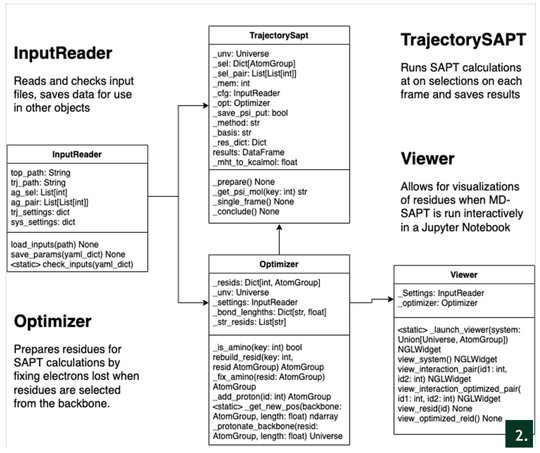
1. Pulling the coordinates for the residues of interests from the simulation trajectory
2. Editing the structure to add protons (Figure 1)
3. Determining the correct charge and multiplicity for the fragments
4. Passing that data into the calculation.

Each of these steps requires different tools and resulted in an unintuitive and error prone process. This challenge provided the impetus for developing MD-SAPT, which automates this workflow.

MD-SAPT is an open-source single-package software solution written in Python for running SAPT calculations for structures from MD simulations. It is is designed for both high-performance computing (HPC) clusters and interactive Jupyter Notebooks. It is distributed using Anaconda, a widely-used tool for managing Python environments in scientific computing. Full documentation for MD-SAPT and installation instructions are online at mdsapt.readthedocs.io which is linked in figure 4.



A: The Polypeptide backbone before residue is selected. B: The residue backbone after being selected from the polypeptide. C: The residue backbone after adding a hydrogen on the amino group. D. The residue after a hydrogen is added on the C-terminus to balance the spin state.



MD-SAPT uml diagram showing the structures or each object.

Methods

We designed MD-SAPT with an object-oriented paradigm, providing users with a simple and modular interface. There are four objects: InputReader, Optimizer, TrajectorySAPT, and Viewer, each encapsulating their own methods, data, and responsibilities. Their design and function is detailed in figure 1. No coding is required to use MD-SAPT; users only need to provide a simple input file (Figure 3) to analyze data for common workflows. However, for more complex tasks, the user can utilize MD-SAPT as a Python library within their own code.
To ensure the reliability of the software, we have automated unit testing suite that covers 95% of our code. Additionally, we have an automated continuous integration workflow that regularly builds and tests our code in order to provide a seamless experience for end users.

```yaml
topology_path: mdsapt/tests/testing_resources/testtop.psf
trajectory_paths:
  - mdsapt/tests/testing_resources/testtraj.dcd
selection_resid_num:
  - 11
  - 199
int_pairs:
  # Place pair of  selections defined above in a list of lists
  - [11, 199]
trajectory_settings:
  start: 1
  stop: 2
  step: 1
system_settings:
  ncpus: 16
  memory: 48GB
  time: '24:00:00'
opt_settings:
  pH: 7.0
sapt_settings:
  method: 'sapt0'
  basis: 'dz'
  settings:
    reference: 'rhf'
  save_psi4_output: true
```

3.

Example input file for MD-SAPT

4.

QR code to access documentation and demonstration notebook

Demonstration of Software

To run a job, the user first generates a blank input file with the *mdsapt_get_runinput.py* script and inputs the settings for their analysis. The YAML input file Figure 2 supplies the code with the MD simulation files, the residue selections, the simulations frames to study, the computing resources to reserve, the pH of the system, and settings used in running the SAPT calculations.

Due to the aforementioned use of objects in MD-SAPT's code, users only need to enter four lines (figure 5) of information. They can also use the provided *mdsapt_run_sapt.py* script which takes an input file and returns the results in the form of a csv file.

```python
mdsap_run.py

from mdsapt import InputReader, Optimizer, TrajectorySAPT
In = InputReader('input.yaml')
Opt = Optimizer(In)
sapt = TrajectorySAPT(In, Opt).run(In.start, In.stop, In.step)
```

5.

Example script for Running TrajectorySAPT

Acknowledgments

We acknowledge the support of NSF award CHE-2018427. This award provided the computational resources to support this project through the MERCURY Consortium Skylight cluster on Palmetto.

REFERENCES

1 Sabsay, K. R.; Lee R. T.; Ravatt, L. M.; Oza J. P.; McDonald A. R. *Journal of Chemical Informatics and Modeling* **2019**, 59, 2383-2393.
2 Smith, D. G. A. et al. *The Journal of Chemical physics* **2020**, 152, 184108
3 Gowers, R. J.; Linke, M.; Barnoud, J.; Reddy, T. J. E.; Melo, M. N.; Seyler, S. L.; Doman´ski, J.; Dotson, D. L.; Buchoux, S.; Kenney, I. M.; Beckstein, O. In *Proceedings of the 15th Python in Science Conference*, ed. by Benthall, S.; Rostrup, S., **2016**, pp 98- 105.
4 Michaud-Agrawal, N.; Denning, E. J.; Woolf, T. B.; Beckstein, O. *Journal of Computational Chemistry* **2011**, 32, 2319-2327.
5 Nguyen, H.; Case, D. A.; Rose, A. S. *Bioinformatics* **2018**, 34, ed. by Valencia, A., 1241-1242.