multiSMD Manual

A comprehensive Python toolkit for multi-directional Steered Molecular Dynamics simulations

Table of Contents

---

1. Introduction

multiSMD is a Python-based toolkit designed to automate multi-directional Steered Molecular Dynamics (SMD) simulations in NAMD and GROMACS. It systematically probes mechanical anisotropy in biomolecular systems by:

- Generating pulling vectors covering a hemispherical space (default: 9 directions)
- Preparing complete input files for SMD simulations
- Providing analysis tools for force profiles and structural changes

Key applications include:

- Studying direction-dependent protein-protein/ligand interactions
- Investigating mechanical stability of molecular complexes
- Preparation for analyzing unbinding pathways and energy landscapes
- Mechanical anisotropy studies of biomolecular systems

---

2. Installation

Requirements:

- Python 3.x
- MDAnalysis
- NumPy
- SciPy

Recommended Installation:

```
pip install -r requirements.txt
```

Alternative Installation:

```
pip install numpy scipy MDAnalysis
```

For consistent and reproducible results, we recommend using a Python virtual environment and the provided requirements.txt file.

3. Quick Start

For NAMD:

```
python multismd_namd.py smd_output system.pdb system.psf system.vel system.coor system.xsc
par_all36_prot.prm template.inp template.run 'name CA and resid 1:50' 'name CA and resid 100:150'
```

For GROMACS:

```
python multismd_gromacs.py smd_output system.pdb system.gro md_files.zip template.mdp 'name CA and
resid 1:50' 'name CA and resid 100:150'
```

4. Detailed Usage

4.1 NAMD Version

Required Parameters:

Parameter	**Description**
`output_dir`	Main output directory for generated files
`input_pdb`	System structure in PDB format
`input_psf`	PSF topology file

`input_vel`	Velocity restart file
`input_coor`	Coordinate restart file
`input_xsc`	Periodic cell parameters
`input_par1`	CHARMM force field parameters
`template_inp`	NAMD configuration template
`template_run`	Job submission script template
`input_sel1`	MDAnalysis selection for fixed atoms

Optional Parameters:

`--repeats REPEATS`	Number of replicates per pulling direction (default: 1)

Example:

```
python multismd_namd.py --repeats 5 output complex.pdb complex.psf eq.vel eq.coor eq.xsc
charmm36.zip smd_template.inp cluster_run.sh 'segid PROT and name CA' 'segid LIG and name C1 C2'
```

4.2 GROMACS Version

Required Parameters:

Parameter	**Description**
`output_dir`	Main output directory
`input_pdb`	System structure (for chain info)
`input_gro`	GROMACS structure file
`input_md`	Zipped simulation restart files (.gro, .top, toppar, .ndx)
`template_mdp`	GROMACS mdp template

`input_sel1`	Fixed atoms selection
`input_sel2`	Pulled atoms selection

Optional Parameters:

`--repeats REPEATS`	Number of replicates per pulling direction (default: 1)

Example:

```
python multismd_gromacs.py --repeats 3 output complex.pdb eq.gro restart_files.zip smd.mdp 'chain A and name CA' 'resname LIG and name C*'
```

---

5. Output Structure

Generated Directory Hierarchy

```
output_directory/
├── SMD_constraints.pdb      # Visualization of restraint assignments
├── vmd_script.tcl           # VMD visualization script
├── master.run               # Master script to run all simulations
└── SMD_theta_X_phi_Y/       # Individual pulling direction directories
    ├── smd_input.inp        # Generated input file
    ├── run.bash             # Individual run script
    └── pulling_vector.dat   # Pulling vector parameters
```

Simulation Execution (for NAMD)

```
# Individual direction
./Output/SMD_theta_0_phi_0/run.bash

# All directions
./master.run
```

6. Output Analysis

Analysis Scripts:

- For NAMD:

  ```
  python analysis_namd.py directory sel_const sel_pull [--repeats REPEATS] [--no-forces] [--no-hb]
  [--red-factor RED_FACTOR] [--plot-only]
  ```

- For GROMACS:

```
python analysis_gromacs.py directory pdb_file sel_const sel_pull [--no-hb] [--red-factor RED_FACTOR] [--plot-only]
```

Output Metrics:

1. Force vs. time profiles (time in fs for NAMD/ns for GROMACS vs force in pN)
2. Force vs. distance plots (Å vs pN)
3. Hydrogen bond dynamics over time
4. Maximum rupture forces with statistical analysis
5. Vector analysis of pulling directions

Generated Files per Simulation:

- smd_force_time_N.dat - Force vs time series for each replicate N
- smd_force_dist_N.dat - Force vs distance
- smd_hb_time_N.dat - Hydrogen bond counts
- bunch_of_vectors.dat - Normalized pulling vectors

---

7. Visualization

The toolkit generates:

- `vmd_script.tcl` for visualizing pulling vectors in VMD
- Matplotlib plots of analysis results organized by pulling direction:
  - Force vs Time with mean ± standard deviation across replicates
  - Force vs Distance with confidence intervals
  - Hydrogen bond dynamics with error bars

Example VMD visualization:

```tcl
source vmd_script.tcl
```

---

8. Case Studies

Demonstrated Applications:

1. SARS-CoV-2 Spike-ACE2 Complex
   - Anisotropic mechanical stability
   - Mutation effects on binding strength
2. Kir6.1/Kir6.2 Channels (Supplementary)
   - Isoform-specific ATP binding mechanics
3. SUR2B-KNt System (Supplementary)

    - Direction-dependent release of disordered regions

---

9. Best Practices

Statistical Considerations:

- Minimum 3 replicates recommended for statistical significance
- 5+ replicates recommended for publication-quality results
- Use identical simulation parameters across all replicates

Template File Requirements:

- NAMD and GROMACS Template: Must include SMD and constraints sections, use NVT ensemble
- Run Script for NAMD: Use `INPF` and `OUTF` variables for input/output files

Selection Guidelines:

- Use MDAnalysis selection syntax enclosed in quotes
- Test selections using MDAnalysis before full simulations

Parameter Files:

- For multiple CHARMM parameter files, create `toppar.zip` archive
- Always use NVT ensemble for SMD simulations
- Validate template files before generation

For questions, bug reports, or feature requests, please contact:
Katarzyna Walczewska-Szewc, kszewc@umk.pl