SoftwareX 30 (2025) 102118

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Gromologist: A GROMACS-oriented utility library for structure and topology manipulation

Miłosz Wieczór [a,b,*], Jacek Czub [b], Modesto Orozco [a,c]

[a] Molecular Modeling and Bioinformatics, IRB Barcelona, Spain
[b] Department of Physical Chemistry, Gdańsk University of Technology, Poland
[c] Faculty of Biology, University of Barcelona, Spain

ARTICLE INFO

Keywords:
Molecular dynamics
GROMACS
Molecular modeling

ABSTRACT

Despite the increasing automation of workflows for the preparation of systems for molecular dynamics simulations, the custom editing of molecular topologies to accommodate non-standard modifications remains a daunting task even for experienced users. To alleviate this issue, we created Gromologist, a utility library that provides the simulation community with a toolbox of primitive operations, as well as useful repetitive procedures identified during years of research. The library has been developed in response to users' feedback, and will continue to grow to include more use cases, thorough automatic testing and support for a broader spectrum of rare features. The program is available at gitlab.com/KomBioMol/gromologist and via Python's pip.

Code metadata

Current code version	0.321
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-24-00630
Permanent link to Reproducible Capsule	github.com/milosz-wieczor/gromologist-for-review (code) or gitlab.com/milafternoon/gml-0.320-capsule (venv)
Legal Code License	GPL 3.0
Code versioning system used	git
Software code languages, tools, and services used	python
Compilation requirements, operating environments	numpy
If available Link to developer documentation/manual	https://github.com/milosz-wieczor/gromologist-for-review/blob/master/README.md
Support email for questions	milosz.wieczor@irbbarcelona.org or milosz.wieczor@pg.edu.pl

1. Motivation and significance

Molecular dynamics (MD) simulations are a cornerstone of modern-day computational chemistry and biology, and the last 4 decades saw the development of a range of general-purpose codes to make simulations more reproducible and accessible to the community [1–7]. Although the preparation of typical biomacromolecular systems, e.g. proteins, lipid membranes or nucleic acids containing standard residues, has become extremely streamlined and automated [8–10], the corresponding topology files – containing a full description of the system's connectivity and energetics – can be far from human-readable, and thus not amenable to simple manual manipulation. As a result, any non-automated departure from these standards, such as removing atoms or adding bonds, can become a painstakingly complex process, in particular for novice users not aware of nuances and caveats of these fixed file formats.

To deal with these routine but non-standard tasks, high-level utility libraries are needed that can accommodate the increasing complexity of inputs and dynamically respond to popular requests from the community. One notable example of such a library, Parmed, has become a stable element of the Amber ecosystem, and contains many utilities specific to OpenMM [11], but its primary goal is to provide easy access to general properties and allow for smooth interconversion between standard file formats. Another one, pmx, focuses on the preparation

* Corresponding author at: Molecular Modeling and Bioinformatics, IRB Barcelona, Spain.
E-mail address: milosz.wieczor@irbbarcelona.org (Miłosz Wieczór).

https://doi.org/10.1016/j.softx.2025.102118
Received 21 December 2024; Received in revised form 26 February 2025; Accepted 27 February 2025
Available online 12 March 2025

of hybrid topologies for "alchemical" simulations and the analysis of thus obtained results [12]. Many routine tasks, though, are either not covered by any popular library or missing in their respective documentations, leaving plenty of room for new developments.

In an attempt to fill that gap, we present Gromologist, a Python utility library oriented at custom manipulation of GROMACS topologies, and a platform for the easy implementation of new protocols. For many years now, GROMACS has been not only the most broadly used but also one of the fastest-growing MD engines in the field of molecular simulations [1,13] and has a well-established user base, which – combined with the ease of format interconversion allowed by other tools – makes Gromologist a broadly applicable tool for all types of non-standard operations including, but not limited to, adding and removing atoms and bonds, introducing amino acid mutations in both the structure and topology, checking and listing force field parameter compatibilities, looking for mismatches between structures and topologies, merging files, manipulating alchemical states, modifying force field parameters or combination rules, parameter optimization, or automated editing of structures. In addition, we keep a minimal number of dependencies (except for a few very specific features), so that the library is conveniently lightweight and can be easily installed with any modern Python distributions without causing dependency conflicts.

2. Software description

2.1. Software architecture

In molecular mechanics, topology files contain (a) a list of all numerical parameters and factors required for the functional forms used by the force field, (b) molecule definitions including atom specifications, bonding patterns and additional restraints, and (c) a list of system components. The non-trivial combination of these three components defines the energy function that is then used to drive molecular mechanics. To handle topology files in GROMACS (`.top`/`.itp`), Gromologist implements a hierarchical representation of sections, subsections, and entries. Sections represent high-level abstractions, such as individual molecule definitions, force field parameter sets or headers/footers. Subsections correspond to individual fragments of the topology file, delimited with the `[ subsection_name ]` syntax, and directly handle most operations. Entries correspond to individual lines representing e.g. individual parameters or bonded terms, and provide a low-level access to all key features of the contents of a topology.

In turn, the structure module deals with collections of atoms as points in 3D space, complementing the static topological information with (potentially dynamic) molecular geometry, and organizing sets of atoms into chains and residues. Gromologist can work with both PDB and GRO files, interconverting between them when necessary; following the introduction of AlphaFold3 [14], basic support for CIF files is also included. Hierarchical abstraction layers include Traj (ordered sets of structures), Pdb (single structures), Residue (structural building blocks) and Atom (point particles representing atomic nuclei). XYZ files, common in quantum chemical calculations, are also supported to facilitate QM-centric workflows.

Internally, Gromologist implements a robust selection language akin to that of VMD [15], allowing for logical operations, selections "within X of Y", and a number of predefined molecule classes such as "protein", "dna", "rna" or "solvent". The library also provides wrappers for certain common operations (e.g. interaction energy calculations), and contains several independent modules designed to facilitate complex tasks performed with GROMACS (such as preparation of topologies for solute tempering simulations).

As discussed below, several modules are also dedicated to interact with residue templates (`.rtp` files), or pre-defined building blocks from which complex polymers can be built. This allows for integration of force field upgrades, custom ligand parametrizations, or covalent modifications of standard residues.

2.2. Software functionalities

Splitting and merging. By default, topologies are saved as a single standalone file, including all .itp files referenced therein, but splitting to individual .itp is also supported. If needed, files can be made lightweight by keeping (a) only molecules listed in the `[ system ]` subsection and/or (b) only parameters used by the system.

Top-class utilities. The `gml.Top` class contains a number of global utilities that allow for creation of topologies for selected subsystems; updating the topology with parameters or molecules read from other files; adding standard features (position restraints, hydrogen mass repartitioning, scaled parameters) without regenerating the topology; or resolving `#define` directives.

Bonds and atoms. Primitive operations on topologies include adding, removing or swapping atoms in molecules while preserving a correct numbering scheme across all sections. Adding or removing bonds is also supported, including new bonds between two molecules, in which case the two become merged into a single molecule entry, and all new bonded terms (pairs, angles, dihedrals) are automatically identified and added, while existing ones are preserved and renumbered correctly. This allows for easy creation of chemical adducts or cyclic molecules, as well as editing of protonation states or simple chemical modifications without the need of extending the basic residue library (.rtp/.hdb files). Another heavily requested feature was the automated introduction of "special bonds" – disulfides and transition metal coordination bonds – that are not always correctly guessed by default GROMACS tools. Protein chains can be split as in proteolytic processing, introducing new N- and C-termini.

Mutations. In this vein, a mutations module allows to introduce standard amino acid mutations to the topology without the need to go through the full process of topology generation. This has proven particularly useful in the study of heavily glycosylated systems, such as the SARS-CoV-2 Spike [16], where ready-made topologies and systems were often shared publicly but introducing even minor changes (such as a point mutation) would entail going through the same complicated server-based pipeline, risking introducing additional errors at this stage. This idea is also extended to protonation states, so that individual titratable residues can be protonated or deprotonated easily and in a consistent manner. As long as standard atom name conventions are followed, the mutation module is force-field agnostic, and the user can point to a specific .rtp file containing the respective residue definitions.

Alchemistry. Another set of utilities is related to the management of alchemical states. Gromologist can add or remove selected alchemical states, or swap them – a useful feature e.g. for the equilibration of B-states that helps avoid the computational overhead associated with the use of the free energy code. It can also create alchemical topologies for standard protonated amino acids, facilitating the calculation of pK_a through alchemical methods.

Force fields. Some further features are meant to facilitate force field development. These include the possibility of automatically cloning types along with all their bonded interaction parameters and renaming all selected atoms to the new type, or by automatically adding NBFIX-type modifications just by specifying the deviation from the standard Lorentz–Berthelot rules [17,18]. Users can make use of the large library of Amber parameter sets to read .frcmod files containing specific modifications into GROMACS topologies. Externally obtained .itp files can be easily converted to force field templates (both .rtp and .hdb). An additional module implements a generic strategy for multiple dihedral fitting to QM data, akin to that available in the FFTK module of VMD for NAMD topologies [19].

Structure manipulation. Common operations include constructing new atoms from existing ones, interpolating between structures, atom-/residue renumbering according to custom rules, chain permutations or generation of virtual sites, e.g. upgrading 3-point water molecules to 4-point ones. While there are some versatile structure manipulation libraries and suites out there, having a basic set of operations implemented internally and without external dependences ensures consistency and a smooth user experience.

In addition, many often-required features of the PDB files can be filled in automatically. Chain assignment, an information typically lost in PDB to GRO conversion can be inferred based on a simple distance criterion, similarly to the CONECT entries required by some analysis programs. Elements can be inferred from atom names, and beta-factors can be set to reflect external data sets, including an option to smooth the values out spatially.

Checking and printing. Another area of applications of Gromologist covers extracting information about the topologies and structures, or making it better visible to the user. On the most basic level, one can list molecules in the system, atoms in the molecule, as well as selected bonded terms (bonds, 1–4 pairs, angles, dihedrals) using atom names or atom types. Charges and masses of both the whole system and individual molecules are easy-to-access attributes. For a protein or a nucleic acid structure, the sequence can be printed chain by chain, and per-residue missing atoms identified, helping identify cases in which residues have to be rebuilt. Gradually diagnostic functions are being incorporated to quickly identify structural issues prior to simulation, like one currently allowing for validation of all chiral centers in a protein.

When required, Gromologist can explicitly list atom names in the comments of bonded interaction entries to facilitate debugging, and numerical values of the parameters can be explicitly included in the topology. Similarly, fields set using the `#define` syntax can be set to their explicit values. Parameters whose numerical definitions are missing can also be identified with a single function call, and – if desired – filled by analogy if a dictionary of type similaries is provided.

GROMACS utilities. Gromologist provides wrappers common operations such as extracting energy terms from a trajectory, preparing a solvated system from a plain PDB file, or checking if two topologies yield the same energies on a single or multiple geometries. It also allows for easy generation of standard run inputs (.mdp) and index (.ndx) files, and provides a simple lightweight API for the creation of custom applications and workflows (see paragraph below). Solvated and energy-minimized systems can also be rapidly prepared.

Additional utilities. Example independent modules that streamline complex tasks performed with GROMACS include: launching custom non-equilibrium free energy calculations [20], generating topologies for solute tempering replica exchange (REST2) [21], or sensitivity analysis of force field terms with respect to experimental observables [22]. On top of that, Amber leaprc files can be read to generate functional `.ff` directories usable by `pdb2gmx`, a function already used to increase interoperability between teams engaged in force field development. This growing list of utilities will gradually cover more and more popular techniques and routines, making them more standardized and more available to the biophysical community. Of note, selected utilities, such as the analysis module for non-equilibrium free energy calculations, have additional dependencies, as indicated in the respective tutorials.

3. Illustrative examples

To facilitate adoption and provide ready-made recipes for common issues, a suite of step-by-step tutorials has been developed and made available at www.gitlab.com/KomBioMol/gromologist/-/wikis/Tutorials (see Fig. 1). This set of tutorials will be expanded on a rolling basis, addressing recurring questions from the community of users, and including new features as they are added.

As an example, two tutorials are reproduced below. The first tutorial illustrates how one can combine e.g. Amber protein and nucleic-acid force fields that both have their respective GROMACS ports, but have never been merged into a single `.ff` directory. Therefore, coinciding type definitions conflicting parameter modifications cannot be ruled out, so it is not safe just to merge all parameters.

A1. Divide your system into components that can be individually passed through `gmx pdb2gmx`, e.g., the protein chains in one PDB file, and nucleic-acid chains in another

```
import gromologist as gml
import os
os.mkdir('protein')
os.mkdir('rna')
p = gml.Pdb('system.pdb')
p.save_from_selection('protein', outname='protein/str.pdb')
p.save_from_selection('nucleic', outname='rna/str.pdb')
```

A2. Run `gmx pdb2gmx` sequentially on the two files, preferably in separate directories

```
cd protein
gmx pdb2gmx -f str.pdb  # choose the respective FFs
cd ../rna
gmx pdb2gmx -f str.pdb  # choose the respective FFs
cd ..
```

A3. Use Gromologist to make both topologies self-contained

```
for system in ['protein', 'rna']:
  t = gml.Top(f'{system}/topol.top')
  t.explicit_defines()
  t.add_ff_parameters()
  t.save_top(f'{system}/merged.top')
```

A4. Choose one of the generated topologies as the main one, and load the molecule(s) and type definitions from the other. Save the full topology, and check for missing parameters (there should not be any)

```
t = gml.Top('protein/merged.top')
t.add_parameters_from_file('rna/merged.top', \
  prefix_type = 'X')
t.add_molecule_from_file('rna/merged.top', \
  molcount = 1, prefix_type = 'X')
t.save_top('complete.top')
```

A5. Merge the structures so that they match the topology

```
p = gml.Pdb('protein/conf.gro')
p.insert_from('rna/conf.gro')
p.save_pdb('system.pdb')
```

As a result, `system.pdb` and `complete.top` will now contain the separately parametrized protein and RNA molecules. By analogy, other combinations of molecules (say, lipids and ligands) can be similarly prepared; moreover, it will also work if you already have a parametrized but not fully compatible topology of a system component.

The second tutorial illustrates the use of the energy decomposition module, an useful utility in e.g. calculating non-bonded interactions, or assessing the contributions of different molecular fragments to an overall interaction.

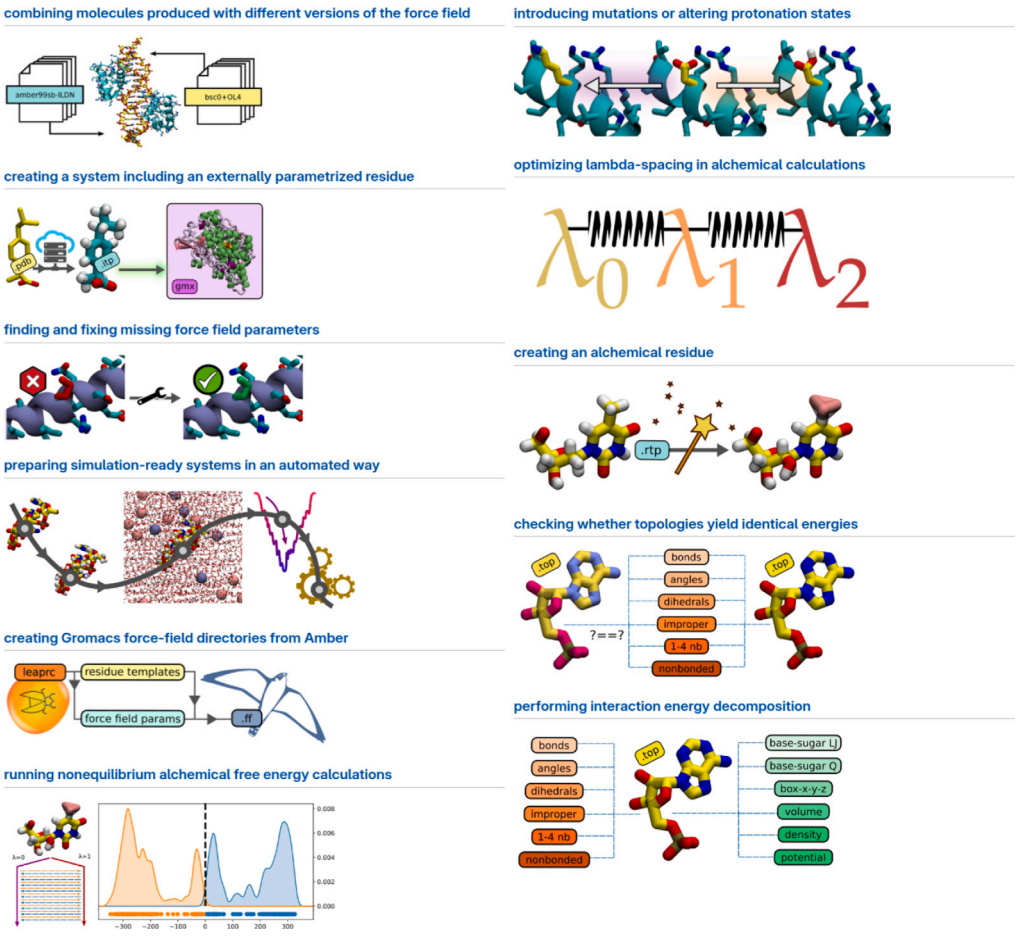

Fig. 1. A selection of step-by-step tutorials currently created for Gromologist.

B1. Prepare the files and trajectories to analyze

```
ls
dyn.xtc   merged_topology.top   box.pdb
```

B2. Verify the selections for decomposition

```
p = gml.Pdb('box.pdb')
p.get_atoms('resid 33 and chain A')
p.get_atoms('resid 5 and chain C')
# check if the printed atoms are correct
```

B3. Calculate the interaction energy

```
# to get interaction between two fragments:
ener = gml.calc_gmx_energy(struct='box.pdb', \
  topfile = 'merged_topology.top', quiet = True, \
  traj='dyn.xtc', group_a = 'resid 33 and chain A', \
  group_b = 'resid 5 and chain C')
print(ener.keys())
>>> dict_keys(['coul-sr:g1-g2', 'lj-sr:g1-g2', 'sum']
# or, to get all terms:
ener_all = gml.calc_gmx_energy(struct='box.pdb', \
  topfile = 'merged_topology.top', quiet = True, \
  traj='dyn.xtc', terms='all')
print(ener_all.keys())
>>> dict_keys(['bond', 'angle', 'proper-dih.',
  'improper-dih.', 'lj-14', 'coulomb-14', 'lj-(sr)',
  'coulomb-(sr)', 'potential', 'box-x', 'box-y', 'box-z',
  'volume', 'density'])
print(ener_all['proper-dih.'])
```

If only `group_a` is specified, the program will create a compatible subset of the system (topology and structure) and report the energetics for that subsystem only. Of note, several other tools make use of this module, including one that identifies differences between pairs of topology files (`gml.compare_topologies_by_energy`) or calculates forces acting on atoms (`gml.analytical_forces` and `gml.numerical_forces`).

4. Impact

Given the existing capabilities, Gromologist is becoming an easily extensible platform for introducing further convenience functions, editing tools, and workflows. The gradual development of the platform is benefitting the GROMACS community through quick implementation of utilities tailored to new or popular protocols, and the ongoing collaboration with the GROMACS development team helps make the best use of existing features. Direct feature requests from users also routinely contribute to expanding the scope of functionalities, with feedback often provided through the GROMACS users' forum (https://gromacs.bioexcel.eu). Over time, extensive testing will be required to ensure that the library covers all features supported by GROMACS, including more cryptic ones. In the long run, however, we expect Gromologist to become a stable element of the GROMACS environment and a valuable contribution to the broad simulation community.

Examples of applications by the community so far span multiple tasks and domains and involve the conversion of recent Amber parameter sets into GROMACS [23], setting up systems with custom covalent linkages [24], facilitating energy decomposition analysis [25], or optimizing alchemical windows in thermodynamic integration calculations [26].

5. Conclusions

In this contribution, we introduced Gromologist, a library that provides a growing set of general and specialized utilities to the GROMACS users' community. It automates routine customization tasks, implements external protocols, provides tools for checking consistency and gives access to multiple properties of interest. The tool's development is driven by everyday scientific practice, but is also guided by the feedback from experts and GROMACS developers. We also provide extensive documentation and a growing set of tutorials that automate common custom workflows. As demonstrated by the selected use cases and existing literature references, Gromologist already solves research-related problems and solves users precious time by offering a reliable and easily accessible set of both higher-level and atomic operations.

CRediT authorship contribution statement

Miłosz Wieczór: Writing – original draft, Software, Project administration, Funding acquisition, Conceptualization. **Jacek Czub:** Supervision, Conceptualization. **Modesto Orozco:** Supervision, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Milosz Wieczor reports financial support was provided by European Commission. Modesto Orozco reports a relationship with Nostrum Biodiscovery SL that includes: board membership, equity or stocks, and travel reimbursement. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 894489.

References

[1] Talirz Leopold, Ghiringhelli Luca M, Smit Berend. Trends in atomistic simulation software usage [Article v1.0]. Living J Comput Mol Sci 2021;3(1):1483.
[2] Phillips James C, Hardy David J, Maia Julio DC, Stone John E, Ribeiro João V, Bernardi Rafael C, et al. Scalable molecular dynamics on CPU and GPU architectures with NAMD. J Chem Phys 2020;153(4):044130.
[3] Thompson AP, Aktulga HM, Berger R, Bolintineanu DS, Brown WM, Crozier PS, et al. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. Comp Phys Comm 2022;271:108171.
[4] Salomon-Ferrer Romelia, Case David A, Walker Ross C. An overview of the Amber biomolecular simulation package. Wiley Interdiscip Rev: Comput Mol Sci 2013;3(2):198–210.
[5] Abraham Mark James, Murtola Teemu, Schulz Roland, Páll Szilárd, Smith Jeremy C, Hess Berk, et al. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. SoftwareX 2015;1-2:19–25.
[6] Brooks BR, Brooks CL, Mackerell AD, Nilsson L, Petrella RJ, Roux B, et al. CHARMM: The biomolecular simulation program. J Comput Chem 2009;30(10):1545–614.
[7] Eastman Peter, Swails Jason, Chodera John D, McGibbon Robert T, Zhao Yutong, Beauchamp Kyle A, et al. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. PLoS Comput Biol 2017;13(7). e1005659.
[8] Suplatov Dmitry, Sharapova Yana, Švedas Vytas. EasyAmber: A comprehensive toolbox to automate the molecular dynamics simulation of proteins. J Bioinform Comput Biol 2020;18(6).
[9] Andrio Pau, Hospital Adam, Conejero Javier, Jordá Luis, Del Pino Marc, Codo Laia, et al. BioExcel Building Blocks, a software library for interoperable biomolecular simulation workflows. Sci Data 2019;6(1):1–8.
[10] Jo Sunhwan, Cheng Xi, Lee Jumin, Kim Seonghoon, Park Sang Jun, Patel Dhilon S, et al. CHARMM-GUI 10 years for biomolecular modeling and simulation. J Comput Chem 2017;38(15):1114–24.
[11] Shirts Michael R, Klein Christoph, Swails Jason M, Yin Jian, Gilson Michael K, Mobley David L, et al. Lessons learned from comparing molecular dynamics engines on the SAMPL5 dataset. J Comput Aided Mol Des 2017;31(1):147–61.
[12] Gapsys Vytautas, Michielssens Servaas, Seeliger Daniel, De Groot Bert L. pmx: Automated protein structure and topology generation for alchemical perturbations. J Comput Chem 2015;36(5):348–54.
[13] Ribeiro João V, Bernardi Rafael C, Rudack Till, Stone John E, Phillips James C, Freddolino Peter L, et al. QwikMD - Integrative molecular dynamics toolkit for novices and experts. Sci Rep 2016;6(1):1–14.
[14] Abramson Josh, Adler Jonas, Dunger Jack, Evans Richard, Green Tim, Pritzel Alexander, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. Nature 2024;630(8016):493–500.
[15] Humphrey William, Dalke Andrew, Schulten Klaus. VMD: Visual molecular dynamics. J Mol Graph 1996;14(1):33–8.
[16] CHARMM-GUI COVID-19 archive. 2022, https://charmm-gui.org/?doc=archive&lib=covid19. [Accessed 22 March 2022].
[17] Yoo Jejoong, Aksimentiev Aleksei. New tricks for old dogs: Improving the accuracy of biomolecular force fields by pair-specific corrections to non-bonded interactions. Phys Chem Chem Phys 2018;20(13):8432–49.
[18] Luo Yun, Roux Benoît. Simulation of osmotic pressure in concentrated aqueous salt solutions. J Phys Chem Lett 2010;1(1):183–9.
[19] Mayne Christopher G, Saam Jan, Schulten Klaus, Tajkhorshid Emad, Gumbart James C. Rapid parameterization of small molecules using the force field toolkit. J Comput Chem 2013;34(32):2757–70.
[20] Crooks Gavin E. Entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. Phys Rev E - Stat Phys Plasmas Fluids Relat Interdiscip Top 1999;60:2721–6, 9.
[21] Wang Lingle, Friesner Richard A, Berne BJ. Replica exchange with solute scaling: A more efficient version of replica exchange with solute tempering (REST2). J Phys Chem B 2011;115(30):9431–8.
[22] Zhu Sheng Bai, Wong Chung F. Sensitivity analysis of water thermodynamics. J Chem Phys 1998;98:8892, 8.
[23] Miletić Vedran, osz Wieczór Mił, Rampp Markus, Kutzner Carsten, de Groot Bert L. Vytautas Gapsys Bringing the last decade of AMBER force field improvements to GROMACS. Cecam55 poster, available at https://hackmd.io/USZM8_OsQ2yP8ZiGI6RF1w?view.
[24] Wu Jiangbo, Gu Zhaoyi, Modica Justin A, Chen Sijia, Mrksich Milan, Voth Gregory A. Megamolecule self-assembly networks: A combined computational and experimental design strategy. J Am Chem Soc 2024;146(44):30553–64.
[25] Garavís Miguel, Edwards Patrick JB, Serrano-Chacón Israel, Doluca Osman, Filichev Vyacheslav V, González Carlos. Understanding intercalative modulation of G-rich sequence folding: solution structure of a TINA-conjugated antiparallel DNA triplex. Nucleic Acids Res 2024;52(5):2686–97.
[26] Arantes Pablo R, Chen Xiaoyu, Sinha Souvik, Saha Aakash, Patel Amun C, Sample Matthew, et al. Dimerization of the deaminase domain and locking interactions with Cas9 boost base editing efficiency in ABE8e. Nucleic Acids Res 2024;gkae1066.