Simulating 3SPN.2C in LAMMPS

Daniel M. Hinckley[1] and Juan J. de Pablo[2]

[1]Department of Chemical and Biological Engineering, University of Wisconsin-Maidon
[2]Institute for Molecular Engineering, University of Chicago

October 12, 2015

1. Introduction

3SPN.2C is a modification of the 3SPN.2 coarse-grained (CG) model of DNA[2]. As described in Ref. [1], 3SPN.2C incorporates sequence-dependent mechanical properties and intrinsic curvature in order to more effectively capture protein–DNA interactions. This documentation has two purposes: first, it explains how LAMMPS has been modified in order to simulate intrinsically curved dsDNA. Second, it explains how to generate the necessary input files in order to simulate these systems in LAMMPS. In this documentation, it is assumed that the reader is familiar with the process by which input files are generated for standard 3SPN.2 simulations. It is highly recommended that users of 3SPN.2C first perform simulations of 3SPN.2 in order to familiarize themselves with the general workflow before simulating 3SPN.2C. if you have any questions or problems, please contact Dan Hinckley (dhinckley@wisc.edu).

2. LAMMPS Modifications

LAMMPS does not natively support bonded parameters that are a function of sequence; instead, LAMMPS assigns a type to every bond, angle, and dihedral. Ostensibly, a unique type can be defined for every bonded interaction in order to effectively capture the variation of equilibrium bond lengths or angles with sequence. Alternatively, the pair style `list` found in `USER-MISC/` demonstrates that such behavior can be captured by reading a file containing a list of equilibrium distances and angles. 3SPN.2C adopts this appoach, namely list files, in order to capture the sequence-dependence that characterizes real dsDNA.

In 3SPN.2C `list` styles are created for the bonds, angles, and dihedrals. These styles borrow heavily from the philosophy of `pair_list.cpp` in that they read a list of interactions (e.g. bonds, angles, or dihedrals); however, the list files are formatted differently than the files used by pair style `list`. In the bond, angle, and dihedral `list` styles, the atom IDs are added together to provide a unique index in arrays from which the relevant values are extracted. This implementation assumes that each of the sums of atom IDs is unique; this is the case for simulations of 3SPN.2C but not necessarily for other systems. The format of each list file is explained in detail below.

A few characteristics are general to all list styles. First, like the pair style `list`, the `bond_coeff list`, `angle_coeff list`, and `dihedral_coeff list` commands take no arguments. Second, the list files do not supercede the contents of the LAMMPS data file processes by `read_data`; every bond, angle, and dihedral specified in the list files must also appear in the data file. Lastly, in all list files, indexing starts with 1 and empty lines and lines starting with # are ignored.

2.1. Bond List File.

The bond list file, `in00_bond.list`, is formated as follows:

```
<ID1> <ID2> <r0> <K2> <K3> <K4> # Bond 1
<ID1> <ID2> <r0> <K2> <K3> <K4> # Bond 2
...
```

with ID1 and ID2 being the atom IDs of the sites involved in the bond. The equilibrium bond length r_0 is provided in simulation units (Å, real units should *always* be used in 3SPN.2 simulations). The functional form of the bonded potential is exactly as the bond style `class2`, i.e.

$$E = K_2(r - r_0)^2 + K_3(r - r_0)^3 + K_4(r - r_0)^4. \tag{1}$$

This choice was made based on the use of the `class2` potential in 3SPN.2.

2.2. **Angle List File.** The angle list file, `in00_angl.list`, is formated as follows:

```
<ID1> <ID2> <theta0> <K> # Angle 1
<ID1> <ID2> <theta0> <K> # Angle 2
...
```

with ID1, ID2, and ID3 being the atom IDs of the sites involved in the angle. The equilibrium angle θ_0 is provided in degrees. The functional form of the angle potential is exactly as the angle style `harmonic`, i.e.

$$E = K_2(\theta - \theta_0)^2. \tag{2}$$

2.3. **Dihedral List File.** The bond list file, `in00_dihe.list`, is formated as follows:

```
<ID1> <ID2> <ID3> <ID4> <phi0> <Kperiodic> <Kgauss> <Sigma> <Type> # Dihedral 1
<ID1> <ID2> <ID3> <ID4> <phi0> <Kperiodic> <Kgauss> <Sigma> <Type> # Dihedral 2
...
```

with ID1, ID2, ID3, and ID4 being the atom IDs of the sites involved in the dihedral. The equilibrium dihedral angle ϕ_0 is given in degrees. The functional form of the dihedral potential depends on the `type` flag as follows:

$$E = \begin{cases} K_{\text{periodic}}\left(1 - \cos\left(\phi - \phi_0\right)\right) & \texttt{type} = 1 \\ K_{\text{periodic}}\left(1 - \cos\left(\phi - \phi_0\right)\right) - K_{\text{gauss}}\exp\left(-\frac{(\phi-\phi_0)^2}{2\sigma^2}\right) & \texttt{type} = 2 \\ K_{\text{periodic}}\left(1 - \cos\left(\phi - \phi_0\right)\right) + \sigma\left(1 - 3\cos\left(\phi - \phi_0\right)\right) & \texttt{type} = 3 \end{cases} \tag{3}$$

These functional forms were chosen based on their use in 3SPN.2, 3SPN.2C or Cafemol[3].

2.4. **Other Modifications.** Two additional modifications merit mention. First, `USER-3SPN` contains a modified version of `angle_hybrid.cpp` that permits the use of the list file as an angle style arguments when using the angle style `hybrid`. The file has only one line modified (line 228 in SVN verson 11423). Second, `USER-3SPN` contains a modified version of `pair_list.{cpp,h}`. This modified version includes a 12-10 Lennard-Jones interaction that is necessary for simulating the Cafemol Gō model for proteins[3].

2.5. **Nonbonded Interactions.** The nonbonded interactions (i.e. base stacking, base pairing, and cross-stacking) do not read list files to obtain parameters, unlike the bonded interactions. The base stacking parameters are specified in the LAMMPS configuration file (read by `read_data`) and are obtained by specifying the correct flag (1 = `bdna/curv`) when running `icnf.exe`. The base pairing and cross stacking parameters are hard-coded into `pair_3spn2.cpp` and are used in simuation by providing the `bdna/curv` argument to `3spn2` pair style.

3. List File Generation

3.1. **Sequence selection.** The first step is selecting the desired dsDNA sequence. Ths sequence should be completely complementary with no dangling ends, as intrinsic curvature and sequence-dependent mechanical properties are only well-defined for such systems. After selecting the sequence, a sequence file should be created formatted as follows:

```
<number of base pairs>
<sense sequence (5'->3')>
```

for example,

```
32
ATACAAAGGTGCGAGGTTTCTATGCTCCCACG
```

This sequence file will be read by `make_bp_params.py` and `icnf.exe`.

3.2. **Generating an All-Atom Topology.** The next step is to use X3DNA to generate an all-atom representation of the desired sequence. If not already installed, it should be installed by following the instructions on `http://x3dna.org`. X3DNA reads a file containing the base pair and base step parameters. The python script `make_bp_params.py` takes one argument, the sequence file created in the previous section, and generates the file `bp_step.par`. The parameters contained therein are from Morozov *et al.*[5] and Olson *et al.*[6] The basis set for B-DNA is the copied to the current directory and the `rebuild` X3DNA is used to build an atomistic representation of the dsDNA in .pdb format. (Please do not contact us with any questions regarding the use of X3DNA. Such questions can be answered by referring to the online documentation of X3DNA.)

3.3. **Mapping to a Coarse-Grained Representation.** After generating the .pdb file, it is the necessary to convert the representation to the 3SPN topology. The python script `pdb2cg_dna.py` does this for us. It takes the name of the atomistic .pdb file as its only command line argument and prints .xyz and .psf file for visualization of the curved dsDNA. It also prints a file named `dna_conf.in` which contains the Atom information for the curved dsDNA. This information is copied into a complete LAMMPS data file as described in the following section.

3.4. **Making the LAMMPS input files.** The configuration generator contained inside `DSIM_ICNF/`, `icnf.exe`, also generates .xyz and .psf files for visualization; more importantly, it generates a LAMMPS data file (`conf_lammps.in`) for straight dsDNA. In order to convert to curved ds-DNA, the `Atoms` section of `conf_lammps.in` is replaced by the contents of `dna_conf.in` using `replace_atoms.sh`. After so doing, we are now ready to generate the needed list files (`in00_bond.list`, `in00_angl.list`,`in00_dihe.list`). The python script `make_list_files.py` reads `conf_lammps.in` and prints these files. These files are used at run time when performing the LAMMPS simulation.

3.5. **Summary.** The work flow for generating the 3SPN.2C input files is presented in Fig. 1. In summary, `make_bp_params.py` is used to generate a file containing the base pair and base step parameters. X3DNA[4] is used to create an all-atom representation of the desired dsDNA system. Then `pdb2cg_dna.py` is used to map to a coarse-grained analog of the system. Lastly, `make_list_files.py` measures the equilibrium bond lengths, bend angles, and dihedral angles and prints out the list files. The individual steps are described in detail in the following section.

The commands are summarized below, assuming that the sequence file is aptly named `sequence`.

```
python make_bp_params.py sequence
x3dna_utils cp_std BDNA
rebuild -atomic bp_step.par atomistic.pdb
python pdb2cg_dna.py atomistic.pdb
icnf.exe sequence 1 1 . 0
replace_atoms.sh conf_lammps.in dna_conf.in bdna_curv_conf.in
python make_list_files.py bdna_curv_conf.in
```

These commands are all contained with the script `build_bdna_curv.sh` found in `utils/`.

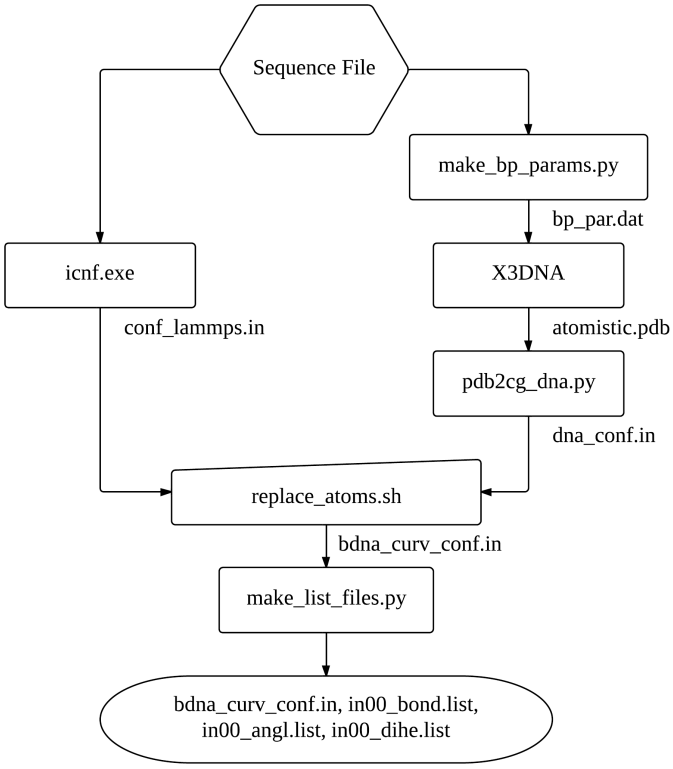

FIGURE 1. Schematic representation of the steps and tools needed to generate the required list files to simulate 3SPN.2C in LAMMPS.

4. Running in LAMMPS

The directory `bdna_curv/` inside `examples/` provides sample input and output of a simulation performed with 3SPN.2C. Please note the syntax in the specification of the bond, angle and dihedral styles. Also note that the first arguments for the pair style `3spn2` is `bdna/curv`; this argument ensures that the correct values for base pairing and cross-stacking interactions are used.

Lastly, simulations with DNA and protein deserve special mention. The list files should be generated for naked dsDNA, as described previously. Once generated, they can be used to model the equilibrium structure of 3SPN.2 DNA that is bound to a protein. This has been done successfully when modeling nucleosome assemblies [citation forthcoming]. The bound DNA should not be used to generate the list files; doing so would create a representation with equilibrium bonded interactions corresponding to the deformed DNA in contact with the protein. This means that from a mechanical perspective the dsDNA would not be experienced deformation which is clearly unphysical.

References

[1] Gordon S. Freeman, Daniel M. Hinckley, Joshua P. Lequieu, Jonathan K. Whitmer, and Juan J. de Pablo. *arXiv preprint arXiv:1404.7726*, 2014.

[2] Daniel M. Hinckley, Gordon S. Freeman, Jonathan K. Whitmer, and Juan J. de Pablo. *J. Chem. Phys.*, 139:144903, 2013.

[3] Hiroo Kenzaki, Nobuyasu Koga, Naoto Hori, Ryo Kanada, Wenfei Li, Kei-ichi Okazaki, Xin-Qiu Yao, and Shoji Takada. *J. Chem. Theory Comput.*, 7(6):1979–1989, 2011.

[4] Xiang-Jun Lu and Wilma K. Olson. *Nucleic Acids Res.*, 31(17):5108–5121, 2003.

[5] Alexandre V. Morozov, Karissa Fortney, Daria A. Gaykalova, Vasily M. Studitsky, Jonathan Widom, and Eric D. Siggia. *Nucleic Acids Res.*, 37(14):4707–4722, 2009.

[6] W. K. Olson, A. V. Colasanti, Y. Li, W. Ge, G. Zheng, and V. B. Zhurkin. *Computational Studies of RNA and DNA*, volume 2, chapter 14. Springer, 2006.