The following instructions document the procedure to generate, simulate, and analyze simulations that were originally presented in Lin et al, Nature Communications 2015, and first released for open use by Rim et al, ACS Biomaterials Science and Engineering. Full citations for these works can be found below. This repository is managed by Sinan Keten’s Computational Nanodynamics Laboratory at Northwestern University.

1. Lin, S., Ryu, S., Tokareva, O. *et al.* Predictive modelling-based design and experiments for synthesis and spinning of bioinspired silk fibres. *Nat Commun* **6**, 6892 (2015). https://doi.org/10.1038/ncomms7892
2. Nae-Gyune Rim, Erin G. Roberts, Davoud Ebrahimi *et al.* Predicting Silk Fiber Mechanical Properties through Multiscale Simulation and Protein Design, *ACS Biomaterials Science & Engineering* **2017** *3* (8), 1542-1556 DOI: 10.1021/acsbiomaterials.7b00292

Notable changes that were made from the original manuscripts:

1. “Generate\_Configuration.m” was modified to include the capability of adding ‘sticky’ terminal ends, introducing a 6th bead type to the system. The file was renamed “Generate\_Configuration\_Sticky.m.”
2. The executable, “network\_noprint,” made available in supplementary information of Rim et al. was replaced with “network\_noprint.py.” This script was written as a direct replacement after running into errors using the original “network\_noprint” executable to analyze higher molecular weight chains.
3. The LAMMPS input file “equil\_shear\_stretch\_sticky.in” was modified from equil\_shear\_stretch.in to run the LAMMPS simulation with terminal ends.

Software Packages Used:

LAMMPS: Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) is a molecular dynamics program from Sandia National Laboratories. The LAMMPS software and its installation manual can be found at http://lammps.sandia.gov/. Here we assume software is installed under the Linux environment. We used the LAMMPS, November 17th, 2016 version.

OCTAVE: Octave is a free open-source programming language primarily intended for numerical computations. It is compatible with MATLAB and can be downloaded from (<http://www.gnu.org/software/octave/download.html>). In this paper, we assume software is installed under the Linux environment.

VMD: Visual Molecular Dynamics is a molecular visualization program for displaying, animating, and analyzing large biomolecular systems using 3-D graphics and built-in scripting. VMD supports computers running MacOS X, Unix, or Windows, is distributed free of charge, and includes source code. <https://www.ks.uiuc.edu/Research/vmd/>

LAMMPS Setup:

Download the LAMMPS tarball for the version released on November 17, 2016 from the LAMMPS website (<https://download.lammps.org/tars/>). Expand the archive in an appropriate location for your new project. Navigate into the directory “lammps-17Nov2016/src” and overwrite “pair\_soft.cpp” using the “pair\_soft\_modified.cpp” filed provided in the directory “DPD-Silk-Analysis-Package/Run\_Simulation.” Compile LAMMPS using make instructions rather than cmake (https://docs.lammps.org/Build\_make.html). Note that the mpi software package is necessary to compile “lmp\_mpi.”

Octave Setup:

Install octave packages per the procedure detailed in <http://www.gnu.org/software/octave/download.html>.

VMD Setup:

Install VMD Version 1.9.3 found at <https://www.ks.uiuc.edu/Development/Download/download.cgi?PackageName=VMD>.

Procedure:

See File Description section for details about each file.

Running a Sample Simulation – Use files in directory ‘Simulation\_Run’

1. Generate initial configurations of randomly distributed peptide chains in a water box in the form of “.psf” and LAMMPS “.data” using “Generate\_Configuration\_Sticky.m”. See file description for more details.
2. In the LAMMPS input file, the name of the data file generated in the previous step should be modified (inside the “#file name” section).
3. In the file submit.sh, change the variable lmp to the path to your compiled lammps executable. Change the mpirun command as necessary to specify number of processors (-np) and name of lammps log file (-log). Change or remove the options specified in lines beginning with #SBATCH as necessary. Current #SBATCH options are compatible with Northwestern Quest Computing Cluster (<https://www.it.northwestern.edu/departments/it-services-support/research/computing/quest/>). This step will generate the following “dcd” trajectory and stress data files: equil\_11111.dcd, equil\_11111\_unwrap.dcd, shear\_1111.dcd, shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111.dcd, equil\_after\_shear\_11111\_unwrap.dcd, stretch\_11111.dcd, stretch\_11111\_unwrap.dcd, all\_stress\_11111.txt.
4. Once the simulation has completed, open VMD and run the command ‘source view.vmd’ to generate a visually appealing representation of the simulation.

Analyzing a Sample Simulation – Use files in directory ‘Simulation\_Analysis’

1. Copy the output files from the LAMMPS simulation into the directory ‘DPD-Silk-Analysis-Package/Simulation\_Analysis” (equil\_11111.dcd, equil\_11111\_unwrap.dcd, shear\_11111.dcd, shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111.dcd, equil\_after\_shear\_11111\_unwrap.dcd, stretch\_11111.dcd, stretch\_11111\_unwrap.dcd, all\_stress\_11111.txt).
2. Copy the psf file without water (the file with the “protein\_only.psf” extension) as ref.psf into this directory.
3. Copy the LAMMPS data file as ref.data into this directory.
4. Open VMD and run the tcl script “make\_refpdb.tcl” using the command ‘source make\_refpdb.tcl’. This will write a file called ref.pdb, which will be used in the following analysis scripts.
5. In “individual\_frame\_coordinate.sh,” change each loop to specify the number of frames to be analyzed from each dcd. This is equal to the number of frames output by each lammps production run (equil, shear, stretch) divided by the -stride specified in the ./catdcd command. Change the variables Nequil\_shot, Nshear\_shot, and Nstretch\_shot in “separate\_coordinate\_single.m” to match the number of frames to be analyzed. Update the names of each dcd file to reflect the random seed chosen for the simulation.
6. Run the bash script: ‘individual\_frame\_coordinates.sh’. In the first three lines (starting with “catdcd”) from the configurations saved as “dcd” file, -stride 2 means that every other frame is saved into “pdb” format as coord.pdb for further analysis. For instance, if there are 14 configurations generated in the equilibrium part of the example, 7 configurations will be saved in the “pdb” format and analyzed inside the first loop.
7. In ‘get\_connectivity.m’ change MAX\_peptide\_repeat such that it is greater than the maximum number of peptide repeats.
8. In ‘individual\_frame\_network\_analysis.sh’ change each loop to specify the number of frames to be analyzed from each dcd as was done for ‘individual\_frame\_coordinates.sh’.
9. Run the bash script: ‘individual\_frame\_network\_analysis.sh’
10. In ‘run\_all\_node\_bridge\_multiplicity.sh,’ change each loop to specify the number of frames to be analyzed from each dcd. This is equal to the number of frames output by each lammps production run (equil, shear, stretch) divided by the -stride specified in the ./catdcd command.
11. To plot the node-bridge diagram, run the bash script ‘run\_all\_node\_bridge\_multiplicity.sh.’
12. Generate .png files of the node-bridge diagram for each frame by running ‘collect\_all\_movie.py.’ All the resulting images will be placed in a directory called ‘movie.’
13. In ‘Connectivity\_Analysis.m’ change the variables Nequil, Nshear, and Nstretch to reflect the number of frames being analyzed. Change the variable timestep to match that specified in the LAMMPS input file. In the ‘% get equil data’ section, change the variable timestep and dumpfreq to match the equilibration run. In the ‘% get shear data’ section, change the dumpfreq and timestep to match the shear run. In the ‘% get stretch data’ section, change the dumpfreq and timestep to match the stretch run.
14. To run ‘Connectivity\_Analysis.m’, either use MATLAB or run the bash script ‘run\_connectivity\_analysis.sh’.
15. In ‘run\_all\_network\_conductance.sh’ change each loop to specify the number of frames to be analyzed from each dcd. This is equal to the number of frames output by each lammps production run (equil, shear, stretch) divided by the -stride specified in the ./catdcd command.
16. To plot the network conductance, run the bash script ‘run\_all\_network\_conductance.sh’.
17. To conduct a stress strain analysis and plot a stress strain curve, run ‘ss\_analysis.py’.

File Descriptions:

**Generate\_Configuration\_Sticky.m** - Writes ‘.psf’ files and a ‘.data’ to be used as input for a LAMMPS simulation. Simulation parameters such as density, protein volume fraction, and silk motif characteristics can be modified. Modify repeat\_motif\_spider to define motif repeating units. Modify num\_hydrophobic\_spider to define the number of ‘a’ beads in an A block. Modify num\_hydrophilic\_spider to define the number of ‘b’ beads in a B block. Modify num\_histidine\_spider to define how many ‘b’ beads in the H block. Modify num\_sticky\_spider to define how many beads to place in each terminal region. OUTPUTS: spider\_....psf, spider\_...\_protein\_only.psf, spider\_...\_evap.psf, spider\_....data

**pair\_soft\_modified.cpp** – Code to overwrite pair\_soft.cpp when compiling lammps. Pair style soft is then used to implement a nonbonded harmonic potential between ‘a’ type beads to represent harmonic bonds.

**equil\_shear\_stretch\_sticky.in** – LAMMPS input file to run equilibration, shear, post-shear equilibration, and stretch simulations which includes parameters for bead type ‘c’ sticky terminal regions. Data file name in “# file name” section must match the ‘.data’ file output by Generate\_Configuration\_Sticky.m. INPUTS: Requires spider\_....data. OUTPUTS: equil\_11111.dcd, equil\_11111\_unwrap.dcd, shear\_11111.dcd, shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111\_unwrap.dcd, stretch\_11111.dcd, stretch\_11111\_unwrap.dcd, all\_stress\_11111.txt.

**equil\_shear\_stretch.in** – LAMMPS input file to run equilibration, shear, post-shear equilibration, and stretch simulations which does not include parameters for bead type ‘c’ sticky terminal regions. Data file name in “# file name” section must match the ‘.data’ file output by Generate\_Configuration\_Sticky.m. INPUTS: Requires spider\_....data. OUTPUTS: equil\_11111.dcd, equil\_11111\_unwrap.dcd, shear\_11111.dcd, shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111\_unwrap.dcd, equil\_after\_shear\_11111\_unwrap.dcd, stretch\_11111.dcd, stretch\_11111\_unwrap.dcd, all\_stress\_11111.txt.

**view.vmd** – Generates an appealing visualization for the default simulation. Run this script in VMD using the command ‘source view.vmd’.

**submit.sh** – Bash commands necessary to run the LAMMPS simulation using mpi.

**make\_refpdb.tcl** – Writes a reference pdb file called ‘ref.pdb’ from ‘ref.data’ which is required by later analysis scripts.

**individual\_frame\_coordinates.sh** – Bash script that runs executable catdcd and separate\_coordinate\_single.m to generate an individual directory for each frame of the equilibration (equil), shear, and stretch simulation runs. Each directory will have a snapshot pdb file corresponding to that frame called ‘coord.pdb’ which is generated by running get\_coordinate.m in each directory.

**individual\_frame\_network\_analysis.sh**  - Bash script that runs network\_noprint.py and get\_connectivity.m in each directory.

**catdcd** – OUTPUTS: equil.pdb, shear.pdb, stretch.pdb

**separate\_coordinate\_single.m** – OUTPUTS: Nbead.txt, equil\_evolve\_\*/coord.pdb, shear\_evolve\_\*/coord.pdb, stretch\_evolve\_\*/coord.pdb.

**get\_coordinate.m**  - OUTPUTS: boundary\_condition.txt, Ndata.txt, coordinate.txt, coordinates.txt.

**network\_noprint.py** – OUTPUTS: cluster\_sizes.txt, clusters.txt, clusters\_resid.txt.

**get\_connectivity.m** – OUTPUTS: connectivity\_analysis.txt, connectivity\_analysis\_numeric.txt, resid\_for\_connecting\_polymer.txt, Ncluster\_of\_nodes\_sizes.txt, cluster\_coordinate.txt, link\_resid\_of\_each\_cluster.txt, links\_per\_cluster.txt, connectivity\_matrix.txt, cluster\_connectivity.txt, cluster\_angles.txt, cluster\_connectivity.net.

**run\_all\_node\_bridge\_multiplicity.sh** - Bash script that runs the matlab script ‘node\_bridge\_diagram\_multiplicity.m’ in each of the directories equil\_evolve\_\*, shear\_evolve\_\*, and stretch\_evolve\_\*.

**node\_bridge\_diagram\_multiplicity.m** – OUTPUTS: movie.png.

**collect\_all\_movie.py** – Organizes the output ‘movie.png’ files from ‘node\_bridge\_diagram\_multiplicity.m’ into a single directory called ‘movie.’

**Connectivity\_Analysis.m** – Conducts a connectivity analysis and outputs figures. OUTPUTS: ave\_size\_Crystal\_equil.png, ave\_size\_Crystal\_shear.png, ave\_size\_Crystal\_stretch.png, num\_Beta\_equil.png, num\_Beta\_shear.png, num\_Beta\_stretch.png, num\_Conn\_equil.png, num\_Conn\_shear.png, num\_Conn\_stretch.png.

**run\_connectivity\_analysis.sh** – Contains the bash commands to run ‘Connectivity\_Analysis.m’ in MATLAB without a display.

**Network\_Conductance.m** – Plot the network conductance.

**run\_all\_network\_conductance.sh** – Bash script that runs the MATLAB script ‘Network\_Conductance.m’ in each of the directories equil\_evolve\_\*, shear\_evolve\_\*, and stretch\_evolve\_\*.

**ss\_analysis.py** – Plot the effective stress versus strain and output as ss\_analysis\_out/effective\_stress.png.