readme

Our code can be used to reproduce all the figures in the main paper, and includes analytical calculations as well. While the code is commented where necessary, the comments are not exhaustive, and we cannot guarantee that the code will work if components of it are altered piecemeal without a full understanding of what is entailed. We are sharing our scientific code under the principle that sharing some is better than sharing none (CRAPL license).

Our code consists of executable Python .py scripts as well as Jupyter notebooks. To successfully be able to run the scripts, please install the following packages: numpy, matplotlib, scipy, time, sympy. A straightforward way to obtain most of these is by installing the Anaconda Python distribution (https://www.anaconda.com/products/distribution), which will also install Jupyter Notebook and the Spyder IDE. Additional packages can then be installed as required. At the time of writing, this is Anaconda for Python 3.10.

The scripts containing our simulation algorithms can be found in the subfolders of "Code_and_Data." When these .py scripts are executed, the output data are saved as .txt files in the same directory. We conducted these simulations using computational resources at ARCH (Advanced Research Computing at Hopkins), where each script was executed on a single core. Each subfolder contains a "print_output.txt" file showing the required computation time for a particular simulation, which should be representative of a typical desktop PC (~1 day for each FSBD script).

To run these scripts, navigate to the desired subfolder (named with bead sizes or adhesion strengths) and execute the .py file within using Spyder IDE. Here are a few pointers:

- To access the simulation time, search for "simtime = np.linspace(0,0.016,5000000)". To run for a shorter duration, you may decrease the maximum simulation time and number of data points, while preserving the timestep interval 'dt'. For example, np.linspace(0,0.0016,500000) would preserve the same 'dt'.
- To ensure that the simulation has successfully completed, check whether the line "The total computation time" has been printed to the console. This indicates the time taken to perform a simulation of a single bead or fluctuating membrane. If there are two beads, this line is printed twice. Figures showing Fourier-space variances are saved separately for each bead.

List of Jupyter notebooks (.ipynb) to generate the figures in the paper:

The folder "Jupyter_Notebooks" contains the .ipynb files used to post-process the simulation data and to generate all the figures in our paper. These notebooks also include analytical calculations, intermediary analyses, and a tutorial on the treatment of Fourier modes.

Assuming that the path directories for our uploaded simulation data were maintained, the following figures can be generated via “Cell → Run All” for each Jupyter notebook. This ensures that the cells are executed sequentially.

The notebooks for modeling association rates (Association_*.ipynb) require the installation of the **lmfit** package **(https://lmfit.github.io/lmfit-py/)** for parameter fitting. This must be installed explicitly, as it is not included with the Anaconda distribution.

- Fig. 4: HistogramsBeads_Simulated
- Fig. 5: SNR_BeadPairs_Simulated
- Fig. 6: FlatMembrane_CurvRhoVarianceProtSize_Simulated
- Fig. 7: Analytical_SNR
- Fig. 8: Analytical_SNR
- Fig. 9: Association_PreferredCurvatureModel
- Fig. 10: Association_ThresholdModel
- Fig. 11: Association_ThresholdModel
- Fig. 12: RelaxationFrequencies_Convergence*
- Fig. 13: MembraneBeadProfiles_Simulated
- Fig. 14: Analytical_SNR
- Fig. 15: Association_ThresholdModel_HighAdhesionLimit
- Fig. 16: Analytical_SNR
- Fig. 17: HistogramsBeads_Simulated
- Fig. 18: Analytical_SNR
- Fig. 19: Analytical_SNR

* “RelaxationFrequencies_Convergence” requires LaTeX packages to be installed to display symbolic outputs correctly.

Additional notebooks:

- FourierModes_ArrangingIndices: This notebook is a walkthrough of how our Fourier indices and modes are being arranged for the addition of thermal noise in our algorithms. (Refer to Appendix A).
- RelaxationFrequencies_Convergence: This can be used to test whether a particular set of simulation parameters satisfy the conditions required for simulation convergence. (Refer to Appendix C subsection “Choosing parameters appropriately for simulation convergence”).