MENG 255 Term Project: Smart Monte Carlo with the Rouse Model

Noah Dohrmann

(Dated: 17 March 2022)

Please see the corresponding code (simulation_term_proj.py) and the example Jupyer notebook which go along with this report.

I. INTRODUCTION AND THEORY

For my end of quarter project, I proposed that I implement a “Smart" adaptation of the MCMC method as developed by Rossky[1] and coworkers.[1] In this version of Monte Carlo particle simulation, both the trial displacements $\{\delta\vec{r}_i\}$ and the acceptance probabilities P_{acc} are modified with the intention of increasing the overall acceptance rate and more efficiently sampling the phase space. In order to test the effectiveness of this method, I will implement a Rouse model of a polymer joined by both a harmonic as well as a shifted Lennard-Jones potential and compare the resulting dynamics of the Smart Monte Carlo system with a standard Monte Carlo System.

A. Smart Monte Carlo Definition

The authors of the above paper prepare an outline of how Smart Monte Carlo is defined, which is summarized here. First, they use define their system with the Langevin equation for dynamics:

$$\ddot{\boldsymbol{r}} = -\dot{\boldsymbol{r}}/\tau + (\boldsymbol{F} + \mathscr{F})/m \tag{1}$$

for a particle of mass m and frictional coefficient τ^{-1}. The vector $\boldsymbol{F}$ is the negative gradient of the potential, and $\mathscr{F}$ is a random force. This aims to preserve the stochastic character of Standard Monte Carlo, while also incorporating information about the physical system around the selected particle according to a Brownian dynamics integration scheme. The original authors implement the following integration scheme provided by Ermak[2,3]

$$\Delta\vec{r}_i = D\beta\vec{F}_i\delta t + \vec{R} \tag{2}$$

which depends on the diffusion constant D, $\beta = \frac{1}{k_B T}$, and $\vec{F}_i$, the force experienced by particle i at the start of the current integration step. The last term on the right-hand side, $\vec{R}$, is a random 3-vector that has components R_i each drawn from a Gaussian distribution W of width A.

$$W(R_i) = (4A\pi)^{-3/2}\exp\left(-\frac{R_i^2}{4A}\right) \tag{3}$$

The parameter A may be chosen to be $A = D\delta t$ or $A = \Delta t^2\ 2m\beta$ in Rossky and coworkers,[1] among other possible options when compared with different methods such as Force-bias Monte Carlo.[10] Again, this method is different than standard Monte Carlo, in which the components of each trial displacement are only drawn from a uniform distribution over $-L$ to L.

The next foundation of smart Monte Carlo implementation are the acceptance probabilities to determine if a move from one state to the next will be accepted in the Markov Chain. These new probabilities may be written as[1]

$$P_{\text{SMC}} = \min\left(1, T^*_{ji}\pi_j / T^*_{ij}\pi_i\right) \tag{4}$$

with

$$T^*_{ij} = C\exp\left[-\frac{\left(\Delta\vec{r} - \beta A\vec{F}_i\right)^2}{4A}\right] \tag{5}$$

and

$$\pi_i = \frac{\exp\left[-\beta U_N(i)\right]}{\int \mathrm{d}(i)\exp\left[-\beta U_N(i)\right]} \tag{6}$$

An equivalent formulation that is more illuminating for the purposes of code-implementation of this method is given in Allen and Tildesley[10] for the transition from state m to state n as $P_{\text{acc}} = \min(1, \psi_{nm})$, if

$$\psi_{nm} = \exp\left[-\beta\left(\delta V_{nm} + 0.5(\vec{f_i}^{\,n} + \vec{f_i}^{\,m})\cdot\delta\vec{r_i}^{\,nm} + \delta W^{\text{ SMC}}\right)\right] \tag{7}$$

with

$$\delta W^{\text{ SMC}} = \frac{\beta A}{4}\left(\left|\delta\vec{f_i}^{\,nm}\right|^2 + 2\delta\vec{f_i}^{\,nm}\cdot\vec{f_i}^{\,m}\right) \tag{8}$$

To be clear, $\vec{f_i}^{\,nm}$ is the vector difference of the force acting on i in state n versus that acting on i in state m.

Apart from these distinct features of smart Monte Carlo (among the most computationally demanding of the new features is the calculation of the force matrix), much of the other code infrastructure is the same for initializing and propagating both standard and smart Monte Carlo Rouse model simulations.

B. Simulation Details

The following reported data are all for a single $N = 64$ Rouse model chain in which both harmonic as well as shifted Lennard-Jones potentials act between sequential members of the chain. Explicitly, the potentials are

$$V_{\text{H}}(r) = \frac{3}{2}\varepsilon r^2 \tag{9}$$

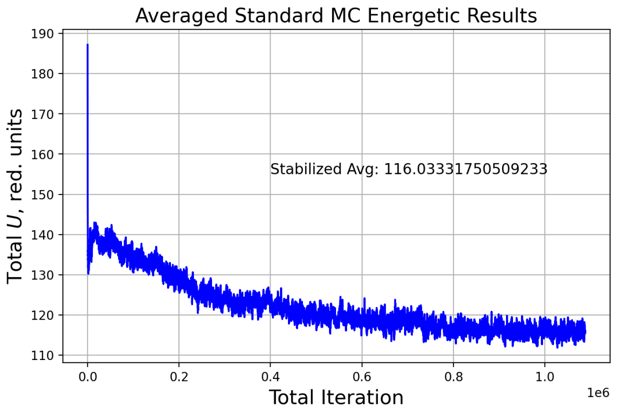

FIG. 1. Reference energy results for the standard Monte Carlo Rouse model method, $N = 64$.

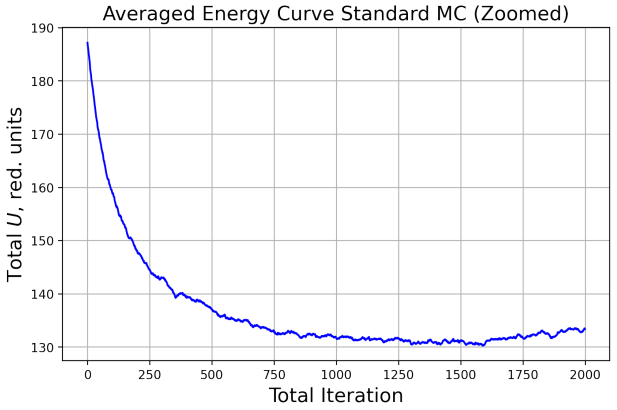

FIG. 2. Zoomed pre-convergence reference energy results for the standard Monte Carlo Rouse model method, $N = 64$.

$$V_{\mathrm{LJS}}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] + \Delta_{\mathrm{LJ}} \qquad (10)$$

with $\varepsilon = k_B T$. Unless otherwise stated, $T = 1.0$, and the above shift in the potential is found to be $\Delta_{\mathrm{LJ}} = 0.0163$.

II. RESULTS

A. Energetics and Bond Distances

The first matter of comparison between the two methods are the energetics of the chain total potential energy U_{tot} and the structural property of the average bond length squared $\langle r^2 \rangle$ between two adjoined monomers on the chain.

The reference results for the standard Monte Carlo method Rouse model are reported below for a total duration length of 1.08e6 $(17000 * 64)$ trial moves. We see in Fig. 1 that the energy converges after several hundred thousand iterations, and settles with an average value of $U_{\mathrm{tot}} = 116.03\varepsilon$. A figure of the zoomed energies to their pre-converged values at low iteration counts is also provided in Fig. 2.

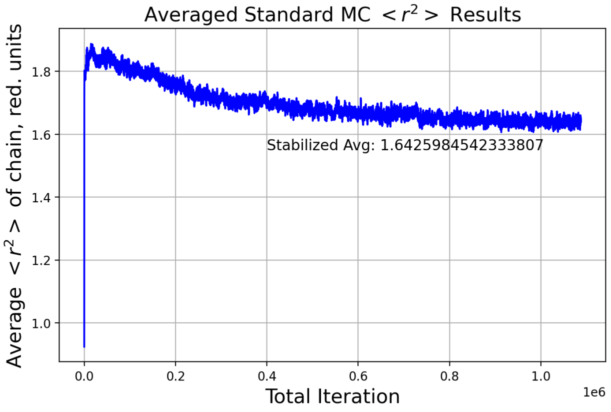

FIG. 3. Reference average bond-length squared results for the standard Monte Carlo Rouse model method, $N = 64$.

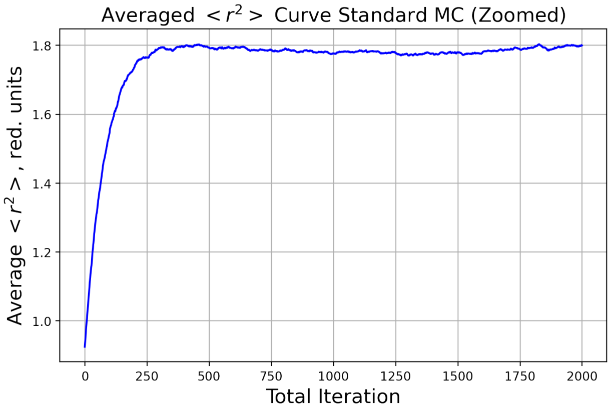

FIG. 4. Reference average bond-length squared results for the standard Monte Carlo Rouse model method, $N = 64$.

The average bond length squared is also reported, and it was found that after chain had moved enough iterations to converge, $\langle r^2 \rangle$ approached an average of $1.642\sigma^2$, seen with Fig. 3. Again, the zoomed values are presented to give higher definition detail on early behaviour of the chain in Fig. 4. As with the energies, there is rapid movement away from the less-favored structure that is generated at initialization time to the more favored, lower energy and higher bond distance structures that are seen at convergence. This indicates that standard Monte Carlo is successful in bringing this system closer to equilibrium conditions.

Moving on to the Smart Monte Carlo results, it is expected beforehand that the two simulation techniques should converge to similar energies, as they are both sampling from the same system. To compare potential energy values first, the recorded U_{tot} for the same system is presented in Fig. 5, in which energy converges top a value of 112.85ε, roughly only 2.7% lower than the standard Monte Carlo results. We see that as opposed standard MC, the energy does not solely near-monotonically decrease towards the convergence value at low iteration counts.

As the two convergence energy averages are not identical, it

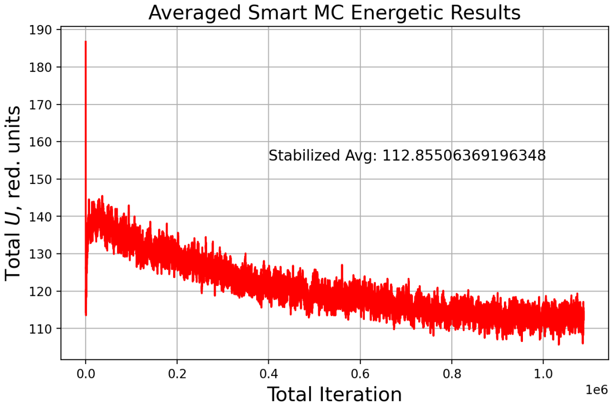

FIG. 5. Potential energy results for the smart Monte Carlo Rouse model method, $N = 64$.

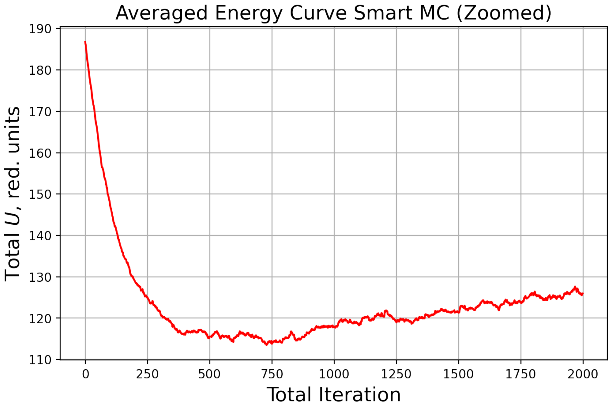

FIG. 6. Zoomed potential energy results for the smart Monte Carlo Rouse model method, $N = 64$.

is also expected that the value of $\langle r^2 \rangle$ will be numerically different as well. This is indeed what is recorded, with the Smart MC chain having an averaged bond length squared of $1.461\sigma^2$ at convergence in Fig. 7, with zoomed values found in Fig. 8. The averaged values here are very sensitive to small changes and outliers in r values for each bond, which leads $\langle r^2 \rangle$ for the two methods to have different reported values even if the reported averaged total energies of the chain appear to be quite similar.

Interestingly, there is not a substantial difference in the number of steps required to obtain these averages among the two methods. For both standard and Smart Monte Carlo, the majority of the absolute change in these quantities from their initialization values takes place within the first 500 iterations. After this, it takes several hundred thousand iterations to settle on a more stable average. The ideal results may have shown slightly faster convergences towards equilibrium as reported by Rossky[1], though there are likely other differences between the two simulations that may have caused this.

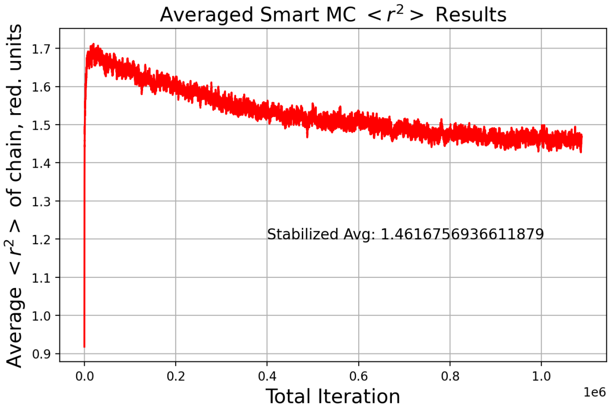

FIG. 7. Average bond-length squared results for the Smart Monte Carlo Rouse model method, $N = 64$.

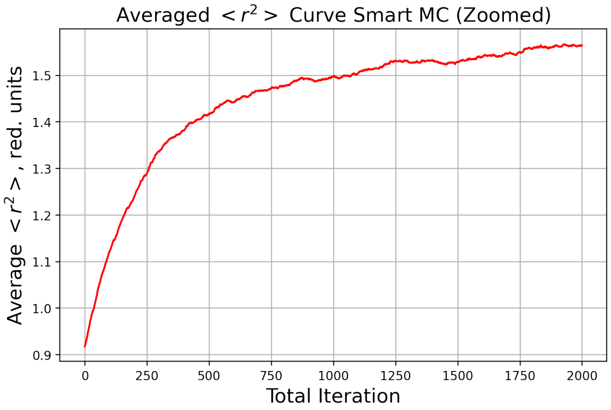

FIG. 8. Zoomed average bond-length squared results for the Smart Monte Carlo Rouse model method, $N = 64$.

B. Mean Squared Displacements

The next calculable quantities of interest are the Mean Squared Displacements (MSDs) of the chain at different points t during the simulation

$$\mathrm{MSD} = \frac{1}{N}\sum_{i=1}^{N}\left[\vec{r}_i(t) - \vec{r}_i(0)\right]^2 \tag{11}$$

Again, the reference results obtained from standard Monte Carlo will be presented first. In the first problem set of the course, it was found that the a parameter for adjusting trial displacements has an optimal value of $a = 1.0$ to obtain the greatest MSD's. The standard MSD results with this value of a are given in Fig. 9, approaching $12\,000\sigma^2$ towards the end of the simulation. As was expected, a smooth, near-monotonically increasing MSD that varies in an approximately linear fashion with the number of iterations is observed.

Now, for the Smart Monte Carlo MSD results, several values of the A parameter are tested and averaged over several runs. This value must also be treated carefully, as it is connected with the step size of the Brownian integrator. Allen and

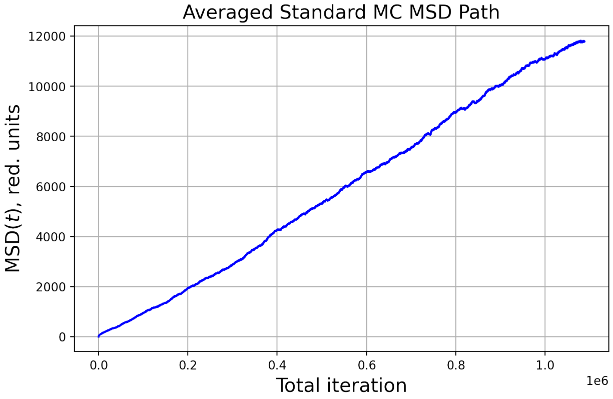

FIG. 9. Reference results for the MSD path of a standard Monte Carlo simulation, $N = 64$, $a = 1.0$.

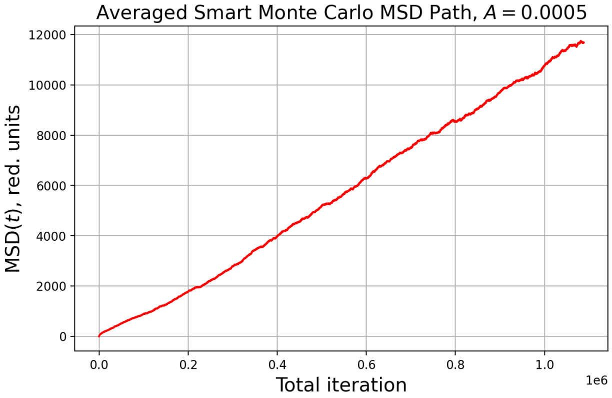

FIG. 10. Results for the MSD path of a smart Monte Carlo simulation, $N = 64$, $A = 0.0005$.

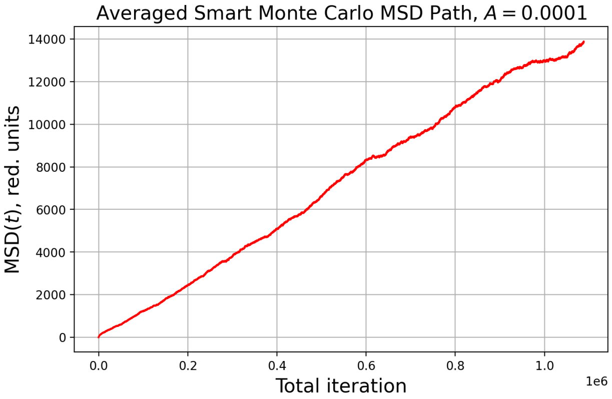

FIG. 11. Results for the MSD path of a smart Monte Carlo simulation, $N = 64$, $A = 0.0001$.

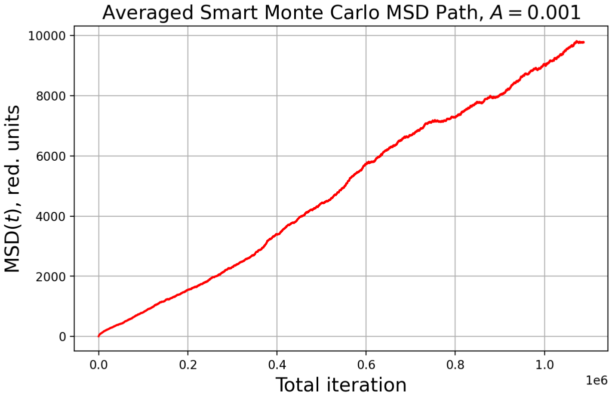

FIG. 12. Results for the MSD path of a smart Monte Carlo simulation, $N = 64$, $A = 0.001$.

Tildesley note that for small values of A, Smart Monte Carlo simulations behave very similarly to true Brownian Molecular Dynamics (MD) simulations.[10] For a trial A value of $A = 0.0005$, the MSD path is recorded in Fig. 10, where a very similar value approaching the 12 000σ^2 of the standard simulation is found. Additionally, MSD results for values such that $A \in \{0.0001, 0.001, 0.002, 0.0025\}$ are also recorded in Figures 11, 12, 13, and 14, respectively.

It is seen that when A is kept sufficiently small, the MSD values qualitatively reflect the path found from standard Monte Carlo, with the best agreement for a simulation of 1.08e6 total trial displacements where $A = 0.0005$. However, the value of a cannot be directly compared to the value for A, since they both hold somewhat different information about the physical system, the most notable difference being that a is for a pure MC simulation and as such does not reference a time step.

C. Autocorrelation of Rouse Modes

The last major quantities of interest are the Rouse modes and their autocorrelation during the simulation. As background, a chain of length N may be "split" into p subchains ($p = 1$ corresponding to the total chain), and the relaxation time of a subchain may be related to to the longest relaxation time by[4]

$$\tau_p \approx \tau_0 \left(\frac{N}{p}\right)^2 \tag{12}$$

Additionally, the coordinates of the chain $\{\vec{r}_i\}$ can be transformed into the p^{th} Rouse mode by[5]:

$$\vec{X}_0 = \sqrt{\frac{1}{N}} \sum_{i=1}^{N} \vec{r}_i \tag{13}$$

$$\vec{X}_p = \sqrt{\frac{2}{N}} \sum_{i=1}^{N} \vec{r}_i \cos\left[\frac{p\pi}{N}(i - 1/2)\right] \tag{14}$$

The autocorrelation of these modes is then accessible with

$$\left\langle \vec{X}_p(t) \cdot \vec{X}_p(0) \right\rangle = \left\langle \vec{X}_p^2 \right\rangle \exp(-t/\tau_p) \tag{15}$$

where the terms on the right-hand side are:

$$\left\langle \vec{X}_p^2 \right\rangle = \frac{\langle r^2 \rangle}{4\sin^2\left(\frac{p\pi}{2N}\right)} \tag{16}$$

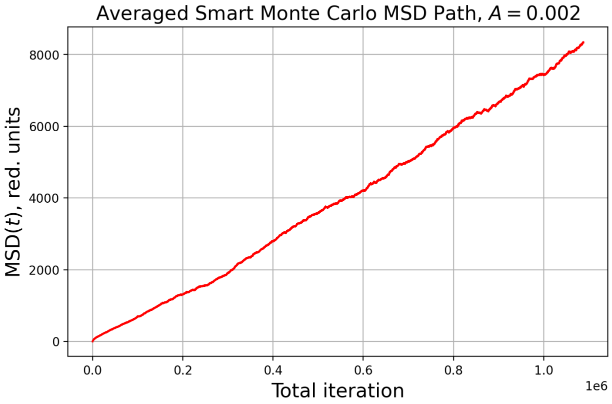

FIG. 13. Results for the MSD path of a smart Monte Carlo simulation, $N = 64$, $A = 0.002$.

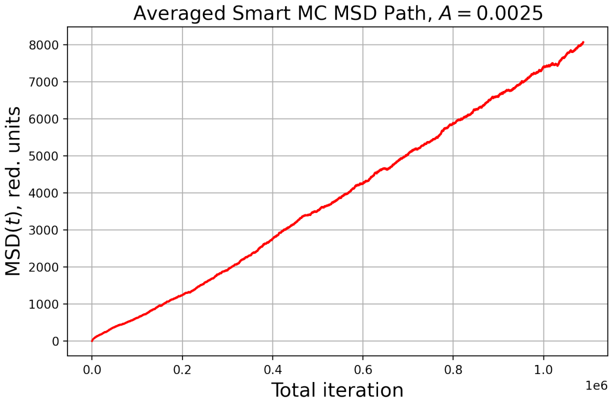

FIG. 14. Results for the MSD path of a smart Monte Carlo simulation, $N = 64$, $A = 0.0025$.

$$\tau_p^{-1} = \frac{12 k_B T}{\zeta \langle r^2 \rangle} \sin^2\left(\frac{p\pi}{2N}\right) \tag{17}$$

Here, I report the standard and Smart Monte Carlo Rouse modes ($p = 1$ to 5) side by side. Beforehand, it is expected that the two should yield similar decay paths for the mode autocorrelations, with the higher Rouse mode autocorrelations decaying faster those of the lower modes. The standard MC results are shown in Fig. 15 with the Smart MC results shown in Fig. 16. Unfortunately, both of these methods return results for the 2nd and 4th Rouse mode autocorrelations that decay at a similar rate as the first. Though, the 3rd and 5th modes comparatively decay as expected in relation to the 1st.

From the equation 12 for τ_p, it is expected that

$$\frac{\tau_p}{\tau_{p'}} = \left(\frac{p'}{p}\right)^2 \tag{18}$$

These values van be found and compared via a curve fit using equation 15 on the data collected. For standard Monte Carlo, the following table (T. I) gives the fitted values of τ_p as well as the expected and found ratios between τ_p and τ_{p+1}. The same for Smart Monte Carlo is written in T. II.

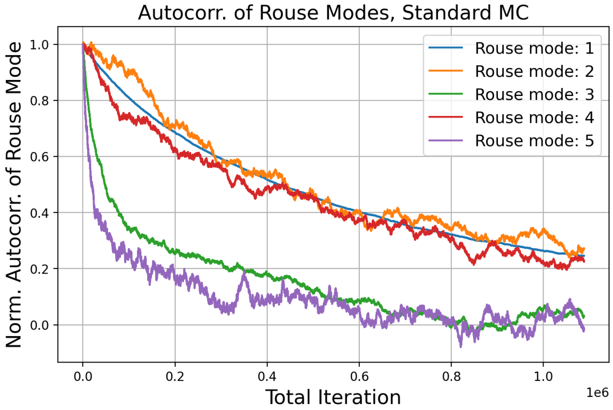

FIG. 15. Results for the averaged decay in the autocorrelation of the first five Rouse modes as found by a standard Monte Carlo simulation for the $N = 64$ chain.

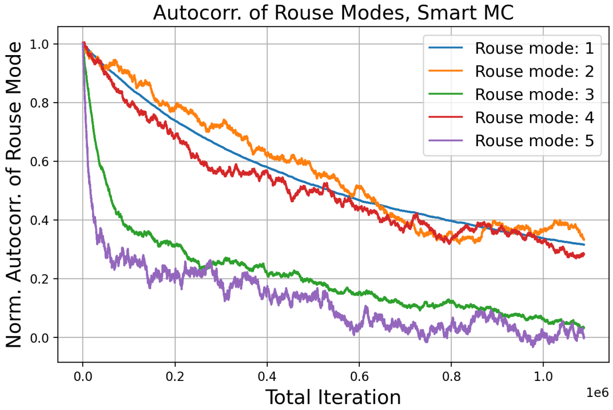

FIG. 16. Results for the averaged decay in the autocorrelation of the first five Rouse modes as found by a standard Monte Carlo simulation for the $N = 64$ chain.

Unfortunately for both methods, the numbers obtained for the τ_p's do not appear to be quantitatively accurate, even though a few of the mode decays appeared to be qualitatively accurate in comparison to one another. A remedy for this error, or what is causing it to behave in this manner, has yet to have been found for the code.

TABLE I. Standard MC Rouse Model τ_p Results

Mode	τ_p	**Expected** τ_p/τ_{p+1}	**Found** τ_p/τ_{p+1}
1	1.5889e12	4.0	7.697e6
2	206449	2.25	0.9139
3	221525	1.777	0.4464
4	495978	1.563	316.91
5	1565	N/A	N/A

TABLE II. Standard MC Rouse Model τ_p Results

Mode	τ_p	Expected τ_p/τ_{p+1}	Found τ_p/τ_{p+1}
1	1.62804e12	4.0	5.478e6
2	297174	2.25	0.8779
3	338472	1.777	0.5068
4	667788	1.563	251.4
5	1565	N/A	N/A

III. CONCLUSION

In this term project, the goal of implementing a version of the "Smart Monte Carlo" simulation method proposed by Rossky[1] and coworkers is worked towards for a Rouse model single polymer system. We see that the two methods give similar structural and energetic predictions, as well as MSD's on the same order of magnitude for small values of the simulation parameter A. When considering the Rouse modes and the exponential decay of their normalized autocorrelation functions, it is found that Modes 1, 3, and 5 all decay with respect to one another in a qualitatively cornet manner, while modes 2 and 4 are seen to erroneously decay with a similar curve as mode 1. Additionally, these decay paths impact the estimated sub chain relaxation times τ_p, which in these results unfortunately do not approximate the analytic behaviour described in Kopf and coworkers[5] as well as other sources.

As a note on the references, the directly mentioned literature and reference texts are included, as well as other indirectly helpful material that was used in the making of this report.

[1] P. J. Rossky, J. D. Doll, and H. L. Friedman, "Brownian dynamics as smart monte carlo simulation," J. Chem. Phys. **69**, 4628–4633 (1978), https://doi.org/10.1063/1.436415.

[2] D. L. Ermak, "A computer simulation of charged particles in solution. i. technique and equilibrium properties," J. Chem. Phys. **62**, 4189–4196 (1975), https://doi.org/10.1063/1.430300.

[3] D. L. Ermak, "A computer simulation of charged particles in solution. ii. polyion diffusion coefficient," J. Chem. Phys. **62**, 4197–4203 (1975), https://doi.org/10.1063/1.430301.

[4] M. Rubinstein and R. H. Colby, "8: Unentagled polymer dynamics," in *Polymer physics* (Oxford University Press, 2003).

[5] A. Kopf, B. Dünweg, and W. Paul, "Dynamics of polymer isotope mixtures: Molecular dynamics simulation and rouse model analysis," J. Chem. Phys. **107**, 6945–6955 (1997), https://doi.org/10.1063/1.474934.

[6] J. S. Shaffer, "Effects of chain topology on polymer dynamics: Configurational relaxation in polymer melts," J. Chem. Phys. **103**, 761–772 (1995), https://doi.org/10.1063/1.470108.

[7] J. T. Padding and W. J. Briels, "Time and length scales of polymer melts studied by coarse-grained molecular dynamics simulations," J. Chem. Phys. **117**, 925–943 (2002), https://doi.org/10.1063/1.1481859.

[8] Y. Li, M. Kröger, and W. K. Liu, "Nanoparticle effect on the dynamics of polymer chains and their entanglement network," Phys. Rev. Lett. **109**, 118001 (2012).

[9] J. T. Kalathi, S. K. Kumar, M. Rubinstein, and G. S. Grest, "Rouse mode analysis of chain relaxation in homopolymer melts," Macromolecules **47**, 6925–6931 (2014), pMID: 25328247, https://doi.org/10.1021/ma500900b.

[10] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed. (Oxford University Press, 2017).