An Introduction to Polymer Physics

Daniel S. Bridges
August 11, 2009

Abstract

An overview of my learning experience at the University of Central Florida: Elementary polymer physics is discussed, with examples of work done.

Contents

1 Introduction 2
2 Projects 2
  2.1 Elementary Reading 2
  2.2 Distinguishable Particles Simulation 2
    2.2.1 The Verlet Algorithm 3
    2.2.2 The Velocity-Verlet Algorithm 3
  2.3 Simple Polymer Simulation 4
    2.3.1 Langevin Dynamics 4
3 Theory 4
  3.1 Diffusion Constant 5
  3.2 Distribution of the Chain 6
  3.3 Einstein’s relation 7
  3.4 End-to-end Distance 7
  3.5 Radius of Gyration 7
  3.6 Relaxation Time for End-to-end Distance (i.e. Polymer Conformation) 8
4 Results 9
  4.1 Distinguishable Particles 9
    4.1.1 velver.f 9
    4.1.2 micro.f 9
  4.2 Simple Polymer 9
5 Code 11
6 Figures 11
1 Introduction

"Graduate boot-camp", as Research Experiences for Undergraduates (REUs) are sometimes called [6, 9], is a worthwhile experience. It facilitates a certain maturity in students as they are placed in unknown circumstances, dealing with new – and sometimes disconcerting – situations. The experience may also introduce the student to new fields of research and academia, providing a context wherein the student may reconsider or refine his career plans.

For example, my career interests currently lie in medicine; I am pursuing a Bachelor of Science degree in Physics while emphasizing courses that relate to Medical Physics\(^1\) at Middle Tennessee State University\(^2\). Yet here I learned about the physics of polymers and attended lectures concerning abstract mathematics and nanoparticle manipulation, fields that are certainly related to Medical Physics, but of diverging interest: I would not have elected to learn about diffusion of salts across metallic surfaces in light of my interest to irradiate cancerous cells, but it was nevertheless enlightening: Such diverse lectures are useful for an undergraduate who needs “food for thought” in career contemplation.

This much being said, this paper summarizes my work done with the theories of polymer physics.

2 Projects

2.1 Elementary Reading

My professor was out of the country for the first week and a half, and I spent this time reading about Molecular Dynamics (MD) and Monte Carlo (MC) simulations [2, 4]. In general, MD involves integrating particles’ equations of motion, working exactly for simple cases, else approximating when necessary [6]. MC involves the extensive use of random numbers for statistical calculations [2]. MC can be used in MD to model distinguishable molecules, such as liquid argon atoms in a Lennard-Jones potential [8]. This simulation was the first performed and is detailed in the next section.

2.2 Distinguishable Particles Simulation

Code (“velver.f”) was provided by Dr. Bhattacharya for 108 particles which yielded a text file of their coordinates printed after discrete steps in time. These coordinates were used to determine the particles’ diffusion constant. This project served as an introduction to FORTRAN77, whose reference manual and user guide may be found online in PDF form. I was initially told that the code used the Verlet algorithm, but later that it was the “velocity” Verlet algorithm. Both algorithms are detailed in the following subsections. From reading

---

\(^1\)For more information, see the American Association of Physicists in Medicine at [http://www.aapm.org/](http://www.aapm.org/).

\(^2\)MTSU’s homepage is [http://mtsu.edu/](http://mtsu.edu/).
through the velver.f code, the positions etc. are advanced via kinematics.\textsuperscript{3} We were unable to obtain the diffusion constant from this code – apparently the imposed periodic boundary conditions mangled the root-mean-square calculations, so Dr. Bhattacharyya instead provided the micro.f code, which corrected the errors present in velver.f. The micro.f code was then used to duplicate a simulation conducted by Rahman [10].

2.2.1 The Verlet Algorithm

The Verlet algorithm, adopted by Verlet in 1967 and attributed to Störmer in 1971, is a numerical integration for determining particles’ position vectors, i.e. their coordinates [2]. It involves two Taylor series expansions and their sum, namely, Eqns (1) and (2) are added to obtain Eqn (3).

\[ \vec{r}(t + \delta t) = \vec{r}(t) + \delta t\vec{v}(t) + \frac{1}{2}\delta t^2\vec{a}(t) + ... \]  
\[ \vec{r}(t - \delta t) = \vec{r}(t) - \delta t\vec{v}(t) + \frac{1}{2}\delta t^2\vec{a}(t) + ... \]  
\[ \vec{r}(t + \delta t) = 2\vec{r}(t) - \vec{r}(t - \delta t) + \delta t^2\vec{a}(t) + ... \]  

Velocities, useful for estimating the kinetic energy (and total energy) may be obtained from the formula

\[ \vec{v}(t) = \frac{\vec{r}(t + \delta t) - \vec{r}(t - \delta t)}{2\delta t} \]  

2.2.2 The Velocity-Verlet Algorithm

A modified form of the Verlet algorithm discussed in Subsection 2.2.1, this algorithm takes the form

\[ \vec{r}(t + \delta t) = \vec{r}(t) + \delta t\vec{v}(t) + \frac{1}{2}\delta t^2\vec{a}(t) \]  
\[ \vec{v}(t + \delta t) = \vec{v}(t) + \frac{1}{2}\delta t [\vec{a}(t) + \vec{a}(t + \delta t)] \]  

where velocities are computed midstep via

\[ \vec{v}(t + \frac{1}{2}\delta t) = \vec{v}(t) + \frac{1}{2}\delta t\vec{a}(t) \]  

Forces and accelerations are then computed at time \( t + \delta t \), and the velocity “move” is completed:

\[ \vec{v}(t + \delta t) = \vec{v}(t + \frac{1}{2}\delta t) + \frac{1}{2}\delta t\vec{a}(t + \delta t) \]

\textsuperscript{3}See one of Dr. Vic J. Montemayor’s noncalculus-based physics textbooks for further clarification: \texttt{http://frank.mtsu.edu/~phys2010/Lectures/Part_1_L1_-_L5/Lecture_4/1-D_Kinematics/1-d_kinematics.html}
2.3 Simple Polymer Simulation

After the diffusion constant was calculated for the distinguishable particles, we considered the various characteristics of a polymer as described in Section 3. The simulation occurred in a cube which lacked solvent molecules, the code instead adding a random term to the monomer’s coordinates to simulate the random forces on the polymer by the solvent. The monomer’s coordinates were somehow normalized with respect to the cube: little is known about the executables provided by Dr. Bhattacharya. The simulation was run for eight-monomer and sixteen-monomer polymers.

2.3.1 Langevin Dynamics

For our purposes, if we assume that the Brownian motion of a given monomer has negligible effect on the polymer’s speed, then the particle displacement $x(t)$ satisfies

$$\frac{dx}{dt} = -\frac{1}{\zeta} \frac{\partial U}{\partial x}$$

The monomer moves to decrease its potential $U(x)$ until it reaches its position of minimum potential. But with Brownian motion, the monomer’s speed fluctuates about its average speed. We thus add a probability term $g(t)$ which varies randomly with time to obtain the Langevin equation:

$$\frac{dx}{dt} = -\frac{1}{\zeta} \frac{\partial U}{\partial x} + g(t)$$

3 Theory

When considering theory, writing the code that implements it is another matter entirely. I used FORTRAN77 exclusively; while it appears very good at summations, I never tried an integral ...

---

From discussion with Dr. Bhattacharya, we had hoped in reduced units that

$$D_G = \frac{k_B T}{N \zeta} = \frac{k_B \left(\frac{T}{\bar{T}}\right)}{N \zeta} = \frac{\epsilon \bar{T}}{N \gamma} = \frac{\epsilon \bar{T}}{N \left(\bar{\gamma} \sigma^2\right)} = \frac{\sigma^2 \bar{T}}{N \bar{\gamma}}$$

because Doi identifies $\zeta$ as the “viscous friction constant” on page 66 and as the “friction coefficient of a bead” on page 72. We had hoped this constant was the same as Allen and Tildesley’s “surface tension constant” $\gamma$ on page 327, as suggested by Bhattacharya, who uses $\gamma$ as “the friction coefficient”. However, it is clear by unit analysis that they are not the same.

Unit analysis of the last expression tells us we’re at a dead end: The diffusion constant has units of length-squared per time. Additionally, for the n8.x executable (for the eight-monomer polymer simulation) we have $N = 8$, $\bar{T} = 2$, but we are not told $\sigma$, $\zeta$, or $\gamma$. This aside is meant to stress the importance of converting between reduced units, a common practice to increase computer processing efficiency: for a discussion about them, see Appendix B of [2]. Furthermore, we hope to illustrate the necessity of tracking which symbols different authors assign to which variables.
3.1 Diffusion Constant

The diffusion constant is given by

\[ D = \frac{1}{2t} \int_0^t dt_1 \int_0^t dt_2 \langle v(t_1)v(t_2) \rangle \]

According to Dr. Bhattacharya [8], the code implementing this expression for a polymer should be “completely analogous” to that of the distinguishable particles. We note that, according to the Rouse model, the center of mass undergoes diffusion with a diffusion constant

\[ D_G = \frac{k_B T}{N \zeta} \] (10)

Also, we have the Einstein relation

\[ 2tD = \frac{1}{3} \left\langle \| \vec{r}_i(t) - \vec{r}_i(0) \|^2 \right\rangle \] (11)

(valid after a long time \( t \)). We note that, for \( N \) monomer coordinates \( \vec{R}_n = (x_n, y_n, z_n) \), the root-mean-square is given by

\[ R_{rms} \equiv \sqrt{\langle \vec{R}^2 \rangle} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} x_n^2 + y_n^2 + z_n^2} \] (12)

where, for the displacement of the particle during an interval of time \( t \), \( \xi(t) \),

\[ \langle \xi(t)^2 \rangle = R_{rms}^2 = \int_0^t dt_1 \int_0^t dt_2 \langle v(t_1)v(t_2) \rangle = 2tD \] (13)

where \( \langle v(t_1)v(t_2) \rangle = C_v(t_1-t_2) \) is the velocity correlation function. For our purposes, we consider the non-normalized autocorrelation function of \( v(t) \),

\[ C_{vv}(\tau) = \langle v(\tau)v(0) \rangle = \frac{1}{\tau_{max}} \sum_{\tau_0=1}^{\tau_{max}} v(\tau_0)v(\tau_0 + \tau) \] (14)

Note from Eqn (14) that the correlation function is a function of time separation \( \tau = |t_1 - t_2| \). Combining Eqns (13) and (14) we get, assuming the normalization is not a problem:

\[ \frac{t^2}{2\tau_{max}} \sum_{\tau_0=1}^{\tau_{max}} v(\tau_0)v(\tau_0 + \tau) = Dt \] (15)

---

5 A variation of (4.6) in reference [1].
6 (4.49) in reference [1].
7 Eqn (2.110) of reference [2] (page 60).
8 Section 4.1.1 (pages 66-67) of reference [1].
9 Section 6.3.1 (pages 185-188) of reference [2].
Unit analysis agrees so far ($\tau_0$, $\tau$ and $\tau_{\text{max}}$ are unitless): the diffusion coefficient has units of length-squared per time. However, we must work with coordinates, not velocities. Dr. Bhattacharya’s acf.f approaches the diffusion constant a different way: From Eqns (11) and (13) we see that

$$R_{\text{rms}}^2 = \frac{1}{3} \left\langle \| \mathbf{r}_i(t) - \mathbf{r}_i(0) \|^2 \right\rangle = 2tD \quad (16)$$

We are forced to make the assumption, based upon Dr. Bhattacharya’s code, that there exists an “autocorrelation function” $\kappa(\tau)$ such that

$$\left\langle \| \mathbf{r}_i(t) - \mathbf{r}_i(0) \|^2 \right\rangle \equiv \kappa(\tau) = \frac{1}{n_{\text{part}} \cdot \tau_{\text{max}}} \sum_{\tau_0=1}^{\tau_{\text{max}}} \sum_{n=1}^{n_{\text{part}}} r_n(\tau_0) r_n(\tau_0 + \tau) \quad (17)$$

so that $\frac{1}{6} \kappa(\tau) = Dt$. Plotting $\frac{1}{6} \kappa(\tau)$ versus $t$ we expect a straight line with slope $D$.

See Algorithm ?? on page ?? for code meant to determine the diffusion constant.

### 3.2 Distribution of the Chain

There are a few different approaches to describing the distribution of distinguishable particles or a polymer. This discussion addresses one method of doing so specifically for the polymer in the Rouse model, although many of these ideas – such as the root-mean-square and Einstein relation – apply for the case of distinguishable particles.

For the distribution of segments in the polymer chain, i.e. the segment pair correlation function $g(\mathbf{r})$ as discussed by Doi\textsuperscript{10}:

Writing $\bar{R}_n (n = 1, 2, ..., N)$ for the position vectors of the segments, we can express the average density of segments at a position $\mathbf{r}$ from segment $n$, $g_n(\mathbf{r})$:

$$g_n(\mathbf{r}) = \sum_{m=1}^{N} \left\langle \delta (\mathbf{r} - (\bar{R}_m - \bar{R}_n)) \right\rangle \quad (18)$$

The pair correlation function is then given by

$$g(\mathbf{r}) = \frac{1}{N} \sum_{n=1}^{N} g_n(\mathbf{r}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{m=1}^{N} \left\langle \delta (\mathbf{r} - (\bar{R}_m - \bar{R}_n)) \right\rangle \quad (19)$$

Since we’re dealing with a time average, Eqn (19) is actually

$$g(\mathbf{r}) = \frac{1}{N \cdot n_{\text{time}}} \sum_{n=1}^{N} \sum_{m=1}^{N} \sum_{t=1}^{n_{\text{time}}} \delta (\mathbf{r} - (\bar{R}_m(t) - \bar{R}_n(t))) \quad (20)$$

\textsuperscript{10}See Section 1.2.1.
Coding an algorithm to compute Eqn (20) appears intuitive and time consuming, but I am not sure how to vary the radius, a vector. It appears Eqn (20) gives the time- and ensemble-average number of monomers at a distance \( r \) from any given monomer, while the pair distribution function discussed by Allen-Tildesley gives “the probability of finding a pair of atoms a distance \( r \) apart”.\(^{11}\) Based on this information, see Algorithm ?? on page ??.

### 3.3 Einstein’s relation

I did not significantly study the theory behind Einstein’s relations, but one is the expression for the diffusion constant \( D \), given by Doi as

\[
D = \frac{k_B T}{\zeta} \tag{21}
\]

where \( \zeta \) is the (viscous) friction coefficient. With the Rouse model, a polymer with \( N \) monomers has a center-of-mass diffusion constant \( D_G \) of

\[
D = \frac{k_B T}{N \zeta} \tag{22}
\]

### 3.4 End-to-end Distance

The end-to-end vector is given by

\[
\vec{R}_{1N} = \sum_{n=1}^{N} \vec{r}_n
\]

where \( \vec{r}_n \) is the vector of the \( n \)th bond.\(^{12}\) That is, given a list of \( N \) coordinates,

\[
\vec{R}_{1N} = \sum_{n=1}^{N-1} \left[ \begin{array}{c} x_{n+1} \\ y_{n+1} \\ z_{n+1} \end{array} \right] - \left[ \begin{array}{c} x_n \\ y_n \\ z_n \end{array} \right] = \left[ \begin{array}{c} x_N \\ y_N \\ z_N \end{array} \right] - \left[ \begin{array}{c} x_1 \\ y_1 \\ z_1 \end{array} \right] \tag{23}
\]

See Algorithm ?? on page ??.

### 3.5 Radius of Gyration

The average radius of gyration is given by\(^{13}\)

\[
R_g^2 = \frac{1}{n_{\text{part}}} \sum_{n=1}^{n_{\text{part}}} \left\langle (\vec{R}_n - \vec{R}_G)^2 \right\rangle \tag{24}
\]

\(^{11}\)See Sections 2.6 (page 54) and 6.2 (page 183) of reference [2].

\(^{12}\)(1.1) of reference [1].

\(^{13}\)(1.30) of reference [1].
Assuming Eqn (24) means a time average, then

\[ R_g^2 = \frac{1}{n_{\text{part}}} \frac{1}{n_{\text{step}}} \sum_{n=1}^{n_{\text{part}}} \sum_{t=1}^{n_{\text{step}}} (\vec{R}_n(t) - \vec{R}_G(t))^2 \]  

(25)

where \( \vec{R}_G \) is the position of the center of mass.\(^{14}\)

\[ \vec{R}_G \equiv \frac{1}{n_{\text{part}}} \sum_{n=1}^{n_{\text{part}}} \vec{R}_n. \]  

(26)

or, in light of the time-dependence shown in Eqn (25),

\[ \vec{R}_G(t) \equiv \frac{1}{n_{\text{part}}} \sum_{n=1}^{n_{\text{part}}} \vec{R}_n(t) \]  

(27)

See Algorithm ?? on page ??\(^{15}\).

3.6 Relaxation Time for End-to-end Distance (i.e. Polymer Conformation)

Doi states,\(^{16}\)

\[ \tau_r = \frac{\zeta N^2 b^2}{3\pi^2 k_B T} \simeq \frac{N b^2}{D_G} \]  

(28)

the rotational relaxation time is also equal to the time required for the center of mass of the polymer to diffuse a distance comparable to the size of the polymer.

We have\(^{17}\)

\[ \langle \vec{P}(t) \cdot \vec{P}(0) \rangle = Nb^2 \sum_p \frac{8}{p^2 \pi^2} \exp(-tp^2/\tau_1) \]  

(29)

where \( p \) varies over positive odd integers, \( b \) appears to be the magnitude of each monomer \( n \) to \( n+1 \) vector,\(^{18,19}\) and \( \tau_1 \) is our desired relaxation time \( \tau_r \), given by Eqn (28).

\[ \vec{P}(t) = \vec{R}(N,t) - \vec{R}(0,t) = -4 \sum_p \vec{X}_p(t) \]

\(^{14}(1.29)\) of reference [1].
\(^{15}\)See also (4.48) of reference [1] for its connection to Brownian motion.
\(^{16}\)The rotational relaxation time appears to be the only relaxation time mentioned by Doi. See also his page 74.
\(^{17}(4.51)\) of reference [1].
\(^{19}\)\[2\] suggest that \( \sigma \) is the intermolecular spacing, so I know not the distinction between \( b \) and \( \sigma \).
where $\vec{X}_p(t) = \frac{1}{N} \int_0^N dn \cos \left( \frac{p \pi n}{N} \right) \vec{R}(n, t)$ for $p = 0, 1, 2, \ldots$. Hopefully, then, I guess

$$\vec{R}(0) = \vec{R}(N, 0) - \vec{R}(0, 0) = -4 \sum_p \vec{X}_p(0)$$

which gives us

$$\left\langle \left( \vec{R}(N, t) - \vec{R}(0, t) \right) \cdot \left( \vec{R}(N, 0) - \vec{R}(0, 0) \right) \right\rangle = Nb^2 \sum_p \frac{8}{p^2 \pi^2} \exp(-tp^2/\tau_1)$$

with positive odd integers for $p$. We must ascertain the limits for $p$ and value of $b$. Why is $p$ over odd integers, and how many odd integers? (Presumably from 1 to $\infty$.) Should we use the average bond length for the value of $b$? I was unable to find the answer to these questions.

4 Results

4.1 Distinguishable Particles

4.1.1 velver.f

Practice was gained in writing FORTRAN77 code, and I learned of the importance of collaboration when using another’s code.

4.1.2 micro.f

Not much can be said; I found Rahman’s diffusion constant to within a factor of five; neither I nor Dr. Bhattacharya were able to explain the lack of agreement.

4.2 Simple Polymer

This section is included for completeness: Not much can be said about the numbers’ meaning because very little is known about the already-compiled code I was given to execute. I did not write it and was not allowed to see it, and its output was unclear.

\footnote{(4.40) of reference [1]}
<table>
<thead>
<tr>
<th></th>
<th>npart = 8</th>
<th>npart = 16</th>
<th>Agrees with .dat?</th>
<th>Notes</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>average radius of gyration, $⟨R_g⟩$</td>
<td>0.82286066649</td>
<td>0.802873143538</td>
<td>No</td>
<td>$npos = ncm = 1000$, $ntime = 2 \times 10^7$. Dat file indicates increasing npart should increase $⟨R_g⟩$.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>average end-to-end distance, $⟨R⟩$</td>
<td>$1.1401633237 \times 10^{-2}$</td>
<td>$1.6114590346 \times 10^{-2}$</td>
<td>No</td>
<td>Off by a factor of 100 and length should almost double when monomer number is doubled.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>Distribution of chain</td>
<td>unknown</td>
<td>unknown</td>
<td></td>
<td>I do not know how to vary the distance vector $\vec{r}$.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>Diffusion constant</td>
<td>unknown</td>
<td>unknown</td>
<td></td>
<td>Results are nonlinear.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>travel time for radius of gyration</td>
<td>unknown</td>
<td>unknown</td>
<td></td>
<td>Center of mass motion appears to be random. Don’t know how to calculate this time.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>relax. time for end-to-end distance</td>
<td>unknown</td>
<td>unknown</td>
<td></td>
<td>We expect to derive it from the Diffusion constant.</td>
<td>?? on page ??</td>
</tr>
<tr>
<td>comparison of these times</td>
<td>should be the same</td>
<td>should be the same</td>
<td>No</td>
<td>$avr1lnlt$ and $avrglt$ for n8.x are different.</td>
<td>?? on page ??</td>
</tr>
</tbody>
</table>

It is possible the executables involve some unit conversions, which are not taken into account by my code. The executables provided by Dr. Bhattacharya write a file polyxyz.dat which contains numbers for me to compare with my results, although I have no idea how those numbers are computed by the code (or even if they are what I think they are: his labels are occasionally unclear).
5 Code

FORTRAN77 (F77) was the programming language used. While the code was running, it would produce simply-formatted text files ("data files" which we commonly end with the extension "data") to be read by the X-Y plotting software Grace. Everything was available freely with the Ubuntu Linux operating system, which is itself available freely online.

6 Figures

Grace was used to read the output data files (see Section 5). Shown below are some examples of graphs readily produced with Grace.

![Figure 1: The end-to-end vector magnitude fluctuates with time.](image)
Figure 2: Movement of the center of mass over time
Figure 3: \texttt{dbacf.f} output for \textit{npart} = 8, \textit{ntime} = 1e6, \textit{npos} = 1e4, \textit{ncorr} = 5e3.
Figure 4: Another attempt at calculating the diffusion constant (should be straight line through the origin with positive slope)

References


