Simulation of Dynamics of polymers
dissolved in solvent

Submitted by
Ruchika Mahajan
(DI1504)

Under the Guidance of
Dr. Prasanth P. Jose

Indian Institute of Technology Mandi
Kamand, Mandi 175005
Himachal Pradesh, INDIA.

Contents

1 Abstract

In this report we are trying to do simulate polymer in a solvent using molecular dynamics code.

2 Introduction

A computer simulation is a simulation, run on a single computer, or a network of computers, to reproduce behavior of a system. The simulation uses an abstract model (a computer model, or a computational model) to simulate the system.In other words it is the manipulation of a model in such a way that it represents the expected behavior of an individual actor or an entire system over time .It allow physicists to explore numerous systems and geometries at relatively low costs and at various levels of detail. Thats why computational physics is widely seen as a third approach to scientific discovery and theoretical techniques. Computational physics can be split into two category:

- Computational physicists generate models that can be solved numerically. When performing such calculations, physicists can be referred to as numerical theoreticians

- Computer simulations based on simple models develop the states of many constituents in time (which rapidly becomes analytically impossible). Relatively computers can calculate countless interactions between particles and produce emergent collective behaviour that would not be visible in models that average-out many degrees of freedom. When computational physicists perform such simulations they can be referred to as in-silico experimentalists.

In-silico experimentalists have both full control over the simulation parameters and the ability to measure every imaginable property or correlation of a system that is incorporated into the model at any instant. Having full control over objects, interactions and forces can allow (perhaps non-physical) simulations to determine which factors play what role. This strategy can lead to better fundamental understanding of phenomena

In this experiment we dissolved polymers as a solute into the solvent that give three types of interactions solvent-solvent, solute-solute and solvent solute. Because of these interactions(although there are many interactions we are not talking about that) the system become complex therefore analytically it is very difficult to calculate these interactions thats why we do the simulation to analyze the properties of the system. Here we are using molecular dynamics (MD) simulation to do this experiment. Here we are using molecular dynamics **(MD)** simulation to perform this experiment.In MD simulations we are Integrating the Newtons equation of motions numerically with appropriate choice of potential using velocity verlet algorithm. Velocity verlet is a numerical method used to integrate Newton's equations of motion and frequently used to calculate trajectories of particles in molecular dynamics **(MD)** simulations. The Verlet method of advancing positions and velocities can be obtained as shown below:

$$\begin{aligned} X_i(t+\Delta t) &= X_i(t) + \Delta t V_i + \frac{1}{2}(\Delta t)^2 \frac{F_i}{m_i} \\ V_i(t+\Delta(t)) &= V_i(t) + \frac{1}{2}(\Delta(t)) + \frac{F_i(t) + F(t+\Delta(t))}{m_i} \end{aligned}$$

3 molecular dynamics simulation of liquid in a cube

In a dilute gas, the molecules are widely separated and the molecules move several (tens) of angstroms before "colliding" with another molecule. In a liquid, the packing is very similar to that of a solid except that the molecules are moving all the time.
Consider N number of particles inside a cubical box. These particles or atoms behave approximately like hard spheres which attract one another with weak van der Waals forces. The forces between two atoms can be approximated quite well by a Lennard-Jones potential energy function

$$V(r) = 4\varepsilon[\frac{\sigma}{r}^{12} - \frac{\sigma}{r}^{6}],$$

where r is the distance between the centers of the two atoms, $\varepsilon = 1.6510\ 21$ J is the strength of the potential energy, and $\sigma = 3.4\ \ 10\ 10$ m is the value of r at which the energy is zero. The $(1/r)^{12}$ term represents a repulsive hard core interaction between the atoms. The $(1/r)^{6}$ term represents an attractive dipole-dipole (van der Waals) interaction between the non-polar atoms. The potential has its minimum V $(2^{1/6}\sigma) = \varepsilon$ at r $= 2^{1/6}$ σ.The shape of the potential and the strength of the Lennard-Jones force

$$\begin{aligned} F(r) &= -\frac{dV(r)}{dr} \\ &= 24\frac{\varepsilon}{\sigma}[2\frac{\sigma}{r}^{13} - \frac{\sigma}{r}^{7}], \end{aligned}$$

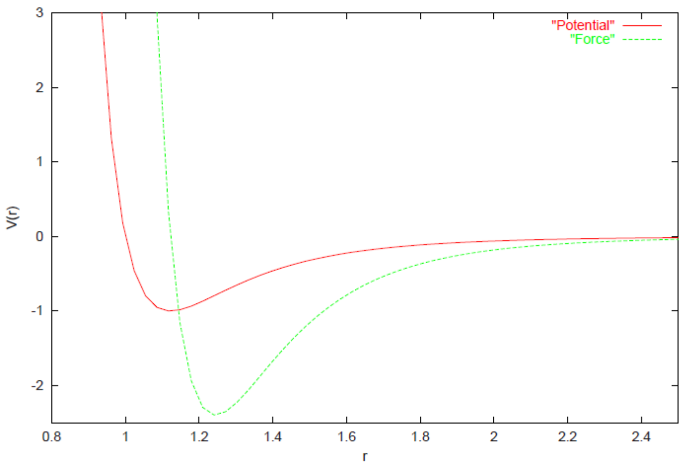

Figure 1: linnard jones potential

We will choose units of mass, length and energy so that m = 1, $\sigma = 1$, and $\varepsilon = 1$.

3.1 A simple MD programme

First include some standard headers
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

Define data structures to describe the kinematics of the system

int np; //Number of particles to simulate
double **r;
r[ ][0-2] - positions
r[ ][3-5] - velociites

r[ ][6-8] - new forces
r[ ][9-11] - old forces

In this program we will put the system in a cubical volume of side L and place the particles randomly in a simple cubic box using ran2 idum function: We next need to set the initial positions and velocities of the particles, the initial configuration is

```
x[m] = 2.0*ran2(&zzz) - 1; // position of particles in x,y and z direction
y[m] = 2.0*ran2(&zzz) - 1;
z[m] = 2.0*ran2(&zzz) - 1;
```

The particles or atoms are also given random velocities

```
//Assign velocites
vs[0]=vs[1]=vs[2]=0.0;

for(i=0 ; i<n ; i++)

r[i][3] = randno();
vs[0]+ = r[i][3];
r[i][4] = randno();
vs[1]+ = r[i][4];
r[i][5] = randno();
vs[2]+ = r[i][5];
```

3.2 Newton's equation of motion

The vector forces between atoms with positions r_i and r_j are given by

$\mathbf{F}_{ij} = -\mathbf{F}_{ji} = 24*(r_i - r_j)[2\frac{\sigma}{r}^{-14} - \frac{\sigma}{r}^{-8}]$

where $r = \mid r_i - r_j \mid$

The net force on atom i due to all of the other N 1 atoms is given by

$\mathbf{F}_i = \sum_{j=1}^{N} F_{ij}\ (j \neq i)$

The equation of motion for atom i is

$a_i(t) = \frac{dV_i}{dt} = \frac{d^2r_i(t)}{dt^2} = F_i/m$

where v_i and a_i are the velocity and acceleration of atom i. These 3N second order ordinary differential equations (ODEs) have a unique solution as function of time t if initial conditions, that is, the values of positions r_i (t=0) and velocities v_i(t<0) of the particles are specified at some initial time t 0 . The equations can be integrated numerically by choosing a small time step dt and a discrete approximation to the equations to advance the solution by one step at a time.

3.3 velocity vertlet algorithm

There are many algorithms which can be used to solve ODEs. Verlet has developed several algorithms which are very widely used in MD simulations. One of them is the velocity Verlet algorithm

$$\begin{aligned} r_i(t+dt) &= r_i(t) + V_i dt + \frac{1}{2}\frac{F_i}{m_i}dt^2 \\ V_i(t+(dt)) &= V_i(t) + \frac{1}{2}\frac{F_i(t)+F(t+dt)}{m_i}dt \end{aligned}$$

The following function advances the positions and velocities of the particles by one time step

```
    for(i=0; i<np; i++)

//Update on velocities
r[i][3]+=0.5*(r[i][6]+r[i][9])*dt;
r[i][4]+=0.5*(r[i][7]+r[i][10])*dt;
r[i][5]+=0.5*(r[i][8]+r[i][11])*dt;

//Update on position
r[i][0]+=r[i][3]*dt+r[i][6]*dt2b2;
r[i][1]+=r[i][4]*dt+r[i][7]*dt2b2;
r[i][2]+=r[i][5]*dt+r[i][8]*dt2b2;

//replace old force array by new forces
r[i][9]=r[i][6];
r[i][10]=r[i][7];
```

```
r[i][11]=r[i][8];
```

3.4 The instantaneous temperature

This is a simulation in which the number of particles N and the volume L^3 of the system are fixed. Because the Lennard-Jones force is conservative, the total energy of the system is also constant.If the system is in thermal equilibrium, then Boltzmanns Equipartition Theorem relates the absolute temperature T to the kinetic energy.

$$3(N-1)\times\frac{1}{2}k_BT \quad = \quad \langle\frac{m}{2}\sum_{i=1}^{N}{V_i}^2\rangle$$

Here the angle brackets represent a thermal ensemble average. The factor 3(N 1) is the number of internal translational degrees of freedom which contribute to thermal motion.

```
//Temperature of the system
double temp;
temp=ke/(double)np/3.0;
//Calculate scaling factor

double sc;

    sc=sqrt(te/temp);

for(i=0; i<np; i++)
r[i][3]*=sc;
r[i][4]*=sc;
r[i][5]*=sc;
```

The volume is not really constant because the particles can move out of it. Periodic boundary conditions will be used to ensure that the number of particles in the simulation volume remains constant
the minimum image convection is used to compute the position

```
for(i=1; i<n; i++)
```

```
dx=x[i+1]-x[i];

while(dx > blb2)dx- = bl;
while(dx ≤ blb2)dx+ = bl;

dy=y[i+1]-y[i];

    while(dy > blb2)dy- = bl;
while(dy≤-blb2)dy+ = bl;

dz = z[i+1]-z[i];

while(dz > blb2)dz- = bl;
while(dz≤-blb2)dz+ = bl;

vol= dx*dx + dy*dy + dz*dz;
```

References

[1] Michael P. Allen, Introduction to Molecular Dynamics Simulation, NIC Series, Vol. 23, ISBN 3-00-012641-4, pp. 1-28, 2004.