Abstract
The Brownian dynamics (BD) simulation has been applied to study a wide range of biological problems that involve interactions with surrounding fluid molecules or the so-called hydrodynamic interactions (HIs). We propose an efficient algorithm to simulate the motion of DNA molecules in linear flows. The algorithm utilizes the integrating factor to cope with the effect of the linear flow of the surrounding fluid and applies the Metropolis method (MIM) to achieve more efficient BD simulation. Thus our method permits much larger time step sizes (often by a factor of 100 - 1000) than previous methods while still maintaining the stability of the BD simulation, which is advantageous for long-time BD simulation. Our numerical results on DNA agree very well with both experimental data and previous simulation results. This will be useful for the study of membranes, long-chain molecules, and a large collection of molecules in the fluids.

Introduction
In recent years, scientists have developed the method to observe the behavior of DNA in flow such as shear flow or extensional flow directly by using fluorescence dye (see Figure 1). Ermak and McCammon (1978) first studied the many particle model with thermal fluctuation and hydrodynamic interaction. Here, we will study the Langevin equation describing the trajectories of the particle positions.

Figure 1: (A) Images of molecular configurations spaced every 0.13 sec. They are the observations of molecules in extensional flow. From top to bottom, there are four configurations: dumbbell, kinked, half-dumbbell, and folded. Source: Perkins 1997. (B) Images of molecule undergoing configuration changes (coiled, stretched, and folded) in shear flow. Source: Smith 1999.

Governing equations

In the past, a semi-implicit predictor-corrector scheme [1, 2] is often used for the temporal integration in BD simulations. A major problem associated with that scheme is that a very small time step size has to be used in order to avoid the numerical instability, which leads to an excessively large number of time steps and a very long total simulation time. Here we apply the Metropolis algorithm to study BD simulations in linear flows. The governing stochastic differential equation is as follows:

\[
\frac{d\mathbf{x}_i}{dt} = \mathbf{a}(\mathbf{x}_t) + \mathbf{F}_f (\mathbf{x}_t) \mathbf{x}_t + \sqrt{2\kappa} \frac{1}{\sqrt{\Delta t}} \sum_j \mathbf{n}_{ij} dW_j, \quad i = 1, \ldots , N,
\]

where \( \mathbf{r}_i \) is the position of bead \( i \), \( \mathbf{F}_f \) is the net interparticle force including worm-like chain spring force and excluded volume force, \( \mathbf{a}(\mathbf{x}_t) \) is the coefficient tensor of the fluctuations, \( \kappa^2 = \frac{6}{8\pi \eta R} \) is the thermal energy, and \( \mathbf{w} \) is so-called Wiener process which is a Gaussian with mean and covariance, where \( \eta \) is the constant velocity gradient tensor and \( \mathbf{r}_i \) is the bead radius and \( \mathbf{F}_f \) is the \( i \times 1 \) identity matrix. Using the integrating factor method to the equation (1) and letting \( \tilde{K} (t) = \exp(-\kappa t) \), we introduce a new variable \( \tilde{\mathbf{x}}_t = \tilde{K}(t) \mathbf{x}_t \). Then the original SDE (1) can be rewritten in terms of \( \tilde{\mathbf{x}}_t \) as follows:

\[
\tilde{\mathbf{x}}_t = \tilde{K}(t) \mathbf{x}_t + \tilde{K}(t) \int_0^t \tilde{K}^{-1}(\tau) \mathbf{a}(\mathbf{x}_\tau) d\tau + \sqrt{2\kappa}(\tilde{\mathbf{x}}_t) dW_t.
\]

It is straightforward to show that the generator of this SDE (4) is self-adjoint with respect to its stationary density function.

Numerical algorithms

1. Compute the vector \( \tilde{\mathbf{x}}^{n+1} \) and update \( \mathbf{x}^{n+1} \) by the following formulas:

\[
\tilde{\mathbf{x}}^{n+1} = \tilde{K}(t) \mathbf{x}^n + \tilde{K}(t) \int_0^t \tilde{K}^{-1}(\tau) \mathbf{a}(\tilde{\mathbf{x}}^\tau) d\tau + \sqrt{2\kappa}(\tilde{\mathbf{x}}^n) dW^n,
\]

where the functions \( \tilde{K} \) and \( \mathbf{a} \) are defined by the formulas:

\[
\tilde{\mathbf{x}}^{n+1} = \tilde{K}(t) \mathbf{x}^n + \frac{1}{2} \tilde{K}(t) \mathbf{F}(\tilde{\mathbf{x}}^n) d\mathbf{t} + \frac{1}{2} \mathbf{K}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t} + \frac{1}{2} \mathbf{F}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t} + \frac{1}{2} \mathbf{K}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t} + \mathbf{F}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t} + \frac{1}{2} \mathbf{K}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t} + \mathbf{F}(\tilde{\mathbf{x}}^n) \tilde{\mathbf{r}}^n d\mathbf{t}.
\]

We then apply the Metropolis integrator to obtain the updated \( \mathbf{x}^{n+1} \). We introduce a new variable

\[
\tilde{\mathbf{x}}_t = \tilde{K}(t) \mathbf{x}_t + \tilde{K}(t) \int_0^t \tilde{K}^{-1}(\tau) \mathbf{a}(\mathbf{x}_\tau) d\tau + \sqrt{2\kappa}(\tilde{\mathbf{x}}_t) dW_t.
\]

2. Calculate the acceptance probability \( \alpha \) as follows:

\[
\alpha(\mathbf{x}^{n+1} | \mathbf{x}^n) = \min \left\{ 1, \exp \left[ -\frac{\mathbf{a}(\mathbf{x}^n)^T \mathbf{a}(\mathbf{x}^n)}{2\kappa} \right] \right\}.
\]

3. Generate a Bernoulli random number \( r \), that is, generate a uniformly distributed random number \( \beta \) on \([0, 1]\) and set \( \gamma \) to 0 if \( \beta \leq \alpha \) and \( \gamma \) otherwise.

4. Compute the updated position vector at time \( t = t_n + \Delta t \) by the formula:

\[
\mathbf{x}^{n+1} = \gamma (\mathbf{K}(\mathbf{x}^n) \mathbf{x}^n + 1) + (1 - \gamma) \mathbf{x}^n.
\]

In other words, the position vector will be updated only if the Bernoulli random number \( \gamma \) is equal to 1.

Highlights of the new scheme

1. The convergence order of the existing methods [11, 22] for linear flows is only 0.5 due to the random effect.
2. The convergence order of our scheme is 1 for deterministic models, and 1/2 with the presence of random terms due to the Metropolis step in our algorithm.
3. Our scheme permits much larger time step size (often by a factor of 100 - 1000) than the existing methods.

References