Simulating of 2D polymers using the Rosenbluth-Rosenbluth and the Pruned-Enriched Rosenbluth method to determine the scaling exponent

Liza de Wilde (4246853) & Aloys Erkelens (4339304)

April, 2016

Abstract

In this paper we discuss the simulation of two dimensional polymers by the Rosenbluth-Rosenbluth method and the Pruned-Enriched Rosenbluth method. The end to end distance is computed for 10 000 polymers with a length of 250 beads, from which the scaling exponent of $\nu = 0.68 \pm 0.01$ was found using the RR method and $\nu = 0.752 \pm 0.003$ using PERM. The value of the scaling exponent determined with the PERM algorithm is in agreement with the theoretical value of $\nu = 3/4$.

Key words: Monte Carlo; Polymer; Rosenbluth; Pruned-Enriched Rosenbluth

1. Introduction

Plastics are important for industry and DNA, RNA, and proteins are essential for life. These are all polymers. Polymers are macromolecules consisting of long chains of monomers. These polymers can show various types of behaviour.

For dilute polymers the interactions between the monomers determines the structure of the protein. Some polymers fold in a specific structure due to interactions between the monomers and the solvent whereas others are represented by flexible chains.

Computer simulations can be used to study the configurations and statistical properties of polymers. Computer simulation techniques such as molecular dynamic methods and Monte Carlo methods are widely used to study polymers [1, 2, 3]. In static Monte Carlo methods a large number of random configurations are created independent from each other. In this paper we will discuss the results of a Monte Carlo simulation of a linear polymer in a 2D self avoiding walk (SAW) by the Rosenbluth-Rosenbluth (RR) method and the Pruned-Enriched Rosenbluth method (PERM).

2. Theory & Method

The behaviour of polymers can be described on different scales, from the atomic scale to the global scale of the chain. We restrict ourselves to polymers consisting of one monomer species only in a dilute polymer solution. For this simulation a mesoscopic model of the polymer is used, which describes the polymers by a long flexible chain of monomers at a fixed distance. If the solvent is good, the monomers repel each other [1]. The interactions of the monomers are modelled by the Lennard-Jones potential which approximates core repulsions with a r^{-12} term and short range Van der Waals contribution with a r^{-6} term:

$$U_{LJ}(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \qquad (1)$$

Here ϵ describes the depth of the potential well and σ the monomer radius. We choose the distance between monomers to be $|\mathbf{x}_n - \mathbf{x}_{n-1}| = 1$ and $\epsilon = 0.25$ and $\sigma = 0.8$ according to Ref. [1].

The SAW is a simple model for linear chain polymers such as polyesters and polyethylene. In this model the polymer is forbidden to cross itself. One of the first methods to simulate polymer configurations is the RR method [4]. An example of a polymer grown according to the RR method is shown in figure 1.

2.1. Rosenbluth-Rosenbluth

The RR method was first described as a on-lattice algorithm but can also be modified to a off-latice algorithm. In RR a chain is built by adding a monomers at the end as a biased random walk where crossing itself is forbidden. The RR method is biased to improve algorithm efficiency. This algorithm starts with two beads placed at (0,0) and (1,0). New beads are added at each step, until the total length of the polymer reaches N beads. For

the new bead position a discrete set of trial positions are chosen, characterised by the angle θ between the last two beads and the new bead. This discrete angle θ has a random offset and is spaced by $2\pi/N_\theta$, where N_θ is the amount of discrete angles.

The local environment around the last bead is scanned and the trial positions which lead to self-intersection are excluded. This is ensured by calculating the weight of each trial position according to the Boltzmann distribution. The weight of the trial positions of a polymer of length n is given by:

$$w_n^j = \mathrm{e}^{-E(\theta_j)/k_B T} \tag{2}$$

and the sum of these weights gives the local partition function:

$$W_n = \sum_j w_n^j \tag{3}$$

Where k_B is the Boltzmann constant and T is the temperature. The energy $E(\theta_j)$ of the trial bead positions with an angle θ_j to the existing beads of the polymer is described by the Lennard-Jones potential, Eq. 1.

One of the trial positions is selected with probability: $P_j = w_n^j/W_n$ by the roulette-wheel algorithm [1]. The angle is chosen by a uniform random number between 0 and 1. The interval [0,1] is divided into N_θ segments with size w_n^j/W_n. The angle corresponding to the segment which includes the random number is then accepted.

When the polymer is grown up to its total length N, the total Boltzmann weight of the polymer is the product of the weights of the chosen angles:

$$\mathrm{e}^{-E_{pol}/k_B T} = \prod_{l=3}^{N} w_n^j \tag{4}$$

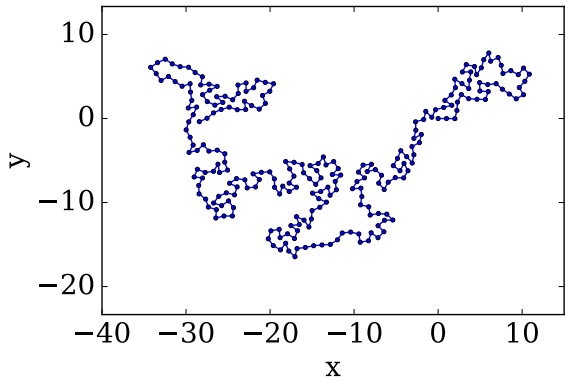


Figure 1: A polymer grown up to 250 beads according to the RR algorithm.

The weight of the polymer at a length of n beads is defined as:

$$W_n^{pol} = \prod_{l=3}^{n} W_l \tag{5}$$

Such that the total weight of the polymer is reached when $n \to N$. The probability of generating a certain polymer is given by:

$$P_N^{pol} = \prod_{l=3}^{N} \frac{w_l^j}{W_l} \tag{6}$$

In Algorithm 1 the pseudo-code of the RR method is shown. This algorithm is written in a recursive structure. Occasionally the polymer will intersect itself or roll itself up. When this happens the polymer is deleted and a new one is grown instead.

The RR method is a very useful method to generate a polymer population. However for polymers with lengths $N > 100$ the error becomes larger [1]. This is caused by the fact that RR methods does not suppress high-energy configurations sufficiently. In order to overcome this, PERM can be used. PERM combines both the RR method with recursive enrichment.

2.2. Pruned-Enriched Rosenbluth Method

PERM can be implemented both as a on-lattice and off-lattice model which combines the RR method with recursive enrichment where 'good' polymers are enriched and 'bad' polymers are pruned [5]. The weights of the polymers are calculated in a similar way as in the RR method. The limits for 'good' polymers and 'bad' polymers to be enriched and pruned are chosen by a constant multiplied by the an estimate of the partition function [1, 5, 6]:

Algorithm 1 Rosenbluth-Rosenbluth Algorithm

function GROWPOLYMER(x, W_n,n)
  Create N_θ angles and corresponding trail positions
  Calculate Weight of new positions as: $w_n^j = \mathrm{e}^{-E(\theta_j)/k_B T}$
  $W_n = \sum_j w_n^j$
  $W_n^{pol} = W_n^{pol} \cdot \frac{W_n}{0.75 N_\theta}$
  while $N < N_{max}$ **do**
    Draw new position from list or trial positions at random according to probability distribution
    GROWPOLYMER(x, W_n, n+1)
  end while
end function

$$W_n^+ = \alpha \cdot \widehat{Z}_n = \alpha \cdot \frac{\widehat{W}_n}{\widehat{W}_3} \tag{7}$$

$$W_n^- = \beta \cdot \widehat{Z}_n = \beta \cdot \frac{\widehat{W}_n}{\widehat{W}_3} \tag{8}$$

Here $\widehat{W}_n$ is the average weigth of a polymer with length n and $\widehat{Z}_n$ is the current estimate of the partition function for polymers of length n. In our simulations we used values of $\alpha = 2$ and $\beta = 1.2$ which have previously been published [1]. There is no estimate for the partition function for the first polymer to be constructed, therefore the first polymer is constructed by the RR method. After the first polymer has been created the estimate of the partition function is updated after each step. For the remainder of the polymers the polymer weight (Eq. 5) is compared to the upper limit W_n^+ and lower limit W_n^- after each growth step. If the polymer weight is larger than W_n^+ the polymer is enriched, which means a copy of the polymer is made and both polymers are grown until the desired length N is reached. The weight of both copies of the polymer have to be corrected by multiplying both with a factor 1/2. For polymer weights lower than W_n^- the polymer is pruned with a probability of 1/2. If the polymer survives, the polymer weight is corrected by multiplying it with a factor 2. The recursive pseudo-code for PERM is shown in algorithm 2.

Algorithm 2 Pruned Enriched Rosenbluth Method Algorithm

```
function growPolymer(x, W_n, n)
    Create N_θ angles and corresponding trial positions
    Calculate Weight of trial positions as: w_n^j = e^{-E(θ_j)/k_B T}
    W_n = Σ_j w_n^j
    W_n^pol = W_n^pol · W_n / (0.75 N_θ)
    while N < N_max do
        Draw new position from list or trial positions at random according to probability distribution
        Update W_n^+ and W_n^-
        W_n^+ = α · Ŵ_n / Ŵ_3
        W_n^- = β · Ŵ_n / Ŵ_3
        if W_n > W_n^+ then
            W_n = W_n/2
            growPolymer(x, W_n, n+1)
            growPolymer(x, W_n, n+1)
        else if W_n < W_n^- then
            if random number < 0.5 then
                W_n = 2W_n
                growPolymer(x, W_n, n+1)
            else
                Stop polymer growth
            end if
        end if
    end while
end function
```

3. Results & Discussion

In our simulation polymers are grown up to a length of 250 beads, with a population size of 10 000. We will not describe a specific polymer species, thus for the sake of generality we take $k_B = 1, T = 1$.

In the simulation the end to end distance is determined for polymers simulated by both the RR method and PERM (Fig.2). The end to end distance for a polymer of length n can be calculated as $\langle R_n^2 \rangle = \langle (\mathbf{x}_n - \mathbf{x}_1)^2 \rangle$. For polymers simulated by the RR method the end to end distances fluctuates for larger values of N (Fig 2(a)). The RR algorithm does not suppress high-energy configurations, but accepts them with a low weight. This makes the errors larger although this can not be seen in the figure, because the number of polymers is very large.

In figure 2(b) the end to end distance of polymers simulated by the PERM algorithm is shown. The population size is no longer constant because of pruning and enriching. Due to the population control the polymers are also correctly described at longer chain lengths, and hereby solves the issues in the RR method.

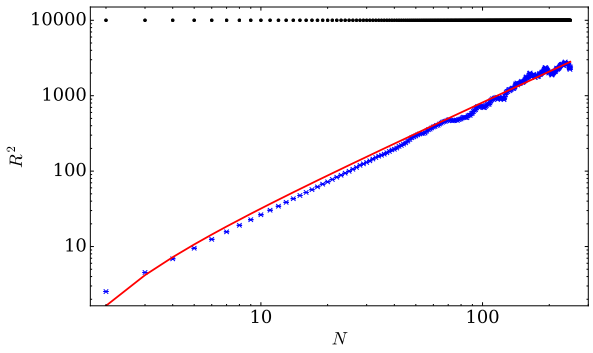
The end to end distance found in the simulations can be related to the number of beads by the scaling exponent ν:

$$\langle R^2 \rangle \propto N^{2\nu} \tag{9}$$

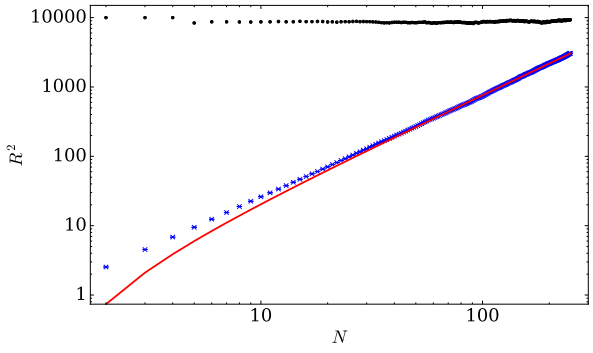
The scaling exponent only depends on the model used for the polymer and its spacial dimensions [7]. For all SAWs in a specific spacial dimension, the scaling exponent is the same for both on-lattice and off-lattice algorithms. In two dimensions the theoretical value for $\nu = 3/4$ [1]. This value is reached asymptotically for $N \to \infty$.

The end to end distance found in the simulation are fitted to the function $R^2 = a(N-1)^{2\nu}$ by a non-linear least square fit. From the simulation by the RR method and PERM the parameters of the fit are found to be:

$$\langle R_N^2 \rangle_{RR} = (1.6 \pm 0.2) \cdot (N-1)^{2 \cdot (0.68 \pm 0.01)}$$



(a)



(b)

Figure 2: Scaling behaviour of the end to end distance R as a function of the number of beads N. Blue crosses are the end to end distances from the simulation by (a) the RR algorithm and (b) the PERM algorithm. The red line is the fit of the function $a(N-1)^{2\nu}$ to the data. Black dots represent the population size.

$$\left\langle R_N^2 \right\rangle_{PERM} = (0.75 \pm 0.02) \cdot (N-1)^{2\cdot(0.752\pm0.003)}$$

The value for the scaling exponent determined by the RR method is below the theoretical value. The scaling exponent determined using PERM is in good agreement with the theoretical value. This confirms effectiveness of PERM compared to the RR method.

4. Conclusion

In this report polymers were simulated in two dimensions using the RR method and PERM. For this simulation a population of 10 000 polymers were grown up to a length of 250 beads. From these simulations a scaling exponent was found of $\nu = 0.68 \pm 0.01$ using the RR method, and $\nu = 0.752 \pm 0.003$ using PERM. The value determined with PERM is in agreement with the theoretical value of $\nu = 3/4$.

References

[1] J. Thijssen. *Computational Physics*, chapter 10: The Monte Carlo method, pages 319–325. Cambridge University Press, second edition, 2007.

[2] J. Baschnagel, J. P. Wittmer, and H. Meyer. Monte Carlo SImulation of Polymers: Coarse-Grained Models. *eprint arXiv:cond-mat/0407717*, July 2004.

[3] K Binder. *Monte Carlo and molecular dynamics simulations in polymer science*. Oxford University Press, New York, 1995.

[4] Marshall N. Rosenbluth and Arianna W. Rosenbluth. Monte carlo calculation of the average extension of molecular chains. *The Journal of Chemical Physics*, 23(2):356–359, 1955.

[5] Peter Grassberger. Pruned-enriched rosenbluth method: Simulations of θ polymers of chain length up to 1 000 000. *Phys. Rev. E*, 56:3682–3693, Sep 1997.

[6] Hsiao-Ping Hsu and Peter Grassberger. A review of monte carlo simulations of polymers with perm. *Journal of Statistical Physics*, 144(3):597–637, 2011.

[7] Nicolas Moreno, Suzana Nunes, and Victor M. Calo. Restrictions in model reduction for polymer chain models in dissipative particle dynamics. *Procedia Computer Science*, 29:728 – 739, 2014. 2014 International Conference on Computational Science.