HA 2 FMSN50 Ekman & Sundell

Agnes Ekman (ag8720ek-s) & Gustaf Sundell (gu0147su-s)

February 2021

1 Theory of Self Avoiding Walks

Throughout this report, the theory of self-avoiding walks (SAW) and the related notations presented in the instructions for Home assigment 2, page 1, will be used. The true values of c_n that are found in table 3 are gathered from Slade's "Self-Avoiding Walks"[1].

1.1 Inequality (Q1)

As a first task, we want to come up with some intuitive argument to make the following inequality plausible:

$$c_{n+m}(d) \leq c_n(d)c_m(d) \tag{1}$$

One may look at LHS as taking n steps, and then m more. For the m extra steps, each trajectory is limited by the n steps already taken, making it probable that each trajectory has $\leq c_m(d)$ possible continuations, since $c_m(d)$ is the number of possible m-step trajectories starting at $\mathbf{0}$, and with no prior trajectories to avoid. In other words, you will have more choices when starting a path with no prior paths to avoid. This means that the number of SAW:s with n steps will, at most, be multiplied with the number of SAW:s with m steps when extending the number of steps to $n+m$, yielding the inequality.

1.2 Proving limit (Q2)

Taking the logarithm of the relationship discussed above in section 1.1 gives a subadditive sequence:

$$ln(c_{n+m}(d)) \leq ln\left((c_n(d)c_m(d)\right) = ln(c_n(d)) + ln(c_m(d)) \tag{2}$$

Hence, set:

[1]Slade, Gordon. "Self-Avoiding Walks". *The Mathematical Intelligence* Vol. 16 No. 1. 1994. Springer Verlag, New York.

$$a_n = ln(c_n) \tag{3}$$
$$\implies \frac{a_n}{n} = \frac{1}{n} ln(c_n) = ln(c_n^{1/n}) \tag{4}$$
$$\implies c_n^{1/n} = e^{a_n/n} \tag{5}$$

Then, using $Fekete's\ lemma$:

$$l = \lim_{n\to\infty} \frac{a_n}{n} = \inf_{n\geq 1} \frac{a_n}{n} \tag{6}$$
$$\implies \mu_d = \lim_{n\to\infty} c_n(d)^{1/n} = \lim_{n\to\infty} e^{\frac{a_n}{n}} = e^{\lim_{n\to\infty} \frac{a_n}{n}} = e^l = e^{\inf_{n\geq 1} \frac{a_n}{n}} \tag{7}$$

Proving that the limit indeed exists. Please note that the case of $\inf_{n\geq 1} \frac{a_n}{n} = -\infty$ is will never occur in this setting of SAW, as this would yield $\mu_d = 0$ for some d, and in question 7 it is proven that for $d \geq 1$ this is impossible.

2 Simulation of SAW

2.1 A brief discussion on notation and parameter choices

Some notation is needed to fully understand the script and report. For every simulation, the number N signifies the number of trajectories simulated. In order to have decent comparisons, we chose N to be the same for all simulations. This was set to 1000. The reasoning behind this is that for the more demanding simulations, the calculations by MATLAB were slow for larger N than this. The variable n_{max} is also introduced in the beginning of the script, and is set to 40. The meaning of the variable is that it is the largest value of n (i.e. length of trajectories) simulated for each method. The reasoning behind this was that we wanted to be able to check $c_n(d)$ even for large n, as this is where SISR should show its value the most. This is of course due to the weight degeneration being stronger for large n. Note however, that for questions 6 and 9, $n_{max69} = 10$ was set, which will be explained in section 2.5. The final variable that might need some introduction is n_{runs}, which is set to 10. It is only applied in questions 6 and 9, and signifies the number of times the simulation is re-done in order to get more estimates of the parameters estimated in these questions. Once again, time complexity makes it difficult to motivate using a larger value.

2.2 Naive approach (Q3)

As a first approach to estimate the $c_n(2)$'s, a simulation of random walks was performed where they were not bound to be self-avoiding. Then, the random walks which were self-avoiding were counted and divided by the total number of simulated walks to compute the ratio between self-avoiding and all random walks, see equation 8. Then the ratio was scaled by the number of possible random walks, $k_n(d)$, see equation 9 (please note the difference from $\boldsymbol{S}_n(d)$), to form c_n, see equations 10.

$$SAW_{ratio} = \frac{N_{SA}}{N} \tag{8}$$

$$k_n(d) = |\{x_{0:n} \in Z^{d(n+1)} : |x_k - x_{k-1}| = 1, \forall k\}| = (2d)^n = [d = 2] = 4^n \tag{9}$$

$$c_n^{naive} = k_n(d) \times SAW_{ratio} \tag{10}$$

The idea behind this setup is that equation 8 gives an idea of the probability of, when picking a random walk of n steps, each of length one, it is also a SAW. I.e. equation 8 estimates $\frac{c_n(2)}{k_n(2)}$. Multiplying this probability with the actual number of walks that are not necessarily self-avoiding ($k_n(2)$), we get a naive idea of the number of possible SAW:s. Essentially, we treat $c_n(2)$ as a binomially distributed stochastic variable, the expectation of which we try to estimate.

The naive estimates of c_n were equal to the true values for $n = 1$ and $n = 2$, but divereged quickly thereafter, see table 3.

2.3 Sequential Importance Sampling (Q4)

In a second approach, all simulated walks were self-avoiding by definition, hence saving computational power since no waste samples were simulated. This was implemented by first saving all previous points of each trajectory. For each step, the possible coordinates of the next position were compared to the previous points in that trajectory, and the points which had not been visited before were defined as $free$. A matrix containing the $free$ points was created.

In the next step, a random number, u, was drawn from the uniform distribution. This number was compared to a threshold, t, depending on the number of $free$ points, s, see equation 11.

$$t = 1/s, \ where \ 0 \leq s \leq 3 \tag{11}$$

For the case when $s = 1$, there is only one free point, and X_{k+1} was naturally set to that point. For the case when there were two free points, X_{k+1} was set to the first point in the $free$ matrix if $u \leq t$, and the second if $t \leq u \leq 2t$. The same technique was used in the cases with three and four $free$ points, then with three and four intervals respectively, each of length t. Note that the only time when there are four $free$ points is in the starting point. Note also that when there were no $free$ points, i.e. $s = 0$, then X_{k+1} was set to X_k

As importance weights, the cumulative product of the number of $free$ points was used. This comes from defining $z(X_{0:n})$ as in equation 12, and the importance function, $g(X_{0:n})$, as the probability of choosing a certain step X_{k+1}, $1/s$.

$$z(x_{0:n}) = \begin{cases} 1, & if\ x_{0:n}\ is\ SA \\ 0, & if\ not \end{cases} \tag{12}$$

Then, the importance weights are defined as in equation 13, see Lecture 6 slide 20.

$$\begin{aligned} w_{n+1}^i &= \frac{z_{n+1}(X_i^{0:n+1})}{z_n(X_i^{0:n}) g_{n+1}(X_i^{n+1}|X_i^{0:n})} \times w_n^i = \\ &= \frac{z_{n+1}(X_i^{0:n+1})}{z_n(X_i^{0:n}) \times 1/s_n^i} \times w_n^i = \\ &= s_n^i \times w_n^i \end{aligned} \tag{13}$$

Note that $s_n^i = 0$ coincides with $z_{n+1}(X_i^{0:n+1}) = 0$, and that when $z_n(X_i^{0:n}) = 0$ then $z_n(X_i^{0:n}) = 0$ as well. This is why the last equality in equation 13 is always true. The implementation in Matlab ensures that no division by zero ever occurs, through handling the non-SA trajectories separately, simply setting all subsequent weights along this trajectory to zero.

The initial weight, w_0, was set to 1, and then the weights were computed recursively according to equation 13.

The estimate of c_n was then computed as in equation 14, see Lecture 7 slide 8, as it was recognize as the normalizing constant. The resulting SIS estimate of c_n was closer to its true values than was the naive estimate, see table 3.

$$c_n^{SIS} = \frac{1}{N} \sum_{i=1}^{N} w_n^i \tag{14}$$

2.4 Sequential Importance Sampling with Resampling (Q5)

In this last approach to estimating c_n, importance sampling with resampling was used. The methology is similar to that of the SIS algorithm, described above in section 2.3. The difference is that now, a resampling is performed at each time step, before letting the walks walk on. For long walks, with a high n, the risk for weight degeneration is high. This means that few trajectories will get an unreasonably high weight while others will have insignificantly small weights. The resampling aims at spreading out the trajectories on paths with higher weights in each time step, to reduce the problems of weight degeneration.

In the resampling step, the weights were normalized by the sum of the weights from all trajectories for that time step, see equation 15

$$\tilde{w_n^i} = \frac{w_n^i}{\sum_{j=1} N w_n^j} \tag{15}$$

Then, N numbers, d, between 1 and N were drawn, with replacement, with probabilities $\tilde{w}$. The drawn trajectories were then used for the subsequent steps of the SAW:s, probably yielding duplicates of the high-weight trajectories and some eliminated low-weight trajectories. In other words, $\tilde{X}_j^{0:n+1} = X_j^{0:n+1}$ with probability $\tilde{w}$, see Lecture 7 slide 19.

Finally, c_n^{SISR} was computed recursively according to equation 16, see Lecture 7 slides 22-23.

$$c_n^{SISR} = c_{n-1}^{SISR} \times \frac{1}{N} \sum_{i=1} N w_n^i \tag{16}$$

$$, \ where \ w_n^i = \frac{z_{0:n}(X_i^{0:n+1})}{z_n(X_i^{0:n-1}) g_{n+1}(X_i^{n+1} | X_i^{0:n})} = s_n^i$$

Note that the weights found in equation 16 are not updated recursively as they were in the SIS approach. This is because the historical weights are represented by the number of copied trajectories in the resampled set.

The resulting SISR estimates of c_n were equal to those estimated using the SIS approach for short walks, $n \leq 5$, see table 3. For medium length walks, $15 \leq n < 20$, the SISR approach still did not yield estimates closer to the true value than did the SIS approach. However, for large n, $35 \leq n < 40$ the SISR estimates were closer to the true value for almost all time steps, see again tabel 3.

2.5 Estimating A_2, μ_2 and γ_2 (Q6)

For the entirety of question 6, $d = 2$, however for generality, the notation in the setup will be with a general d, as this setup will be re-used in question 9. In order to estimate the parameters, we make use of the (conjectured for $d = 2$) relationship given in (3) in the instructions for this assignment. Taking the logarithm of both sides gives a relationship that is linear in n as well as $ln(n)$.

$$c_n(d) = A_d \mu_d^n n^{\gamma_d - 1} \tag{17}$$

$$\implies ln(c_n(d)) = ln(A_d) + n ln(\mu_d) + (\gamma_d - 1) ln(n) \tag{18}$$

This linear relationship suggests a multidimensional linear regression can be done. In matrix form, let:

$$Y = \begin{bmatrix} ln(c_1(d)) \\ ln(c_2(d)) \\ \vdots \\ ln(c_{n_{max}}) \end{bmatrix} \tag{19}$$

$$X = \begin{bmatrix} 1 & 1 & ln(1) \\ 1 & 2 & ln(2) \\ \vdots & \vdots & \vdots \\ 1 & n_{max} & ln(n_{max}) \end{bmatrix} \tag{20}$$

$$\beta = \begin{bmatrix} ln(A_d) \\ ln(\mu_d) \\ (\gamma_d - 1) \end{bmatrix} \tag{21}$$

Where n_{max} now set to 10, and not 40. The reason for this change is that the estimates were poor when $n_{max} = 40$ was used. A possible explanation for this might be found in table 3: for large values of n, the estimates of c_n vary very much, and may be off from the true value by as much as a power of 10^{16}. Including this extreme variation in the parameter estimates results in poor performance. After trying different approaches, the one presented below was the best-performing approach. The simultaneous estimation is done by the least square estimate:

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{22}$$

Finally, the parameters are de-transformed to obtain the final estimations. This process is repeated 10 times ($n_{runs} = 10$, see section 2.1), and the mean and variances of the estimates are seen in Table 1.

Param	Mean	Variance
A_2	1.4868	1.9741E-05
μ_2	2.6642	2.0579E-04
γ_2	1.2163	2.8140E-04

Table 1: Parameter estimations, question 6

The estimates found indicate that $V(\hat{A}_d) < V(\hat{\mu_d}) < V(\hat{\gamma_d})$. It is not very clear why, but surely it must be to do with the nature of the least square estimate. Surprisingly, for both A_d and μ_d, the estimates are actually exponents of the original least square estimates, which intuitively would yield a larger variance. It is hence notable that they still have lower variance than that of

γ_d. Since γ_d is hardly transformed, it could be interpreted as the easiest one to estimate. However, the variances indicate that $A-d$ is the easiest one, and γ_d the hardest one, as they vary the least and the most respectively. Moreover, the estimate of γ_2 is indeed quite close to the given value $\frac{43}{32}$, see table 1.

3 Parameter bounds

3.1 Bounds of μ_d (Q7)

We want to verify the bound given in question 7 in the instructions. First, we note that, for any d, the first step from the origin may be taken in $2d$ directions. For every step after the first step, at most $2d-1$ directions are possible, as the trajectory can never go back to the most recently visited point. This means that the following should hold for $c_n(d)$:

$$c_n(d) \leq (2d)(2d-1)^{n-1} \tag{23}$$

This, in combination with the definition of the connective constant μ_d, gives μ_d the upper bound:

$$\begin{aligned}\mu_d = \lim_{n\to\infty} (c_n(d))^{\frac{1}{n}} &\leq \lim_{n\to\infty} \left((2d)(2d-1)^{n-1}\right)^{\frac{1}{n}} = \\ &= \lim_{n\to\infty} (2d)^{\frac{1}{n}} (2d-1)^{\frac{n-1}{n}} = 2d-1\end{aligned} \tag{24}$$

The lower bound for μ_d is found, analogously by first finding a lower bound for $c_n(d)$. One may go about this by choosing for each of the d basis vectors, to only take positive or negative steps. In 2 dimensions, one example of this would be taking only positive steps in the x-direction, and negative steps in the y-direction. This gives the possibility of taking n steps in positive x-direction, or n-1 steps in positive x-direction and 1 step in negative y-direction and so on. This generalizes to d dimensions, and will always generate a SAW. Further, this generates d^n choices of trajectories, giving us $c_n(d) \geq d^n$. Applying this to the connective constant, we get:

$$\mu_n = \lim_{n\to\infty} c_n(d)^{\frac{1}{n}} \geq \lim_{n\to\infty} (d^n)^{\frac{1}{n}} = d \tag{25}$$

Combining equations 24 and 25, it is verified that $d \leq \mu_d \leq 2d-1$.

3.2 Bound of A_d (Q8)

The relationship which is to be proven, that $\gamma_d = 1$ for $d \geq 5$, means that where $n^{\gamma_d - 1}$ occurs, this is simply equal to one. Furthermore, as $d \geq 5$, the conjecture in (3) in the instructions is a proven equality. As hinted, the relationship is proven through plugging (3) into (1). This yields:

$$c_{n+m}(d) = A_d\mu_d^{n+m}(n+m)^{\gamma-1} = A_d\mu_d^{n+m} \tag{26}$$

$$c_n(d)c_m(d) = A_d\mu_d^n A_d\mu^m n^{\gamma-1}m^{\gamma-1} = A_d\mu_d^n A_d\mu^m = A_d^2\mu^{n+m} \tag{27}$$

$$c_{n+m}(d) \le c_n(d)c_m(d) \implies A_d\mu_d^{n+m} \le A_d^2\mu^{n+m} \implies$$

$$\implies A_d \le A_d^2 \implies A_d \ge 1 \tag{28}$$

Proving the bounds for A_d for $d \ge 5$.

3.3 Asymptotic bound of μ_d (Q9)

In order to solve this problem, the same algorithm as in question 6 was used, with a few changes in the script, making it more dynamic for varying values of d. This was run for a few different values of d, and a representative d, $d = 5$, was chosen to be presented in the results. This is because 5 is a fairly large number of dimensions, mainly concerning the asymptotic approximation of μ_d given in the instructions. Also, for large values of d, the algorithm is slow, and it was deemed good to be able to quickly check the answers presented in the report. The relationships which are to be verified in question 9 are:

$$A_d \ge 1 \tag{29}$$

$$\gamma_d = 1 \tag{30}$$

$$d \le \mu_d \le 2d-1 \tag{31}$$

$$\mu_d \approx 2d - 1 - \frac{1}{2d} - \frac{3}{(2d)^2} - \frac{16}{(2d)^3} + O(\frac{1}{d^4}) \tag{32}$$

Param	Mean
A_5	1.1309
μ_5	8.8295
γ_5	1.0349

Table 2: Parameter estimations, question 9

With the naked eye it is easy to verify that the three parameter estimations presented in Table 2 stay in their respective bounds given by (29-31). Now, it only remains to check that (32) seems reasonable.

$$2d - 1 - \frac{1}{2d} - \frac{3}{(2d)^2} - \frac{16}{(2d)^3} = [d=5] = 8.854 \approx 8.8295 = \hat{\mu_5} \tag{33}$$

Indeed, the approximation appears to hold. It is however notable that the approximation should become better for larger values of d than the chosen 5 (because of the big O-notation). For example, setting $d = 10$ gives $\hat{\mu_{10}} = 18.9214$, and the approximation evaluates to 18.9405, which indeed is closer.

	c_n			
n	Naive	SIS	SISR	True
1	4	4	4	4
2	16	12	12	12
3	64	36	36	36
4	256	100	100	100
5	1023	282	282	284
⋮	⋮	⋮	⋮	⋮
15	1.0727E9	6.1805E6	6.7395E6	6.4165E6
16	4.2907E9	1.6485E7	1.8055E7	1.7245E7
17	1.71673E10	4.4697E7	4.8568E7	4.6466E7
18	6.8651E10	1.1913E8	1.3113E8	1.2465E8
19	2.7460E11	3.1804E8	3.5590E8	3.3511E8
⋮	⋮	⋮	⋮	⋮
35	1.1794E21	2.0908E15	2.4706E15	2.2525E15
36	4.7176E21	5.3311E15	6.5941E15	5.9957E15
37	1.8871E22	1.4032E16	1.7270E16	1.5969E16
38	7.5482E22	3.7753E16	4.5713E16	4.2487E16
39	3.0193E23	1.0113E17	1.2457E17	1.1310E17

Table 3: c_n estimates for SAW, using the different methods and rounded to the nearest integer. The true values are from Slade's "Self-Avoiding Walks".