Home Assignment 2: Self-Avoiding Walks in Z^d & Filter Estimation of Noisy Population Measurements

FMSN50 - Monte Carlo and Empirical Methods for Stochastic Inference

Victoria Lagerstedt & Kajsa Hansson Willis

February 25, 2025

Contents

1 Introduction

This assignment focuses on sequential Monte Carlo methods and techniques to solve problems of uncertainty. In the first of the two parts, self-avoiding walks (SAWs) on a 2D lattice are explored. A self-avoiding walk is a path that never crosses itself and the objective is to estimate how many possible SAWs there are of a given length. Instead of trying to count all valid walks exactly, the number of such walks ia estimated using Sequential Importance Sampling (SIS) and an improved version called SIS with Resampling (SISR). Using these estimates, some key parameters related to the behavior of long self-avoiding walks, including the connective constant are also approximated.

In the second part 2, instead a filter estimation of noisy population measurements is conducted with a population size model that evolves over time with some randomness. To estimate the true population size from the noisy observations, a particle filter is used. Here, many possible population trajectories (particles) are simulated, and adjusted for their likelihoods based on the measurements, and then resampled to stay focused on the most likely population paths.

2 Part 1: Self-avoiding walks in Z^d

2.1 Problem 1

In this first problem the aim is to show that the inequality in Equation 1 holds for all n $\geq$ 1 and m $\geq$ 1.

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

$c_n(d)$ is the possible number of SAW of a given length n, $|S_n(d)|$, and $c_{n+m}(d)$ is the possible number of SAW of length n+m. The inequality says that the number of SAWs of length n+m in a d-dimensional space is smaller than or equal to the number of SAW of length n in a d-dimensional space multiplied with the number of SAW of length m in another separate d-dimensional space. Instead of continuing on the SAW where multiple of the positions has already been visited and the number of possible self-avoiding steps has decreased from the start a new SAW is started in an empty d-dimensional space with no visited positions. There is, therefore, naturally fewer, or equal, possible SAW if doing one longer SAW of length n than multiple smaller self-avoiding walks where the sum of the lengths is the same as the length of the longer SAW. The authors can thus be considered convinced of the legitimacy of the claim.

2.2 Problem 2

For each step in a self-avoiding random walk, there are x possible moves. This value ranges between four and zero, depending on if the surrounding steps have already been visited or not. An average of this value converges to a certain value when the steps go towards infinity in a SAW problem.

As described before, $c_n(d)$ is the total number of SAWs that can possibly be made in dimension d with n steps. This value is the number of possible self-avoiding moves for each step multiplied with each other. For SAW with two steps, this value is 4*3=12 since the first step always has 4 possible self-avoiding steps and the second always has 3. The average number of possible self-avoiding steps in a SAW of length n is therefore n:th root of $c_n(d)$. When n tends towards infinity, the value of the average number of possible self-avoiding steps converges to the value μ_d, which can be interpreted as the geometric mean of the number of neighbors not yet visited along a SAW.

In problem 2, the aim is to use Fekete's lemma to prove this limit, described in Equation 2, exists.

$$\mu_d = \lim_{n \to \infty} c_n(d)^{1/n} \tag{2}$$

Fekete's lemma states that for every subadditive sequence $(a_n)_{n\geq 1}$ the limit $\lim_{n \to \infty} a_n/n$ exists and is equal to $\inf_{n\geq 1} a_n/n$. A subadditive sequence is a sequence where $a_{m+n} \leq a_m + a_n$. The subadditive property was determined in Problem 1 in this case.

If the logarithm of Equation 1 is obtained, it is a subadditive sequence, as is visualized in Equation 3.

$$\ln c_{n+m}(d) \leq \ln c_n(d) + \ln c_m(d) \tag{3}$$

Applying Fekete's lemma to this subadditive sequence, it is clear that the limit described by Equation 4 exists.

$$\lim_{n \xrightarrow{} \infty} \frac{1}{n} \ln c_n(d) \tag{4}$$

Equation 4 can be further rewritten as Equation 5 using the rules of logarithms which corresponds to the limit which existence was to be proven.

$$\lim_{n \xrightarrow{} \infty} \ln c_n(d)^{\frac{1}{n}} \tag{5}$$

2.3 Problem 3

In problem 3, a naive approach is considered where $c_n(2)$ is estimated using the sequential importance sampling (SIS) algorithm with the instrumental distribution (g_n) as a standard random walk, neglecting the self-avoidance property.

It is known that the amount of self-avoiding walks can be described by Equation 6, but here a simpler estimation approach is considered.

$$c_n(d) \sim \begin{cases} A_d \mu_d^n n^{\gamma d - 1}, & d = 1, 2, 3, \quad d \geq 5 \\ A_d \mu_d^n \log(n)^{1/4}, & d = 4 \end{cases} \quad \text{as } n \to \infty \tag{6}$$

This is done by simulating a large amount of particles, $N = 10^5$ samples was determined to be reasonable, with n steps taken (the number of positions is thus n+1 since it starts at (0,0)), and having them be decided randomly. One vector of zeroes was created for each of the X- and the Y-positions in the grid in Matlab. These were of dimensions (n+1,N). A variable is also created to keep count of the amount of non-self-avoidant walks. A boolean was inserted into every sample that was true at first, but if duplicates were identified in the path of the walk it was set to false.

For every step of the duration of the walk, a number between 1 and 4 was randomly selected. This number then corresponded to a direction that the walk would move in either the X- or the Y-plane. This position was then compared to the previous paths of the walk. If the X- and Y-coordinates were the same as a previous step in the vector, the boolean was set to false. If the boolean was set to false after the n steps of the walk had been completed, the counter went up by one.

After all the walks had been completed, the number of self-avoidant walks, N_{SA}, were calculated by subtracting the number of non-self-avoidant walks from the total number of walks (N). The ratio N_{SA}/N was then calculated to obtain the ratio of self-avoidant walks. This was then multiplied by the amount of possible random walks, 4^n, to obtain an estimate of the number of self-avoidant walks in 2 dimensions, $c_n(2)$, which can be seen in Equation 7. The estimations and correct values for walks up to 10 steps can be observed in Table 1. The true values are displayed beside for reference. [1]

$$c_n(2) = \frac{N_{SA}}{N} * 4^n \tag{7}$$

Number of steps:	Estimated c_n:	True c_n:
1	4.00	4
2	12.03	12
3	36.05	36
4	100.02	100
5	284.35	284
6	780.25	780
7	2 168.09	2 172
8	5 829.43	5 916
9	16 336.81	16 268
10	44 585.45	44 100

Table 1: The amount of possible $c_n(2)$ walks from the naive estimate from Problem 3 as well as the true theoretical value of self-avoiding walks, for n = {0,1, 2, ..., 10}

From the table, it is apparent that the estimates are very good for short walks, and it follows extremely closely up to n=6. After that, the estimates become increasingly worse. This estimation method is not very efficient since it results in a large amount of unnecessary random walks that are not self-avoidant, and a large number of particles is required to create a fair estimate. For larger values of n, it is not possible to run all possible walks and there will be randomness that gives rise to variance in the estimation.

2.4 Problem 4

In the fourth problem, the naive approach is improved to allow only self-avoidant walks among the samples. This is done by using sequential importance sampling (SIS) with an instrumental distribution g_n. In this case, g_n is the distribution of self-avoidant walks in the two-dimensional plane starting from origo. A good choice for an instrumental distribution when using SIS is a distribution that satisfies Equations 8 and 9. From these equations the conclusion that a good instrumental distribution has to be the same distribution for every step can be drawn.

$$g_{n+1}(X_{0:n+1}) = g_{n+1}(X_{n+1}|X_{0:n})g_{n+1}(X_{0:n}) \tag{8}$$

$$g_{n+1}(X_{0:n+1}) = g_{n+1}(X_{n+1}|X_{0:n})g_n(X_{0:n}) \tag{9}$$

For each particle, n self-avoiding steps (in this case n=10) were taken by saving each passed position and only making moves to unvisited positions. The number of possible self-avoiding next steps for each step, that is the number of free neighbor positions, was saved in a vector called the z-vector. If the particle had no possible self-avoiding steps, the rest of the z-vector for this particle was filled with zeros and the loop for the particle's walk was exited.

When updating the weights in the loops in SIS, equation 10 is used. In this problem, the z function is an indicator function and the g function is 1 divided by the z-vector. The indicator function is one when there are possible moves and zero when there are no possible moves left. The case when there are no possible moves is handled earlier in the code, and the weight is then set to zero. Because of this, the z function in this problem is always set to one. Since the weight is always the last weight multiplied by some value, the weight will continue to be zero for all infinity once one weight becomes zero.

$$\omega^i{}_{k+1} = \frac{z_{k+1}(X_i{}^{0:k+1})}{z_k(X_i{}^{0:k})g_{k+1}(X_i{}^{k+1}|X_i{}^{0:k})} \times \omega^i{}_k \tag{10}$$

Using SIS is a better method for calculating the amount of SAWs than the method used in Problem 3 in section 2.3. The method in Problem 3 creates many unnecessary walks which is computationally expensive and requires much memory. Using SIS does not require as much memory and time.

One issue with using sequential importance sampling is weight degeneracy. Weight degeneracy is when one weight is close to one and all the other almost zero. This causes the approximation to be based on only one particle, which does not give a very good approximation. [2] One solution to this problem is to introduce resampling in the sequential importance sampling.

Once the weights are calculated, they can be used to estimate the $c_n(2)$ value according to Equation 11 by taking the mean of the weights for each step of the path.

$$c_n(2) = \frac{1}{N}\sum_{i=1}^{N}\omega_n^i \tag{11}$$

The results from this method up to n=10 can be observed in Table 2 with the true values for comparison. Compared to the previous values in Table 1, the early estimates are more precise, and the estimations for growing path lengths are also better for most values, but with some exceptions. The precision of the estimates has increased from the naive approach and is expected to increase even more as the n-value grows, additionally, the approach is more computationally efficient. However, it is important to note that too few values are considered to be able to prove this relationship.

Path length:	SIS estimate:	True c_n:
0	1	1
1	4	4
2	12	12
3	36	36
4	99.97056	100
5	284.02812	284
6	779.94144	780
7	2 165.09436	2 172
8	5 904.71532	5 916
9	16 191.68616	16 268
10	43 927.99452	44 100

Table 2: The amount of possible $c_n(2)$ walks from the SIS estimate from Problem 4 as well as the true theoretical value of self-avoiding walks, for n = {0,1, 2, ..., 10}

2.5 Problem 5

In the fifth problem, resampling was introduced to avoid the weight degeneracy problem. This was done using sequential importance sampling with resampling (SISR). SISR is performed in the same way as SIS, in problem 4, but introducing resampling after each step. Resampling means, in simple words, that the particles with low weights are removed and the particles with higher weights are multiplied, as the weight corresponds to the probability of the sample's existence. After resampling there are the same amounts of particles as before resampling, but the positions of the new particles are, for some, different from before resampling. This prevents the weight degeneracy problem, since there will never be only one particle with a high weight, and the particles with a weight close to zero will be exchanged.

The resampling was performed using the Matlab command *randsample* with the weight vector as input, which gave the new particles, at the end of each step. The values produced by this command were then used to get the new positions of the particles by changing the x- and y-vectors to only be the positions of the new particles.

The weight vector is updated slightly differently than when using SIS without resampling. Instead of always multiplying by the last weight vector, a new weight vector is created for each step. See Equation 12, where X is the particles before resampling, $\tilde{X}$ is the particles after resampling, z and g are the same as in problem 4, and ω is the weights.

$$\omega^i{}_{k+1} \leftarrow \frac{z_{k+1}(X_i{}^{0:k+1})}{z_k(\tilde{X}_i^{0:k}) g_{k+1}(X_i{}^{k+1} | \tilde{X}_i^{0:k})} \tag{12}$$

When the weights have been calculated the amount of SAWs can be estimated with the help of Equation 13.

$$c_{N,n}^{SISR} \approx \Pi_{k=0}^{n} \left(\frac{1}{N} \sum_{i=1}^{N} \omega_k^i\right) \tag{13}$$

The results of this calculation are displayed in Table 3. It is clear that the first estimates are very precise, but they successively get worse, especially for n=10 where there is a considerable jump in prediction prowess.

Comparing the results of the SISR approach to the SIS and naive, it is apparent that the estimates are marginally worse for the first few path lengths. However, for increasing n, the estimates become better. This is in line with theory, as the resampling should improve accuracy for higher values of n as the most common paths should be represented to account for that fact. The supremacy of the SISR approach is expected to continue for path lengths exceeding the ones examined in this paper.

2.6 Problem 6

In this section, the aim is to estimate the constants A_2, μ_2 and γ_2 using the SISR estimates of $c_n(2)$ that were calculated in part 2.5. In order to estimate the constants easier, the relationship described

Path length:	SISR estimate:	True c_n:
0	1	1
1	3.99997	4
2	11.99983	12
3	35.99925	36
4	100.00447	100
5	284.04671	284
6	780.12862	780
7	2 171.839	2 172
8	5 919.673	5 916
9	16 263.9468	16 268
10	43 999.9939	44 100

Table 3: The amount of possible $c_n(2)$ walks from the SISR estimate from Problem 5 as well as the true theoretical value of self-avoiding walks, for n = {0,1, 2, ..., 10}

by Equation 6 can be logarithmized so that it for all reasonable values of the dimension d except 4, follows Equation 14. However, it is only the case d=2 that is explored in this problem.

$$\ln\, c_n(d) = \ln\, A_d + n * \ln\, \mu_d + (\gamma_d - 1) * \ln\, n \tag{14}$$

A linear regression can be conducted then, with the set-up described in Equation 15. This regression will find the most appropriate coefficients for the n different equations.

$$y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \tag{15}$$

Combining Equations 14 and 15 are for all n set to

$$y = ln(c_n(2),\ x_1 = 1,\ x_2 = n, x_3 = ln(n) \tag{16}$$

and

$$\beta_1 = ln(A_2),\ \beta_2 = ln(\mu_d), \beta_3 = \gamma_d - 1 \tag{17}$$

Using the Matlab feature *regress* for linear regressions, the beta coefficients were estimated to the following:

$$\beta_1 = -0.8411,\ \beta_2 = 0.8458, \beta_3 = 0.7429$$

This yields the linear regression described in Equation 18.

$$y = -0.8411 + 0.8458n + 0.7429 * ln(n) \tag{18}$$

The constants can then be calculated using the relationships in Equation 17. A_2 is determined by Equation 19. μ_2 is determined by Equation 20. γ_2 is determined by Equation 21.

$$A_2 = exp(-0.8411) = 0.4312 \tag{19}$$

$$\mu_2 = exp(0.8458) = 2.3299 \tag{20}$$

$$\gamma_2 = 1 + 0.7429 = 1.7429 \tag{21}$$

This estimation was redone in several independent replicates to understand how the estimates vary. The c_n estimates for each length of n can be viewed in Table 4. The above process was conducted for each of the replicates.

The results of the coefficient estimates for the replicates as well as the mean can be viewed in Table 5.

The variance of A_2 is $2.667 * 10^{-8}$, the variance of μ_2 is $2.7322 * 10^{-6}$ and the variance of γ_2 is $3.2403 * 10^{-6}$. Clearly, the variance is the lowest for A_2, which indicates that it is the easiest to estimate and has the highest precision. The difference between the two other coefficients is negligible. A_2 can be viewed as the intercept term for the logarithmized function, whereas the other coefficients represent growth factors that are more difficult to estimate.

Replicate number:	Path length 0:	Length 1:	Length 2:	Length 3:	Length 4:	Length 5:	Length 6:	Length 7:	Length 8:	Length 9:	Length 10:
1	1	3.99997	11.99983	35.99925	100.001596	284.28953	781.327154	2175.066345	5915.76719	16271.79093	44138.86009
2	1	3.99997	11.99983	35.99925	100.017796	284.04554	779.625475	2168.364539	5896.65053	16200.28076	43894.33673
3	1	3.99997	11.99983	35.99925	100.004476	284.04671	780.128621	2171.839075	5919.67317	16263.946876	43999.99395
4	1	3.99997	11.99983	35.99925	99.859399	283.84834	780.301941	2176.566433	5929.24991	16330.933051	44297.49259
5	1	3.99997	11.99983	35.99925	99.873439	283.753424	778.528595	2167.252334	5907.79981	16235.5201	43992.25232
6	1	3.99997	11.99983	35.99925	99.965957	283.84034	779.777536	2171.727225	5917.78295	16276.21105	44080.21135
7	1	3.99997	11.99983	35.99925	99.940757	284.05661	779.425797	2172.134989	5922.71703	16273.90882	44186.59175
8	1	3.99997	11.99983	35.99925	100.042995	284.12811	780.050997	2173.034865	5923.14978	16287.77343	44172.93019
9	1	3.99997	11.99983	35.99925	99.937157	283.99241	780.479331	2173.353965	5924.47597	16288.45802	44120.70913
10	1	3.99997	11.99983	35.99925	99.927078	283.63002	778.348848	2167.180048	5903.03003	16229.55465	43960.99468

Table 4: Independent replicates of the SISR estimates for path lengths up to 10 from Problem 6

Replicate number:	A_2	Mu_2	Gamma_2
1	0.402894960624457	2.55972799919368	1.51991617754476
2	0.403243716204994	2.55674129649079	1.52268021358660
3	0.403002021846066	2.55887582506061	1.52061520185367
4	0.402736885897704	2.56294247288790	1.51576696107976
5	0.403164333406588	2.55900079052743	1.51958121823629
6	0.402981094561282	2.55998686611404	1.51904758122069
7	0.402926279674726	2.56089693138051	1.51798408266892
8	0.402893882182648	2.56058165337143	1.51867633146774
9	0.402890426284179	2.56046945954132	1.51874654509349
10	0.403211210911860	2.55854748067677	1.52005916939427
Mean	0.4030	2.5598	1.5193

Table 5: Estimates of A_2, μ_2 and γ_2 from the independent replicates, as well as the mean from the ten replicates

2.7 Problem 7

In this part, the general bound in Equation 22 is explored. However, it is wise to first verify the bound for the estimate from Problem 6. For dimension two, the estimate of μ_2 should be between 2 and 3. It is apparent from Table 5 that all estimates of μ_2 including the mean, lie between the appropriate bounds. The general case can then be examined.

$$d \leq \mu_d \leq 2d - 1 \tag{22}$$

μ_d is called the connectivity constant and it can be interpreted as the geometric mean of the number of unvisited neighbors along a self-avoiding walk, as previously stated.

There are always $2 * d$ possible steps to take if it is a random walk without self-avoidance, provided that it is a lattice random walk. This is because it can move one step in either a positive or a negative direction along the plane. If the dimension is, for example, two, there are always four possible steps to take. When doing a SAW there will always be one direction that is not possible for all steps except the first, as it cannot go back to where it came from. Therefore the maximum amount of SAWs must consider the fact that it is not possible to move back to the position that was already visited. If the number of walks goes towards infinity the first move, which had $2 * d$ possible self-avoiding walks, will be statistically insignificant. Therefore, the maximum of μ_d is $2 * d - 1$.

The lower bound of the inequality is d which is the number of dimensions. If a particle is only able to move in positive directions, all the walks will be self-avoiding. The number of possible SAW that can be made with the particle only moving in positive direction is smaller or equal to the total number of possible SAW, see equation 23, because only moving in positive directions is limiting the possible SAWs. Only moving in a positive direction gives d^n possible moves for each step.

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

Taking the n:th root of both sides of the inequality gives the equation: $d \leq c_n(d)^{1/n}$. If n goes towards infinity, the right side of the inequality goes towards μ_d and the left side stays the same, see Equation 24.

$$d \leq \lim_{n \to \infty} c_n(d)^{1/n} = \mu_d \tag{24}$$

Therefore, the general bound in Equation 22 is considered verified.

2.8 Problem 8

In problem 8, the general bound described by Equation 25 is analyzed.

$$A_d \geq 1 \text{ for } d \geq 5 \tag{25}$$

To solve this, the relationship in Equation 6 is plugged into Equation 1 to obtain the relationship described by Equation 26 for $d \geq 5$, leveraging the fact that $\gamma_d = 1$ for $d \geq 5$.

$$A_d \mu_d^{n+m} (n+m)^{1-1} \leq A_d \mu_d^n n^{1-1} * A_d \mu_d^m m^{1-1} \tag{26}$$

This can be written as in Equation 27 and further simplified to Equation 28

$$A_d \mu_d^{n+m} \leq A_d^2 \mu_d^{n+m} \tag{27}$$

$$1 \leq |A_d| \tag{28}$$

Assuming A_d is positive, which is quite reasonable given that it is a part of the estimate of an amount of paths which cannot take on negative values. It is known that the connective constant is positive from Problem 7 and the length of the paths cannot be negative, alas A_d must be positive. Therefore, it is true that $A_d \geq 1$ for $d \geq 1$ and the general bound is verified.

2.9 Problem 9

In this section, the objective is to apply the SISR approach estimate of A_d, μ_d and γ_d similarly to Problem 6 for some $d \geq 3$ and compare it to the bounds in Problem 7 and 8, as well as the asymptotic bound described by Equation 29.

$$\mu_d \sim 2d - 1 - 1/(2d) - 3/(2d)^2 - 16/(2d)^3 + O(1/d^4) \tag{29}$$

For the case of d=3, the logarithmized function can be described by Equation 30. The values of $c_n(3)$ must then be estimated. This is done by using the method from Problem 5 as a blueprint but expanding the dimension and adding an additional plane that the path can move along.

$$\ln\, c_n(3) = \ln\, A_3 + n * \ln\, \mu_3 + (\gamma_3 - 1) * \ln\, n \tag{30}$$

The regression produced the values: $A_3 = 0.2227$, $\mu_3 = 4.5711$, and $\gamma_3 = 1.3123$, when run on simulations of SAW with length 9. Multiple simulations and regressions were made, but the result was around the same each time.

According to equation 22, the value of μ_3 should be greater than 3 and smaller than 2*3-1=5. For all the simulations run this was the case. The value of A_3 is less than 1 which does not satisfy the inequality in Equation 25. Since the inequality only holds for dimensions larger than 5 this result was not surprising.

Using Equation 31 to calculate μ_3 gave the result: $\mu_3 \approx 4.676$. This value is a good approximation of μ_3, with only a difference of about 0.15 between the approximation of μ_3 and the μ_3 from regression. The value of the approximations also satisfies the inequality in Equation 22.

$$\mu_3 \sim 2 * 3 - 1 - 1/(2 * 3) - 3/(2 * 3)^2 - 16/(2 * 3)^3 + O(1/3^4) \tag{31}$$

The error term $O(1/d^4)$ shows that there will be a difference between the real value of μ_3 and the approximated value. Another possible cause of the difference is the fact that a walk of length 9 was used in the simulations. To get an even more exact value from the regression, the length of the walks should go towards infinity. This small discrepancy can also be seen in the fact that the value of μ_3 slightly changed each simulation.

3 Part 2: Filter estimation of noisy population measurements

In part 2 of this exercise, a population was considered and its relative population size was estimated with theory about hidden Markov models (HMMs).

3.1 Problem 10

In the first part of the problem, the filter expectation was estimated. The relevant coefficients and distributions were given from the problem instructions so the problem could be set up with Equation 32 where Equation 33 describes X_0 and Equation 34 describes the observation density. In this case, X_k denotes the relative population size in generation k, R is the stochastic reproduction rate, and Y is the observed event in the Markov chain that is available in a vector from the imported file "population_2024.mat".

$$X_{k+1} = R_{k+1}X_k(1X_k) \quad , R_{k+1}U(0.8, 3.8), iid, k = 0, 1, 2, \tag{32}$$

$$X_0 \in U(0.6, 0.99) \tag{33}$$

$$Y_k|X_k = x \quad \in U(0.8x, 1.25x). \tag{34}$$

The filter mean τ_k is computed through Equation 35 where ω is the weight.

$$\tau_k \approx \sum_{i=1}^{N} \frac{\omega_k^i}{\sum_{l=1}^{N} \omega_k^l} \phi(X_i^{0:k}) \tag{35}$$

3.1.1 Part A

To do this in Matlab, empty vectors were created to store the filter means and the weights. Additionally, a function was created to extract values from the observation density, described by Equation 34.

A loop was created so that the estimations of X were updated correctly in each iteration, either based on Equation 33 or 32. The weighting of the iteration was then defined according to the observation density function previously described. The filter means were then computed according to Equation 35. The values of τ for each iteration were then plotted together with the real values from the data. This process was redone for different numbers of samples, with N=500, 1000, and 10,000. The plot for 500 samples can be seen in Figure 1. The plot for 1 000 samples can be seen in Figure 2. The plot for 10,000 samples can be seen in Figure 3.

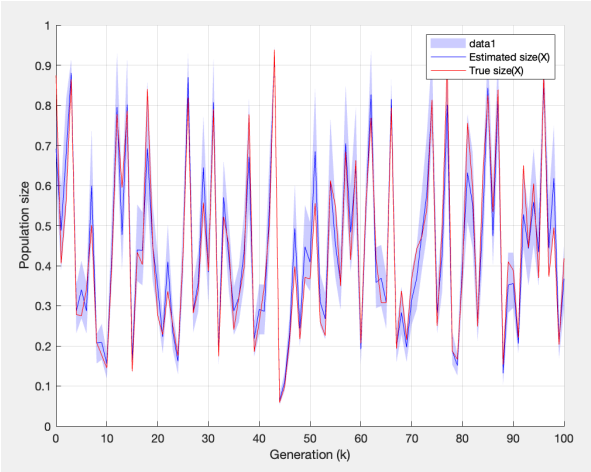

Figure 1: The estimated relative population size for the different generations with confidence intervals from 500 samples plotted with the true values

When comparing the graphs from the different amounts of samples, it is apparent that they are quite similar. Only very small differences can be made out, where the plot with 10,000 samples appears marginally better. However, this slight difference may not justify the added computational requirements that additional samples contribute. Therefore, using 500 samples is very reasonable.

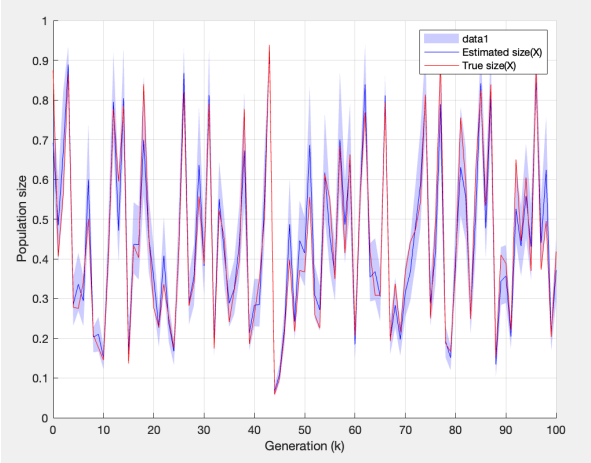

Figure 2: The estimated relative population size for the different generations with confidence intervals from 1 000 samples plotted with the true values

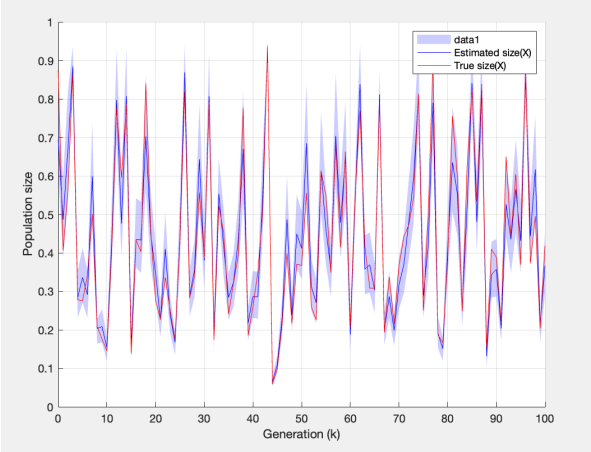

Figure 3: The estimated relative population size for the different generations with confidence intervals from 10,000 samples plotted with the true values

3.1.2 Part B

In the next part of problem 10, a point-wise confidence interval was added. This was done by sorting the particles by their value and computing the cumulative normalized weights with the help of the Matlab function *cumsum*. The indices for the 2.5% and 97.5% quantiles were then identified with the Matlab *find* function. These values were then stored in vectors and later plotted.

These confidence intervals are the purple part of Figure 1, 2 and 3. These are the graphs with 500, 1,000 and 10,000 samples, respectively. It is apparent that the visible difference is a very small. To get an understanding of the difference, the sum of the upper levels and the sum of the lower levels were computed. These values can be observed in Table 6. The difference between the sums is the lowest in the case of 500 samples, which is the lowest amount of samples. However, this difference is very small.

Number of samples:	Sum of lower confidence levels	Sum of upper confidence levels	Difference
500	37.5589	53.5197	15.9608
1,000	37.5285	53.5210	15.9925
10,000	37.4918	53.6012	16.1094

Table 6: The sums of the upper bounds of the confidence intervals and the sums of the lower bounds of the confidence for the 100 generations for the estimates based on different amounts of samples, and the difference between them to illustrate the integral of the widths of the confidence intervals

Further, it is clear that most points fall within the confidence interval, so intuitively the 95% confidence interval works well with the data set. As the difference is so low, it would be sufficient and more time-efficient to use 500 samples, which is the lowest amount of samples considered. To get an understanding of how the confidence interval would shape itself for a lower amount of samples, N=80, was also tested, and the results can be viewed in Figure 4.

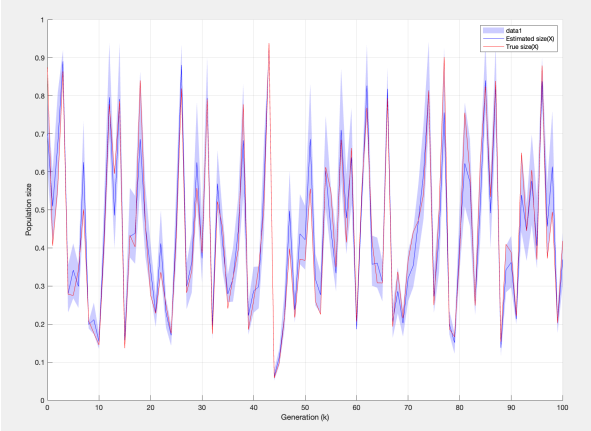

Figure 4: The estimated relative population size for the different generations with confidence intervals from 80 samples plotted with the true values

In this case, the estimation appears to be worse with a larger spread of the confidence interval. This indicates that 500 samples is a reasonable implementation.

4 Final words

In this assignment, self-avoiding walks in the Z^d were explored along with a filter estimation of noisy population measurements – the common denominator being sequential importance sampling (SIS). The former concentrating on the benefits of SIS and SIS with resampling compared to each other and naive estimations and the latter concentrating on hidden Markov models. However, both showcasing the relevance of sample sizes and efficiency differences between different methods, particularly with large sample sizes.

The results with two decimal points from the first part of the assignment are collected in Table 7, where the SISR estimate seems to be the best based on the path lengths considered.

Path length:	True c_n:	Naive estimate:	SIS estimate:	SISR estimate:
0	1	1	1	1
1	4	4	4	4
2	12	12.03	12	12
3	36	36.05	36	36
4	100	100.02	99.97	100
5	284	284.35	284.03	284.05
6	780	780.25	779.94	780.13
7	2172	2 168.09	2 165.09	2 171.84
8	5916	5 829.43	5 904.72	5 919.67
9	16 268	18 336.81	16 191.69	16 263.95
10	44 100	44 585.45	43 927.99	43 999.99

Table 7: Comparison of the true value of c_n with the naive estimate, the SIS estimate and the SISR estimate

Overall, this assignment has provided valuable insights into Monte Carlo-methods and their practical applications. It has reinforced the significance of choosing appropriate estimation techniques and

demonstrated how different strategies can enhance the accuracy and efficiency of probabilistic modeling and data-driven decision-making.

All MATLAB implementations, including the main script (proj2.m) and supporting files have been submitted in CANVAS and this report has been uploaded in PDF format to CANVAS, as well.

References

[1] Iwan Jensen (2013) "A new transfer-matrix algorithm for exact enumerations: self-avoiding walks on the square lattice". https://arxiv.org/pdf/1309.6709

[2] A. Wigren, L. Murray, & F. Lindsten, "Improving the particle filter in high dimensions using conjugate artificial process noise", IFAC-PapersOnLine, Volume 51, Issue 15, 2018, Pages 670-675, ISSN 2405-8963. https://doi.org/10.1016/j.ifacol.2018.09.207.