Literature Review of Numerical Simulation of Stochastic Differential Equation

M. Sc. Sagnik Sarkar
Research in Computer and Systems Engineering
Technische Universität Ilmenau
Supervisor: Prof. Armin Zimmermann

June 9, 2024

1 Introduction

This report is a literature review on the Numerical Simulation of Stochastic Differential Equation. The primary purpose of this report is to comprehend the idea of Brownian Motion and interpret the idea numerically using different integrals namely Ito and Stratonovich Integrals.

2 Brownian Motion

A Scalar standard Brownian Motion [2] or Wiener process over [0, T] is a random variable W(t) that depends continuously on t ϵ [0, T] and satisfies following three conditions:

1. W(0) = 0 (with probability 1). This corresponds to the idea that the first Wiener value in a Brownian Motion has to be 0 or in other words values in Brownian Motion starts from 0.

2. For 0 $\leq$ s $\leq$ t $\leq$ T, increment W(t) - W(s) is normally distributed with mean value zero and variance t-s. Numerically it is also possible to define $W(t) - W(s) \approx \sqrt{t-s}N(0,1)$. Here N(0, 1) is a normally distributed random variable with zero mean and unit variance.

3. For 0 $\leq$ s < t < u < v $\leq$ T, the increments W(t) - W(s) and W(u) - W(v) are independent.

3 Brownian path Simulation

Listing 1: Brownian Path Simulation

```
1  randn('state', 100)
2  T = 1; N = 500; dt = T/N;
3  dW = zeros(1, N);
4  W = zeros(1, N);
5
6  dW(1) = sqrt(dt)*randn;
7  W(1) = dW(1);
8  for j = 2:N
9      dW(j) = sqrt(dt)*randn;
10     W(j) = W(j-1) + dW(j);
11 end
12
13 plot([0:dt:T], [o, W], 'r-')
14 xlabel('t', 'FontSize', 16)
15 ylabel('W(t)','FontSize',16, 'Rotation', 0)
```

In *Listing 1*, a basic Brownian motion path is simulated in Matlab. In Line #1, a *randn* function is defined with seed values as 100. In Matlab, *randn* function generates numbers from standard Normal Distribution (i.e. mean=0, Standard Deviation=1). The state parameter allows to specify the internal state of the generated random numbers. Subsequent execution of *Listing 1* will produce the same output

In Line #2, few variables are initialized which will help to simulate the Brownian Motion path. T refers to the final time limit of the motion. N defines the number of steps that will be used for the simulation. dt is the individual step size derived by dividing *T/N(1/500)*.

In Line #3 and Line #4, two arrays are created with index upto N(500) and with fill value zero. Unlike other programming languages, the indices in Matlab starts from 1 instead of 0. So, the first value is calculated separately in Line #6 and assigned to the first index of Wiener variable W in Line #7. Following the Second principle of Brownian Motion, the numbers from *randn* are scaled by $\sqrt{dt}$ and used as increments in the for loop. The next elements of the Wiener variable W(j) is calculated by adding the previous element W(j-1) and the increment dW(j). This follows the 2nd principle of Wiener process which states $W(t) - W(s) \approx \sqrt{t-s}N(0,1)$. Here t and s are different time steps where t>s.

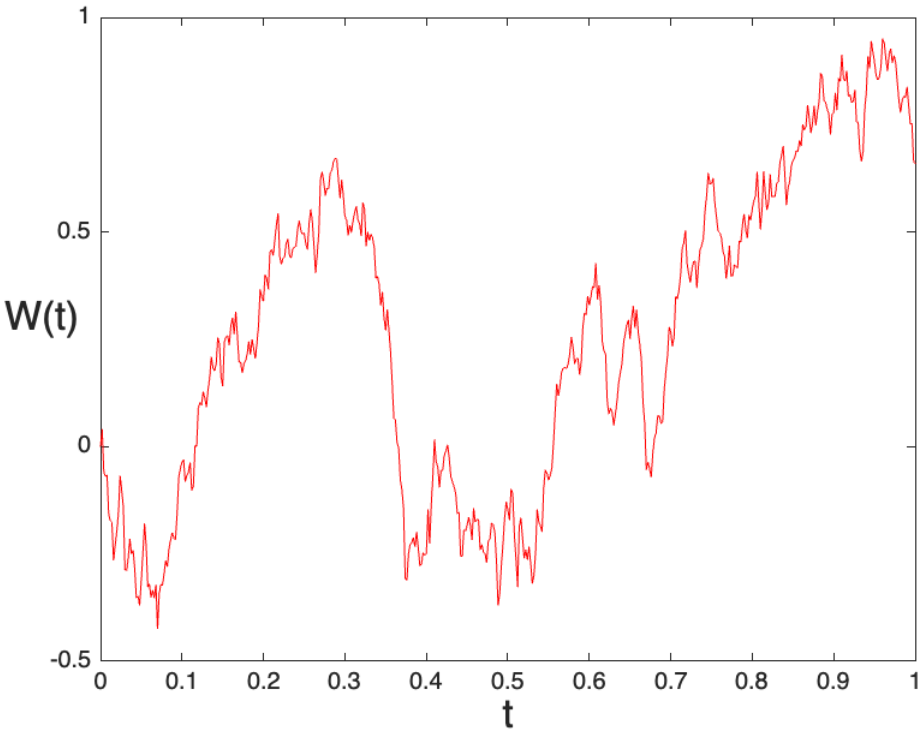

Figure 1: Brownian Path Simulation

The above figure defines a Brownian Path Simulation:

Now, the simulation can be written more efficiently and elegantly using *vectorization* in Matlab instead of writing for loops. For loops can be much slower for large datasets and computation than vectorization. In *Listing 2*, vectorized accumulation function *cumsum* is used for the Brownian Path simulation which will produce the same result as *Listing 1*.

Listing 2: Brownian Path Simulation

```
%Brownian path simulation: Vectorized
randn('state', 100) %set the state of randn
T = 1; N = 500; dt = T/N;

dW = sqrt(dt)*randn(1, N); %increments
W = cumsum(dW) %cumulative sum or integral of all the
    values

plot([0:dt:T],[0,W],'r-') % plot W against t
```

```
xlabel('t','FontSize',16)
ylabel('W(t)','FontSize',16,'Rotation',0)

%This code is much more efficient and readable than
   the for loop code in
%Bpath1
```

Now in *Listing 3*, the mean of 1000 Brownian paths and the difference between the mean path and the theoretical solution is calculated. In this snippet till line #4, everything is same as the above listings except a new array t is created with initial value dt, step size dt and final values equal to 1. In Line #4, dW matrix is created with shape 1000*500 and random numbers multiplied with $\sqrt{dt}$. In Line #5, a new matrix is created which stores the sum of subsequent elements along the second dimension effectively creating the Wiener process integrating the Wiener increments. Below image shows a typical working of a cumsum function along a second dimension:

```
A = [1 3 5; 2 4 6]
```

```
A = 2×3

     1     3     5
     2     4     6
```

Find the cumulative sum of the rows of A. The element B(3) is the sum of A(1) and A(3), while B(5) is the sum of A(1), A(3), and A(5).

```
B = cumsum(A,2)
```

```
B = 2×3

     1     4     9
     2     6    12
```

Figure 2: Cumsum Function

Now in Line #6, *repmat* function is used, which repeats the t vector M(1000) times along with Wiener Process W. After that exponential function *exp* is used on each element in the matrix. U creates matrix of shape 1000*500 with t vector repeated M(1000) times. In this we have M sample paths with t time points.

Listing 3: Mean of 1000 Brownian Paths

```
randn('state', 100);
T = 1; N = 500; dt = T/N; t = dt:dt:1;
M = 1000;
dW = sqrt(dt)*randn(M, N);
W = cumsum(dW, 2);
U = exp(repmat(t, [M, 1]) + 0.5*W);
Umean = mean(U);
plot([0, t], [1, Umean], 'b-'), hold on
plot([0, t], [ones(5, 1), U(1:5,:)], 'r--'), hold off
```

```
10 xlabel('t', 'FontSize', 16)
11 ylabel('U(t)', 'FontSize', 16, 'Rotation', 0, '
       HorizontalAlignment', 'right')
12 legend('mean of 1000 paths', '5individual paths')
13 averr = norm((Umean - exp(9*t/8)), 'inf');
```

In Line #7 we, mean is calculated for all the sample paths along each time point. Hence, the mean creates an array of shape 1*500. Line # 8 plots the mean path in blue and Line #8, plots first five individual paths in red dashed line and the subsequent lines puts the labels and legend in the graph. Finally, the graph of the mean path appears to be smooth since it can be proved that the mean of the Brownian path appears to be *exp(9*t/8)* [1]. Now in line #13, the maximum discrepancy is calculated between the mean path and the theoretical solution *exp(9*t/8)*. *norm* function along with *inf* parameter helps to calculate the *infinity norm* which basically means the absolute maximum difference.

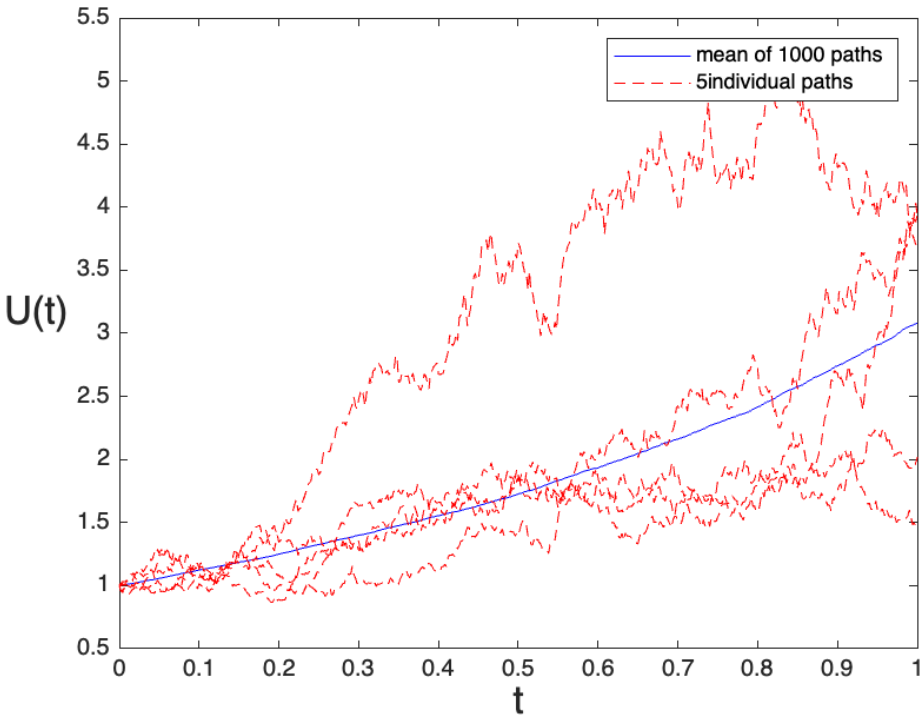

Figure 3: Mean Calculation Graph

The *averr* value appears to be 0.0504. But if we calculate the mean with more number of sample paths then the error decreases consequently. As from the graph we can see that the mean passes through almost the theoretical value. For instance at time t=0.5, the value of exp(9t/8) is 1.7551. If M(sample paths) is increased to 4000, then the *averr* value reaches 0.0268.

4 Stochastic Integrals

In the context of Stochastic Differential Equations, two types of Stochastic Integrals are commonly employed: the Itô Integral and the Stratonovich Integral [3]. In definite Integral, given a suitable function h, the integral $\int_0^T h(t)dt$ can be approximated by the Reimann sum where the discrete points $t_j = j\partial t$

$$\sum_{j=0}^{N-1} h(t_j)(t_{j+1} - t_j) \tag{1}$$

In this equation, $h(t_j)$ can be considered as the height parameter and $(t_{j+1} - t_j)$ can be considered as the width parameter for a small section of a curve in a graph. And adding these small areas in these intervals provides the whole area of the curve. Now, in a similar way the approximation of Stochastic integral of the form $\int_0^T h(t)dW(t)$ can be written in a sum form

$$\sum_{j=0}^{N-1} h(t_j)(W(t_{j+1}) - W(t_j)) \tag{2}$$

In equation (2), N is the number of time steps/partitions over the interval [0, T]. $h(t_j)$ is a function of time t_j representing the time step at each discrete time point t_j. $W(t_{j+1}) - W(t_j)$ represents the Wiener increment between the time points t_j and t_{j+1}. This integral is commonly known as Ito integral. There is an alternative to (1) which is given by

$$\sum_{j=0}^{N-1} h((t_j + t_{j+1})/2)(t_{j+1} - t_j) \tag{3}$$

The equation (3) is also a Riemann sum approximation to $\int_0^T h(t)dt$. The corresponding alternative to Stochastic integral (4) is

$$\sum_{j=0}^{N-1} h((t_j + t_{j+1})/2)(W(t_{j+1}) - W(t_j)) \tag{4}$$

These two *"stochastic Reimann sums"* (2) and (4) give markedly different results. Further experiments shows that these differences is present even with smaller ∂t as ∂t $\to 0$. The *left hand sum* as shown in (2) give rise to what is known as *Itô* and the *"mid point sum"* as shown in (4) produces *Stratonovich* integral.

Listing 4: Stochastic Integral Calculation

```
randn('state',100)
T = 1; N = 500; dt = T/N;
```

```
3
4  dW = sqrt(dt)*randn(1,N);
5  W = cumsum(dW);
6  ito = sum([0,W(1:end-1)].*dW)
7  strat = sum((0.5*([0,W(1:end-1)]+W) + 0.5*sqrt(dt)*
       randn(1,N)).*dW)
8  itoerr = abs(ito - 0.5*(W(end)^2-T))
9
10 straterr = abs(strat - 0.5*W(end)^2)
```

In Listing #4, *Itô* integral is calculated by multiplying each elements of the W array with the dW vector followed by the equation (2) and then the sum is calculated on the elements. Similarly, *Stratonovich* integral is calculated following the equation (4). 0 is added in [0, W(1:end-1)] for both the integral calculation since W(0)=0 as mentioned in *section 2*. Furthermore, it can be proved mathematically that *Itô* version is the limiting case of

$$\int_0^T h(t)dW(t) = 1/2(W(T)^2) - (1/2)T \tag{5}$$

Simplifying, (5), we get

$$\int_0^T h(t)dW(t) = 1/2[(W(T)^2) - T] \tag{6}$$

The variable defined in Line #8 in *Listing #4* describes the amount by which ito sum differ from their respective $\partial t \to 0$ limits as mentioned in (6).
Similarly, the *Stratonovich* integral is the limiting case of

$$\int_0^T h(t)dW(t) = 1/2(W(T)^2) \tag{7}$$

And the variable defined in line #10 of *Listing #4* records the amount by which *Stratonovich* sum differs from $\partial t \to 0$ limits as mentioned in (7).
It appears that the $itoerr = 0.158$ and $straterr = 0.186$. Hence, it can be seen that both *Itô and Stratonovich integrals do not converge on the same solution.* There is no concrete bench marking possible to select any one of the above integrals since the selection task is more of a modelling issue than a mathematical issue. However, *Itô* integral is a *martingale*[3] and *Stratonovich* integral is not. Thus it gives a computational advantage over Stratonovich.

The key properties of a martingale can be summarized as below:

1. The martingale processes describes a fair game process where the Expectation $E(X) = 0$. Now suppose we have a process C with three outcomes alongside their probabilities mentioned as below:

$$10, \text{p}=1/4 \tag{8}$$
$$8, \text{p}=1/4 \tag{9}$$
$$-9, \text{p}=1/2 \tag{10}$$

Now, for this case expectation *E(C)* = 10*(1/4) + 8*(1/4) - 9*(1/2) = 0. Hence, this process can be described as a Martingale.

2. The future outcome does not depends on the past information. Suppose we have past information values $x_1, x_2, x_3, ...x_N$ stored in the Filtration Information F_n. Now a martingale states that the future outcome x_{N+1} does not depends on the information stored in F_N. Instead, the future outcome will be equal to the current outcome and in this case it is equivalent to x_N.

5 The Euler-Maruyama Method

A scalar autonomous SDE can be written after applying Euler Maruyama method in Linear SDE:

$$dX(t) = X_0 + \lambda X(t)d(t) + \mu X(t)dW(t) \tag{11}$$

In the equation (11), the first term of R.H.S is the initial Wiener value, the second term of the right hand side term i.e.

$$\lambda X(t)d(t) \tag{12}$$

is the *drift* term representing the deterministic part or the rate at which process changes deterministically and the third term in the right hand side i.e

$$\mu X(t)d(t) \tag{13}$$

is the *diffusion* term representing the stochastic part or the impact of randomness on the process. In the *diffusion* term, the stochastic integrals can be applied as discussed before i.e. *Ito* and *Stratonovich* integrals in 4.

6 Application Example

To explain SDE in context to SDCPN, we can take an example of a place with a token with three attributes as shown in the below diagram:

Place with one token

Speed	Continuous	Real
Location	Continuous	Real
X	Continuous	Non real

Token with three attributes

Figure 4: Application Example

In this example, a car movement example is considered. A particular Petri Net place contains a token which has three continuous attributes. Among these three attributes, Speed and Location of the car are real valued elements and attribute X is a non real valued element. Now to calculate Stochastic Differential Equations only real valued elements are considered. For the car example, the simulation of Speed and Location variables can be simulated using SDE. The initial values of Speed and Location variables are initialized as 0. *S[0]=0(Speed)* and *X[0]=0(Location)*. Here Speed is the continuous value and in order to use SDE it has to be initialized to zero as per the Brownian Motion conditions. Now, the equation (11) is used to calculate the Speed. *S[i] = S[i-1] + mu * dt + sigma * dW*. In this equation, *S[i-1]* is the previous Speed value of the time step, dW is the Wiener increment, dt is the time step which can be derived by dividing the total time by the number of time steps and mu and $sigma$ are the drift and diffusion coefficient. Now, in order to calculate the Location variable, speed variable can be integrated i.e. *X[i] = X[i-1] + S[i-1] * dt*. S and X variables are stored in a list variable and can be plot against time in order to visualize the simulation.

7 Conclusion

Numerical Implementation of Stochastic Integrals is widely used in various domains such as Quantitative Finance, Physics, Chemistry, Biology etc. Since in real life almost every natural processes are stochastic in nature, hence the use of Stochastic Integrals is enormous. It is imperative to say that selection of Stochastic Integral is important and it depends on the choice of the modelling task, numerical considerations and complexity of the implementation. A careful

choice of Stochastic Integral can facilitate proper analysis and good interpretation of results.

References

[1] Desmond J. Higham. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Review*, 43(3):525–546, 2001.

[2] Peter Mörters and Yuval Peres. *Brownian motion*, volume 30. Cambridge University Press, 2010.

[3] Holger von Jouanne-Diedrich. Ito, stratonovich and friends. *Available at SSRN 2956257*, 2017.