Stochastic Differential Equations

Nga Yu Lo

July 2020

1 Analytic Solutions

Note: x, W, and u are functions of t, where W(t) is the Wiener Process

1.1 $dx = \sigma dW$

Solution:

$$\int_{t_0}^{t} dx = \int_{t_0}^{t} \sigma dW$$

$$x(t) - x(t_0) = \sigma[W(t) - W(t_0)]$$

$$x(t) = x(t_0) + \sigma[W(t) - W(t_0)]$$

Ito's Lemma, Change of Variable formula

We use Ito's lemma to create a change of variable formula we can use to solve a stochastic differential equation. In Gardiner[1], if we consider an arbitaury function, y:=f(x(t)), where x is a stochastic process such that $dx = u(x,t)dt + v(x,t)dW$, then Ito's lemma lets

$$d[f(x)] = f'(x)dx + \frac{1}{2}f''(x)(dx)^2$$

Then, substituting dx, and using the rules that $(dt)^2 = 0$, $(dt)(dW) = 0$, and $(dW)^2 = dt$, we get

$$d[f(x)] = [f'(x)u(t) + \frac{1}{2}f''(x)(v(t))^2]dt + f'(x)v(t)dW \tag{1}$$

Suppose we y:=f(x,t), partial differentials are used, and Ito's formula becomes

$$d[f(x,t)] = [f_t + f_x u(x,t) + \frac{1}{2}f_{xx}(v(x,t))^2]dt + f_x v(x,t)dW \tag{2}$$

1.2 $dx = \sigma x dW$

Solution: Let $y = \log x$. Applying Ito's formula by 1, we have

$$dy = \frac{1}{x}(\sigma x dW) - \frac{-1}{2x^2}(\sigma x)^2 dt$$

$$dy = \sigma dW - \frac{\sigma^2}{2}dt$$

Integrating gives us,

$$y(t) = y(t_0) + \int_{t_0}^{t} \sigma dW - \int_{t_0}^{t} \frac{\sigma^2}{2}dt$$

[1]C.W. Gardiner. Handbook of Stochastic Methods. Springer. Third Edition. 95

$$y(t) = y(t_0) + \sigma[W(t) - W(t_0)] - \frac{\sigma^2}{2}(t - t_0)$$

Since $x = \exp(y)$, our final solution is

$$x(t) = x(t_0)\exp\left(\sigma[W(t) - W(t_0)] - \frac{\sigma^2}{2}(t - t_0)\right)$$

1.3 $dx = axdt + \sigma dW$

Solution: Let $y = x\exp(-at)$. Applying Ito's formula for f(x,t) by 2, we have

$$dy = \Big[- ax\exp(-at) + ax\exp(-at)\Big]dt + \sigma\exp(-at')dW(t')$$

$$y(t) = y(t_0) + \int_{t_0}^{t} \sigma\exp(-at')dW(t')$$

Then we substitute y with x,

$$x(t)\exp(-at) = x(t_0)\exp(-at_0) + \int_{t_0}^{t} \sigma\exp(-at')dW(t')$$

$$x(t) = x(t_0)\exp\{a(t - t_0)\} + \int_{t_0}^{t} \sigma\exp\{a(t - t')\}dW(t')$$

1.4 $dx = [a + u(t)]xdt + \sigma dW$

Solution: Assume that u(t) is integrable over an interval (b,t). We denote $\alpha(t) = at + \int_b^t u(s)ds$. Let y be a stochastic process of the form, $y = x\exp\{-\alpha(t)\}$ Applying Ito's formula by 2, we get

$$dy = \Big[- (a + u(t))x\exp\{-\alpha(t)\} + (a + u(t))x\exp\{-\alpha(t)\}\Big]dt + \sigma\exp\{-\alpha(t)\}dW$$

$$y(t) = y(t_0) + \int_{t_0}^{t} \sigma\exp\{-\alpha(t')\}dW(t')$$

Substituting y with x, we get

$$x(t)\exp\{-\alpha(t)\} = x(t_0)\exp\{-\alpha(t_0)\} + \int_{t_0}^{t} \sigma\exp\{-\alpha(t')\}dW(t')$$

$$x(t) = x(t_0)\exp\left(a(t - t_0) + \int_{t_0}^{t} u(s)ds\right) + \int_{t_0}^{t} \sigma\exp\left(a(t - t') + \int_{t'}^{t} u(s)ds\right)dW(t')$$

1.5 $dx = [a + u(t)]xdt + \sigma xdW$

Solution: Let $y = \log x$. Applying Ito's formula by 1, we have

$$dy = \frac{1}{x}(a + u(t))xdt - \frac{1}{2x^2}\sigma^2x^2dt + \frac{1}{x}\sigma xdW$$

$$dy = (a + u(t))dt - \frac{1}{2}\sigma^2dt + \sigma dW$$

$$y(t) = y(t_0) + \int_{t_0}^{t}(a + u(t) - \frac{\sigma^2}{2})dt + \int_{t_0}^{t} \sigma dW$$

$$y(t) = y(t_0) + \int_{t_0}^{t}(a + u(t) - \frac{\sigma^2}{2})dt + \sigma[W(t) - W(t_0)]$$

Since $x = \exp(y)$, our final solution is

$$x(t) = x(t_0)\exp\left(\int_{t_0}^{t}(a + u(t) - \frac{\sigma^2}{2})dt + \sigma[W(t) - W(t_0)]\right)$$

2 Numerical Solutions

2.1 The Wiener Process

In our stochastic differentials, diffusion is expressed by a Wiener Process. The one-dimensional Wiener Process is also known as Brownian Motion. Wiener process is also a Markov process and has independence of increments, which means that the change in W (i.e. $W(t+dt) - W(t)$) over each interval of time, is independent from another. In summary, the Wiener Process has the following properties:

1. $W_0 = 0$
2. W has independent increments, i.e. for any $t > 0$, the results of $W(t+dt) - W(t)$ are independent of past values $W(s)$ where $s < t$.
3. The increments of W follows a Gaussian distribution with mean 0 and a variance dt, i.e. $W(t+dt) - W(t) \sim N(0, \sqrt{dt})$
4. W is continuous

2.1.1 Simulating the Wiener Process

For our numerical solutions, we simulate the Wiener process by the following algorithm,

1. For an interval of time T, we choose a step size dt, and the number of time step is $n = T/dt$.
2. We generate n random variable from $N(0, \sqrt{dt})$. This is because $W(t+dt) - W(t)$ has a Gaussian distribution, $N(0, \sqrt{dt})$. We denote each random variable by $\Delta W_i = W_{t_{i+1}} - W_{t_i}$ where $i \in \{0, 1, 2, ...n\}$
3. For each t_{i+1}, we find a value for W_{i+1} by taking the nth random variable and adding it to W_i. Thus, we generate values for W at each t_i

2.2 Euler-Maruyama Method

For any stochastic differential $dx = F(x,t)dt + G(x,t)dW$, Euler-Maruyama uses the following algorithm.

$$x_{t_{i+1}} = x_{t_i} + F(x_{t_i}, t_i)dt + G(x_{t_i}, t_i)\Delta W_i \tag{3}$$

where $dt = t_{i+1} - t_i$ and $\Delta W_i = W_{t_{i+1}} - W_{t_i}$

2.2.1 Example 1: $dx = adt + bdW$

For $dx = adt + bdW$ where a and b are constants, the exact solution is

$$x(t) = x(t_0) + at - at_0 + b[W(t) - W(t_0)]$$

Suppose $x(t_0) = 0$, $a = b = 0.5$ and $W(t_0) = 0$, Euler-Maruyama returns the solution shown in Figure 1. Compared to the exact solution shown in Figure 2, the Euler-Maruyama method depicts the same behavior as the exact solution. Note: for this happen, the same Wiener process must be used. The python code for this example is shown in Figure 3.

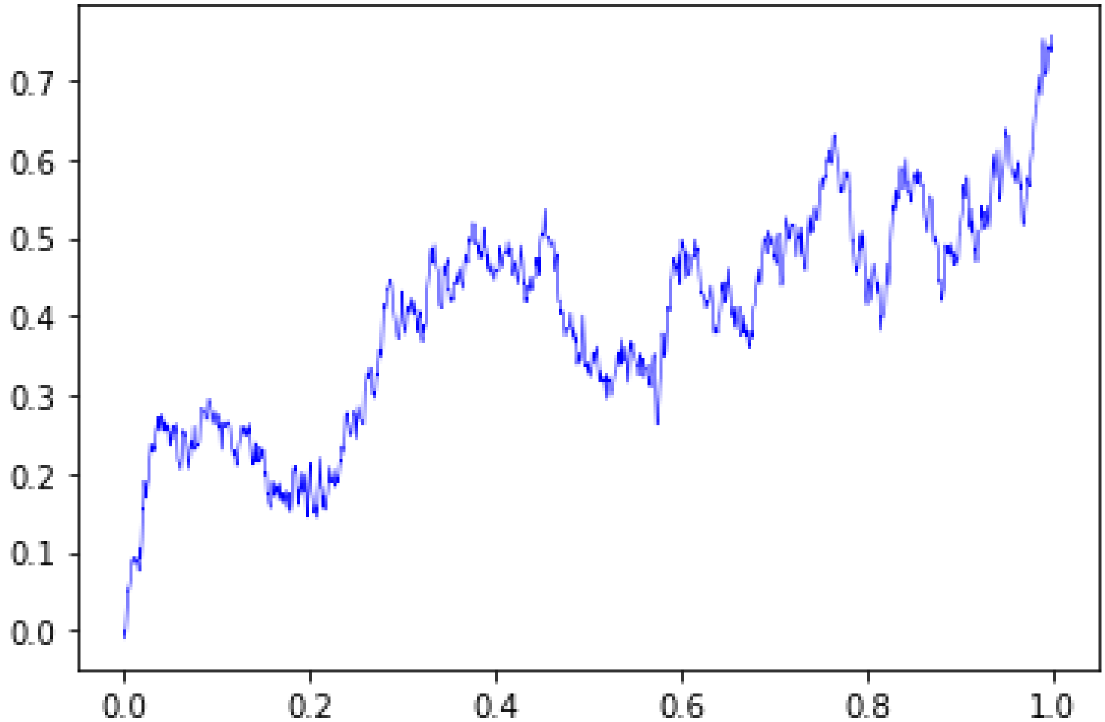

Figure 1: The numerical solution of the Stochastic differential

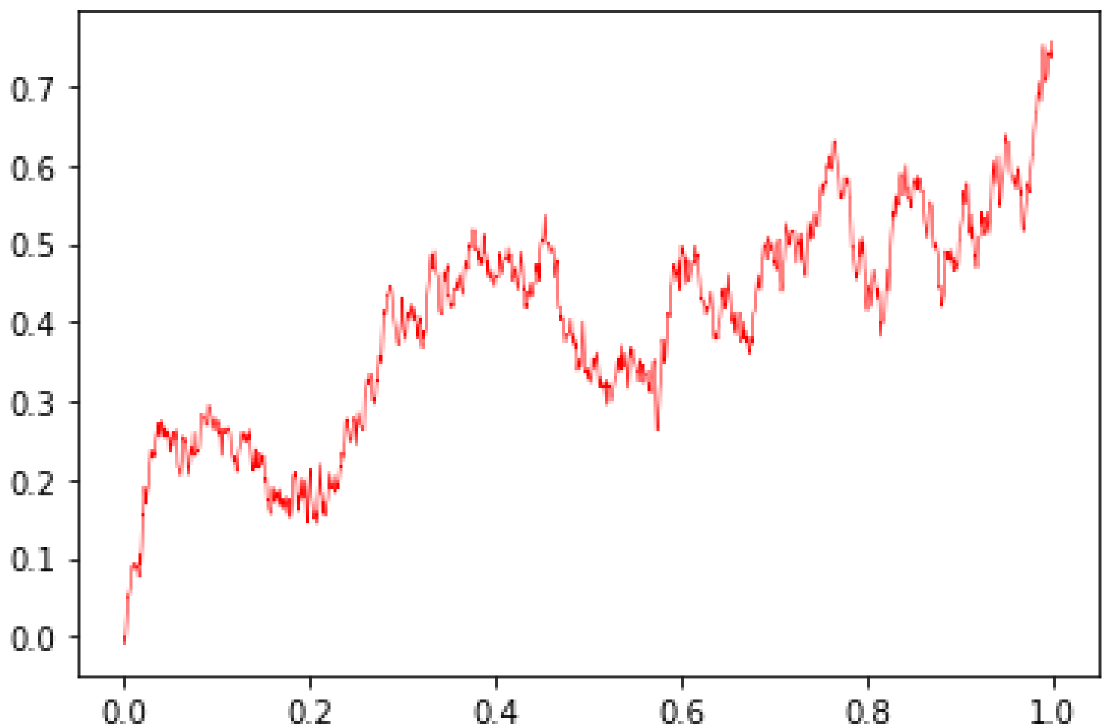

Figure 2: Exact solution of the Stochastic differential

```
import numpy as np
import matplotlib.pyplot as plt

#this solves the stochastic differential equation dx=adt+bdW,

dt = .001  # Time step.
T = 1.  # Total time.
n = int(T / dt)  # Number of time steps.
t = np.linspace(0, T, n)  # Vector of times.

def wiener_process(dt,T):
    W0 = [0]
    n=int(T/dt)
    # simulate the increments by normal random variable generator
    increments = np.random.normal(0, 1*np.sqrt(dt),n)
    W = W0 + list(np.cumsum(increments))
    return W

W=wiener_process(dt,T)

def numerical(x_0,a,b,t,dt,W):
    x= np.zeros(t.shape[0])
    x[0]=x_0
    #implementing euler-maruyama method
    for i in range (0,t.shape[0]-1,1):
        x[i+1]=x[i]+a*dt+b*(W[i+1]-W[i])
    return x

x=numerical(0.1,0.5,0.5,t,dt,W)
```

Figure 3: Python code for the numerical solution of Example 1

2.2.2 Example 2: $dx = axdt + bxdW$

For $dx = axdt + bxdW$ where a and b are constants, the exact solution is

$$x(t) = x(t_0) \exp\left(\int_{t_0}^{t} (a - \frac{b^2}{2})dt + b[W(t) - W(t_0)] \right)$$

Suppose $x(t_0) = 1$, $a = b = 0.5$ and $W(t_0) = 0$, the Euler-Maruyama solution and the exact solution are shown in Figure 4 and 5. The behaviors of the two solutions comparably resemble each other.

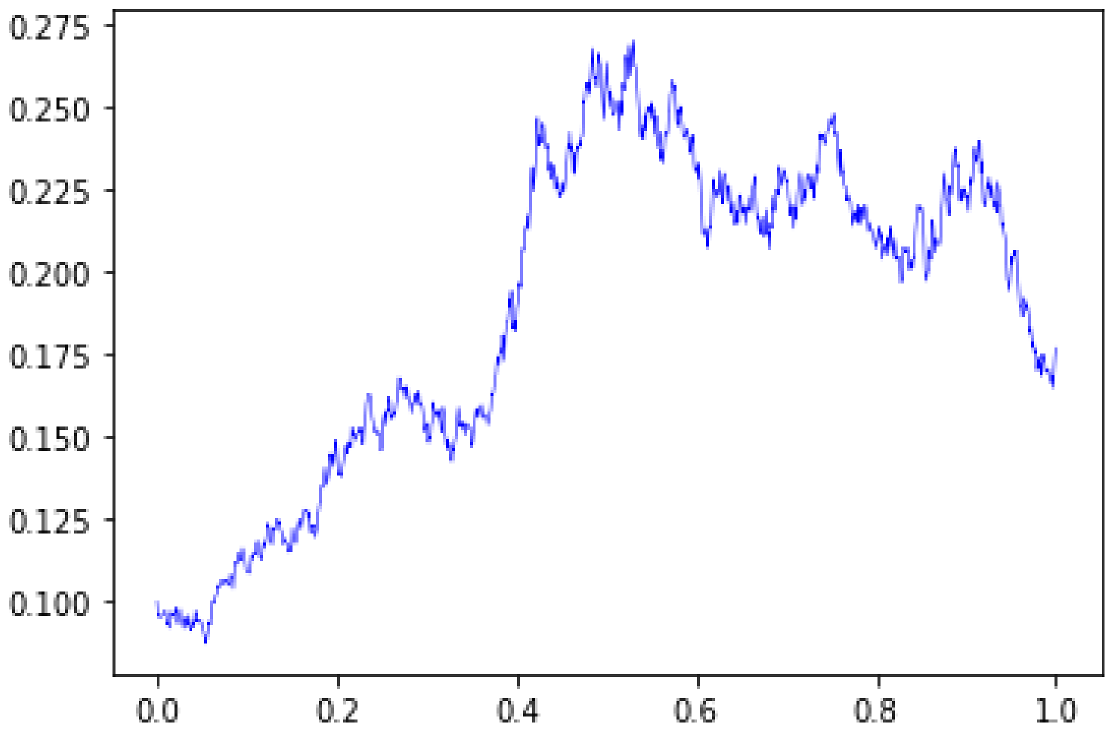

Figure 4: Numerical Solution for Example 2

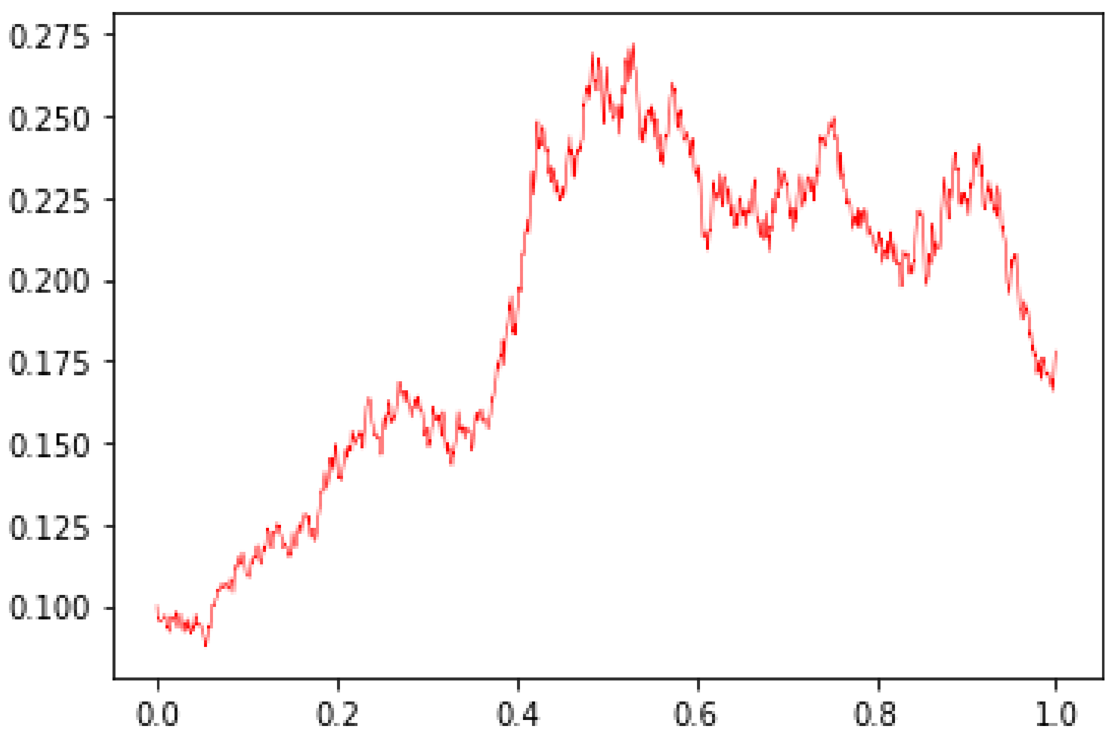

Figure 5: Exact Solutions for Example 2

2.2.3 Example 3: $dx = [a + u(t)]xdt + bxdW$

In our SDE, u(t) is some known function. Let's suppose u(t) is the sigmoid function,

$$u(t) = \frac{1}{1 + \exp(-20(t - 0.5))}$$

. The exact solution is

$$x(t) = x(t_0) \exp\left(\int_{t_0}^{t} (a + u(t) - \frac{b^2}{2})dt + b[W(t) - W(t_0)] \right)$$

Suppose $x(t_0) = 1$, $a = b = 0.5$ and $W(t_0) = 0$, the Euler-Maruyama solution and the exact solution is shown in Figure 6.

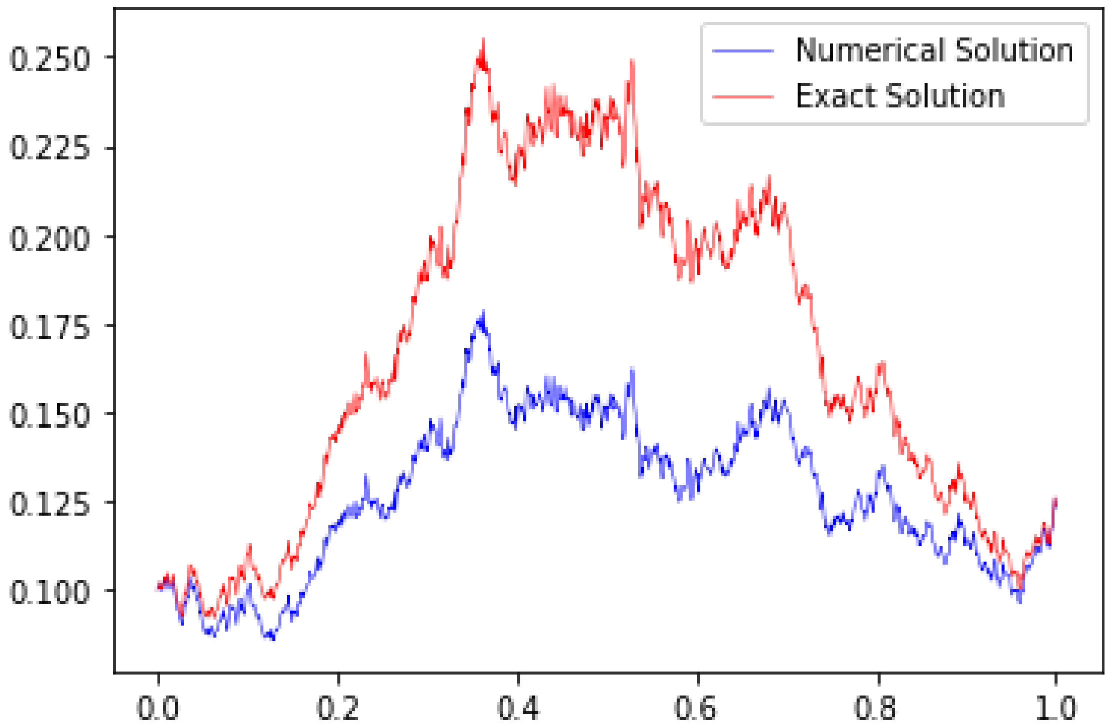

Figure 6: Solutions for Example 3

2.2.4 Example 4: $dx = [a + u(t)]xdt + bdW$

In our exact solution for this stochastic differential, we assume that $u(t)$ is integrable. For our numerical solution, let's suppose u(t) is the sigmoid function,

$$u(t) = \frac{1}{1 + \exp(-20(t - 0.5))}$$

Suppose $x(t_0) = 1$, $a = b = 0.5$ and $W(t_0) = 0$, the Euler-Maruyama solution is shown in Figure 7. The code for stimulating the numerical solution is shown in Figure 8.

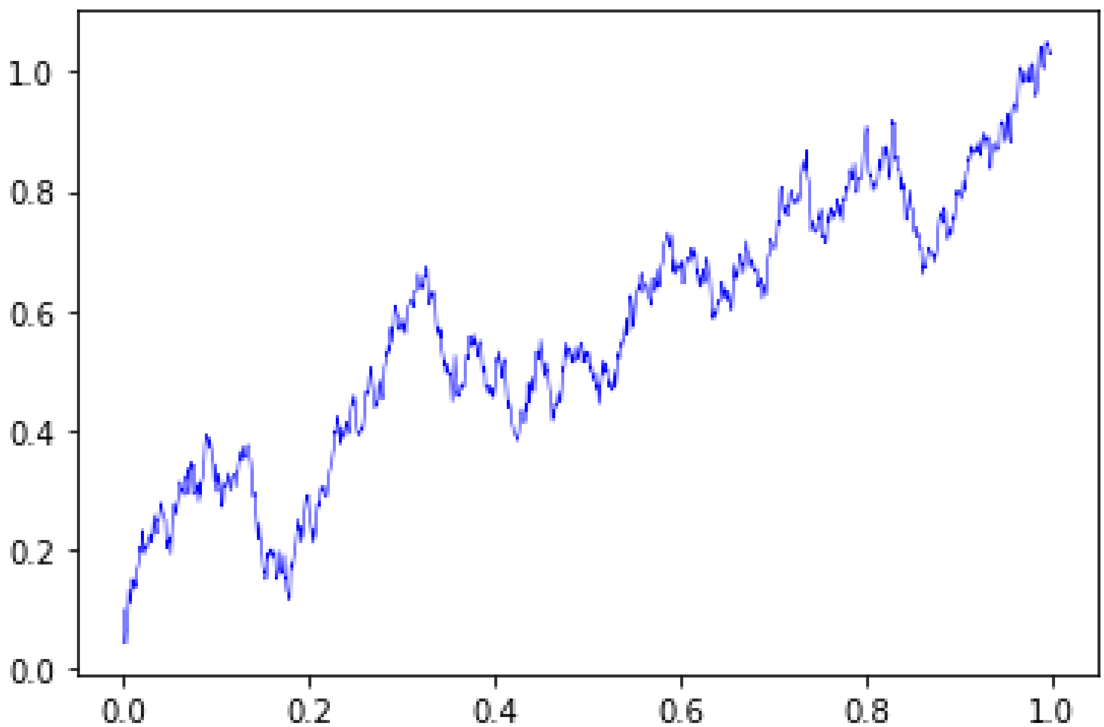

Figure 7: Numerical Solution for Example 4

```
import matplotlib.pyplot as plt
import numpy as np

#dx=(a+u(t))xdt+ bdW

dt = .001  # Time step.
T = 1.  # Total time.
n = int(T / dt)  # Number of time steps.
t = np.linspace(0, T, n)  # Vector of times.

def sigmoid(t):
    return 1/(1+np.exp(-20*(t-0.5)))

def wiener_process(dt,T):
    W0 = [0]
    n=int(T/dt)
    # simulate the increments by normal random variable generator
    increments = np.random.normal(0, 1*np.sqrt(dt),n)
    W = W0 + list(np.cumsum(increments))
    return W

W=wiener_process(dt,T)

def numerical(x_0, a, b,t,dt,W):
    x= np.zeros(t.shape[0])
    x[0]=x_0
    #implementing euler-maruyama method
    for i in range (0,t.shape[0]-1,1):
        x[i+1]=x[i]+[(a+sigmoid(t[i]))*x[i]*dt]+[b*(W[i+1]-W[i])]
    return x

x=numerical(0.1,0.5,0.5,t,dt,W)
```

Figure 8: Python Code for the numerical solution of Example 4

We can also solve stochastic differential equations on matlab using the Fiance Toolbox. Figure 9 shows the matlab code for solving example 4. Matlab allows user to create a SDE object by defining the drift and diffusion rates (i.e. F and G) and the Wiener Process is inherently defined. simByEuler then solves the differential by the method of Euler-Maruyama.

```
%dx=F(t,x)dt+G(t,X)dW
%dx=(a+u(t))xdt+bdW
F=@(t,X) (0.5+(1/(1+exp(-20*(t-0.5)))))*X;
G=@(t,X) 0.5;

X_sde=sde(F,G);
[X1,T]=simByEuler(X_sde, 100,'Deltatime',0.01);
figure; plot(T,X1)
```

Figure 9: Matlab Code for solving Example 4