PH3205-Computational Physics

Spring 2022

Bipradeep Saha (19MS135)

Indian Institute of Science Education and Research, Kolkata,
Mohanpur, West Bengal, 741246, India.

February 19, 2022

Worksheet 7

A notebook is provided containing solution to both the problems: *WS7_notebook.ipynb*. Also individual *.py* files are provided separately for this.

Problem 1: Black-Scholes Equation

We will implement Euler-Maruyama method and solve the Black-Scholes equation. The Black-Scholes equation is the following:

$$dS_t = S_t \mu dt + \sigma S_t dW_t$$

for the noise term, we are given, $\Delta W_{t_i} = z_i\sqrt{dt}$, where z_i is a random variable with zero mean and unit variance, i.e $z_i \in \mathcal{N}(0,1)$. Thus the approximated solution will be:

$$S_{t_i} = S_{t_{i-1}} + S_{t_{i-1}}\mu dt + \sigma S_{t_{i-1}} dW_{t_i}$$

Two functions are defined for this part, $BS(S0, T, N, mu, sigma)$ and a complementary $w(N, dt)$. The former one solves the BS equation while the later helps in solving.
The required *.py* file is *WS7_1.py* and the plots are *Problem1_a.jpg* and *Problem1_b.jpg*.
Explaining the plots:

- The plot for part a of the problem is *Problem1_a.jpg*, it shows the approximated and analytical solution for one realization.
- The plot for part b of the problem is *Problem1_b.jpg*. For this one we computed RMS error for 100 realization for a given dt or N and then computed the mean of RMS and plotted. One can clearly see that with incresing dt the RMS error increases.

Note: Since Black-Scholes equation is used to model evolution of financial instruments, the Doc-string is tailored to that need only.

Problem 2: Birth Death equation

The birth-death process is given by the following equation:

$$dn = k_1 dt - k_2 dt + dW(t)$$

where the stochastic process is given by,

$$\Delta W_i(t) = z_i\sqrt{dt(k_1 + k_2 n_{i-1})}$$

Here also similar function are defined, but for computing the analytical solution a separate function is defined to save computational time as it doesn't depend on the initial value so doesn't needs to be computed for each iterations.

The required *.py* file is *WS7_2.py* and the plots are *Problem2_a.jpg* and *Problem2_b.jpg*. Explaining the plots:

- The plot for part a of the problem is *Problem2_a.jpg*, it shows the average of approximated solution and analytical solution for 10000 realizations. In a first look, it may look like there is no error, but the bottom plot shows the relative error between the analytical and the average solution. We can see that the error becomes stable as time evolves.
- The plot for part b of the problem is *Problem2_b.jpg*. We plot the distribution of end $N(t)$ when the solution is stable and plot fit it to a Poisson distribution. Since number of Realization is large, don't be alarmed if the distribution looks similar to a Gaussian distribution as for large no. of Poisson random variable, it takes the shape of a bell curve (Proved in MA2202)

NOTE: Make sure you have Tqdm python library installed. It is used to track the progress of code.