Numerical and analytical solution of 1D diffusion equation

Abstract

I used a C program to solve 1D unsteady diffusion equation with periodic boundary conditions. The objective is to use a given numerical scheme and test its stability and accuracy. For testing accuracy, I used an analytical solution and compared the time evolution obtained from the numerical solution. We observed that with decreasing time step the accuracy improved. I could recover the behavior of the diffusion equation from the numerical program.

Introduction

The process of diffusion is described mathematically by a second order partial differential equation. The two variables on which the PDE is dependent are space and time. The process of diffusion follows the principle of Brownian motion. Brownian motion describes the random motion of the particles. However, here as per the objective of this assignment, we use a PDE describing spatiotemporal evolution of ‘u’ to understand the diffusion process.

$$\frac{\partial u}{\partial t} = K\frac{\partial^2 u}{\partial x^2} \qquad 0 \le x \le 1$$

where K is the diffusion coefficient, which is defined as the ratio of Thermal conductivity to the product of specific heat and density.

$$\frac{K_0}{c\rho}$$

To solve PDE of first order in time and second order in space, it requires an initial condition and conditions at both boundaries of the 1D domain.

$$InitialCondition : u(\ x, 0) \ = \sin(\ 2\pi x)$$
$$BoundaryConditions : u(\ 0, t) \ = u(\ 1, t)$$

Using the above conditions, I explore the nature of the diffusion equation.

Analytical solution of Diffusion Equation:

To find the solution of the diffusion equation, I use the separation of variables method.

$$u(x,t) = X(x)\,T(t)$$

Taking the required derivatives

$$u_{xx} = X''(x)\,T(t)$$

$$u_t = X(x)\,T'(t)$$

Using the PDE $u_t = u_{xx}$

$$X(x)\,T'(t) = X''(x)\,T(t)$$

$$\frac{T'(t)}{T(t)} = \frac{X''(x)}{X(x)} = -\lambda, \qquad \lambda = constant$$

Using the BC $u(0,t) = u(1,t)$

$$X(0)\,T(t) = X(1)\,T(t)$$

Solving for X(s)

$$\frac{X''(x)}{X(x)} = -\lambda$$

$$X''(x) + \lambda X(x) = 0$$

With Boundary condition,

$$X(0) = X(1)$$

To choose an appropriate value of λ

Case I: $\lambda < 0$

Let $\lambda = -k^2 < 0$,

the solution is

$$\begin{aligned}
&X = Ae^{(\ kx)}\ + Be^{(\ -kx)}\\
&X(\ 0)\ = Ae^{k(\ 0)}\ + Be^{-k(\ 0)}\\
&X(\ 0)\ = A + B\\
&X(\ 1)\ = Ae^{(\ k)}\ + Be^{(\ -k)}
\end{aligned}$$

Using BC $X(\ 0)\ = X(\ 1)$

$$\begin{aligned}
&A + B = Ae^{k} + Be^{-k}\\
&A(\ 1 - e^{k})\ = B(\ e^{-k} - 1)\\
&A = e^{-k}B
\end{aligned}$$

Since |k| >0, we have A=B=u=0 which is the trivial solution.

Case II: $\lambda = 0$

$$\begin{aligned}
&X''(\ x)\ = 0\\
&X(\ x)\ = A(\ x)\ + B\\
&X(\ 0)\ = B\\
&X(\ 1)\ = A + B
\end{aligned}$$

Using BC $X(\ 0)\ = X(\ 1)$

$B = A + B$

Thus $A = B = u = 0$ which gives a trivial solution.

Case III: $\lambda > 0$

For periodic condition

$$\begin{aligned}
&X(\ x)\ = Acos(\ \sqrt{\lambda x})\ + Bsin(\ \sqrt{\lambda x})\\
&X(\ 0)\ = A\\
&X(\ 1)\ = Acos(\ \sqrt{\lambda})\ + Bsin(\ \sqrt{\lambda})\\
&Acos(\ \sqrt{\lambda})\ + Bsin(\ \sqrt{\lambda})\ = A\\
&A(\ cos(\ \sqrt{\lambda})\ - 1)\ + Bsin(\ \sqrt{\lambda})\ = 0\\
&\frac{A}{B} = \frac{-sin(\sqrt{\lambda})}{cos(\sqrt{\lambda}) - 1}
\end{aligned}$$

For $\lambda = n^2\pi^2$ satisfies the above equation.

The values of λ are called eigen values.

Solving for $T(\ t)$

$$\frac{T^{'}(\ t)}{T(\ t)} = -\lambda$$
$$T^{'}(\ t)\ = -\lambda T(\ t)$$
$$= -n^2\pi^2 T(\ t)$$

The above equation is in the form $\frac{dT}{dt} = KT$ which results in an exponential solution, and hence

$$T(\ t)\ = e^{-n^2\pi^2 t}$$

Complete solution:

$$u(\ x,t)\ = X(\ x)\ T(\ t)$$
$$X_n(\ x)\ = a_n cos(\ n\pi x)\ + b_n sin(\ n\pi x)$$
$$T_n(\ t)\ = c_n e^{-n^2\pi^2 t}$$
$$u_n(\ x,t)\ = X_n(\ x)\ T_n(\ t)$$
$$= (\ a_n cos(\ n\pi x)\ + b_n sin(\ n\pi x)\ )\ c_n e^{-n^2\pi^2 t}$$
$$u_n(\ x,t)\ = (\ A_n cos(\ n\pi x)\ + B_n sin(\ n\pi x)\ )\ e^{-n^2\pi^2 t}$$

Where $A_n = a_n c_n \quad B_n = b_n c_n$

Under initial conditions at t=0

$$u_n(\ x,0)\ = (\ A_n cos(\ n\pi x) + B_n sin(\ n\pi x)\ )$$

as $u_n(\ x,0)$ is solution to pde,then as per principle of superposition any finite sum is also solution to pde.

$$u(\ x,t)\ = \sum_{n=1}^{\infty} u_n(\ x,t)$$

Let

$$f(\ x)\ = u(\ x,0)\ = \sum_{n=1}^{\infty} u_n(\ x,0)\ = \sum_{n=1}^{\infty}(\ A_n cos(\ n\pi x)\ + B_n sin(\ n\pi x)\ )$$

Multiply $\int_0^1 sin(\ m\pi x)\ dx$ on both sides

$$\int_0^1 sin(\ m\pi x)\ f(\ x)\ dx = \sum_{n=1}^{\infty} A_n \int_0^1 sin(\ m\pi x)\ cos(\ n\pi x)\ dx + \sum_{n=1}^{\infty} B_n \int_0^1 sin(m\pi x)\ sin(\ n\pi x)\ dx \qquad (A)$$

Using the orthogonality property,

$$\int_0^1 sin(\ m\pi x)\ sin(\ n\pi x)\ dx = \quad 0 \quad m \neq n$$

$$= \quad \frac{1}{2} \quad m = n \quad = \frac{1}{2}\delta_{mn}$$

$$\begin{aligned} \delta_{mn} &= 0 \quad m \neq n \\ &= 1 \quad m = n \end{aligned}$$

$$\frac{1}{2} * 2 \int_0^1 sin(\ m\pi x)\ sin(\ n\pi x)\ dx$$

$$\frac{1}{2} \int_0^1 [cos(\ (\ m-n)\ \pi x)\ - cos(\ (\ m+n)\ \pi x)]\ dx$$

$$\frac{1}{2} \int_0^1 cos(\ m\pi x)\ dx = \frac{1}{2}\delta_{m0}$$

Taking equation (A) and considering the values of $m = n = 2$

$$\int_0^1 sin(2\pi x)\ f(\ x)\ dx = A_2 \int_0^1 sin(\ 2\pi x)\ cos(\ 2\pi x)\ dx + B_2 \int_0^1 sin(\ 2\pi x)\ sin(\ 2\pi x)\ dx$$

$$= \frac{A_2}{2} \int_0^1 sin2(\ 2\pi x)\ dx + B_2 \frac{1}{2}\delta_{22}$$

$$= \frac{A_2}{2} \left[-\frac{cos(\ 4\pi x)}{4\pi} \right]_0^1 + \frac{B_2}{2}$$

$$\int_0^1 sin(2\pi x)\ f(\ x)\ dx = \frac{B_2}{2}$$

Rearranging the terms $$B_2 = 2\int_0^1 sin(2\pi x)\ f(\ x)\ dx$$

Thus

$$u(x,t) = B_2 sin(2\pi x) e^{-4\pi^2 t}$$

Numerical SImulation of Diffusion Equation

A code is developed in C to study the nature of 1D Diffusion equation using explicit Euler (time derivative) and second order central difference scheme.

All the source files are available at : https://github.com/shilpa12vinu/FLAME_app

To run the code:

```
$gcc diffusion_final1.c -o diffusion_final1 -lm
$ ./diffusion_final1
$./plot_1.sh (This script requires gnuplot to be installed. All the figures will be generated by this script)
```

Nomenclature

1. n : grid size
2. L : Length of the domain
3. Constant : value of diffusion coefficient
4. u[ ] : diffusion parameter
5. dudt[ ] : change in diffusion
6. u_a[ ] : analytical value of diffusion
7. err[ ] : error encountered at each node
8. abs_err[ ], avg_err[ ] : absolute and average value of error
9. T_final : Total time of simulation
10. dt : incremental step in time 0.00001
11. Tot_step = (t_final/dt) = total calculated time
12. dx : size of subdomain
13. ic : initial condition
14. ex : exponential term
15. test

For the ease of understanding the code, I will introduce the following, which are the main components of the numerical code.

Section 1: Initializing 0 to all nodal points

Traversing through each node (for a grid size (n) 128), zero is assigned to each node.

Section 2: Initial Condition

Assigning Initial condition $u(x,0) = sin(2\pi x)$ to all nodal points and these values are written to a file named “initial”.

Section 3. Boundary Condition

Diffusion is studied at time steps of 0,100, 500 and 1000.

Solutions at different time steps are computed for the two cases as shown below

Case 1: $u[0] = u[n] = 0$
Case 2: $u[0] = u[n]$

Traversing through each time step (**for** loop of “j”) the change in value of diffusion is updated at all nodal points.

At zeroth node, nodal value updation is carried out in the following manner.

$$dudt[k] = constant * ((u[1] - u[0])/power)$$
$$power = dx^2$$

At the end point, nodal value updation is carried out in the following manner.

$$dudt[k] = constant * ((u[n-1] - u[n])/power)$$
$$power = dx^2$$

Updation at remaining points is done in the following manner

$$dudt[k] = constant * ((u[k+1] - 2 * u[k] + u[k-1])/power)$$
$$power = dx^2$$

The value of diffusion in each node is updated in the following manner.

$$u[k] = u[k] + (dudt[k] * dt)$$

Section 4: Computing analytical value of diffusion, along with the average and absolute errors

$$u(x,t) = sin(2\pi x)e^{-2^2\pi^2 t}$$

Error is calculated as the difference between the analytical and computed value of diffusion at each node.

$$AverageError = \frac{\sum_1^n Error}{n}$$
$$AbsoluteError = \frac{\sum_1^n |Error|}{n}$$

Results

Using periodic boundary condition the diffusion problem is solved in the following two cases

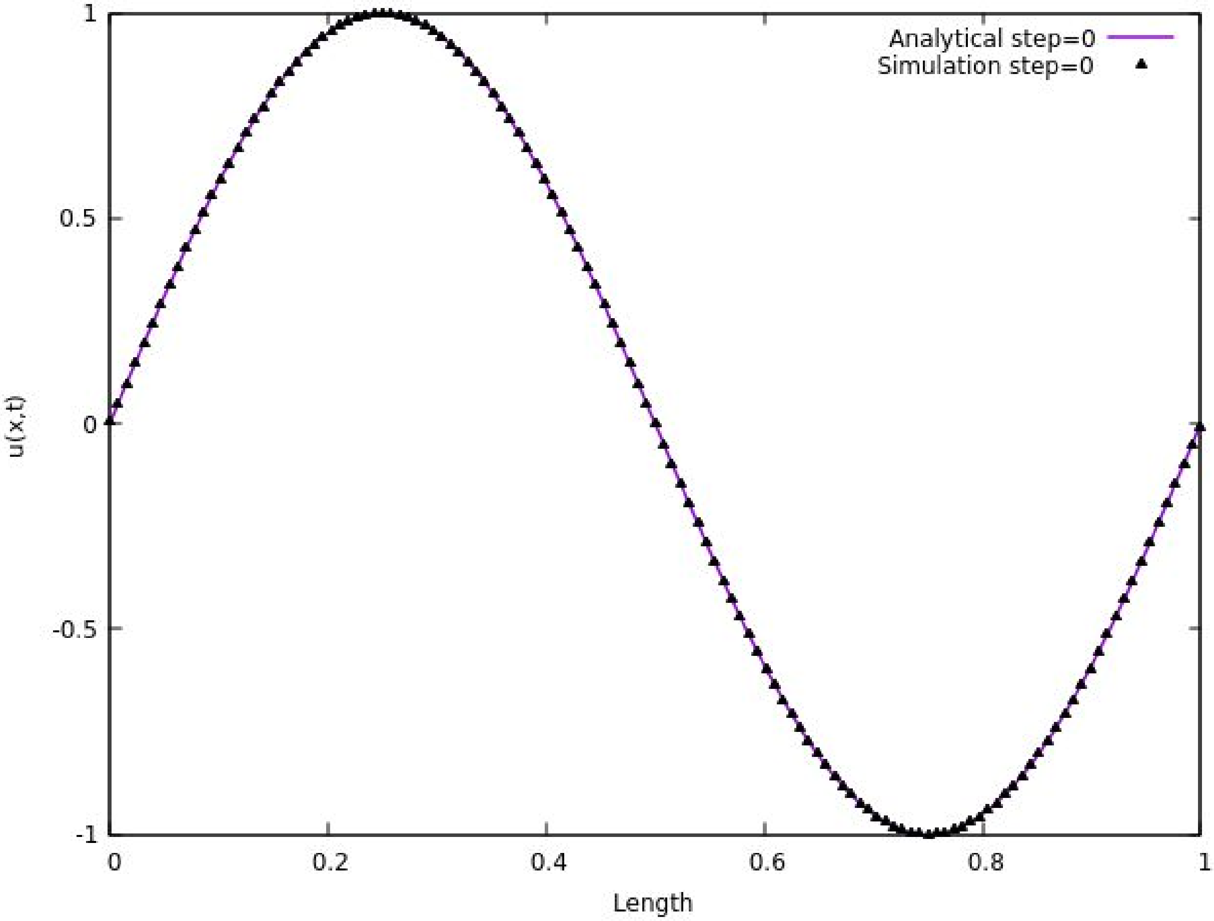
Case 1: $u[0] = u[n] = 0$

Case 2: $u[0] = u[n]$

Case 1:
Periodic boundary $u[0] = u[n] = 0$
File name: diffusion_final1.c
Script file: plot_1.sh (This script requires gnuplot to be installed. All the figures will be generated by this script)

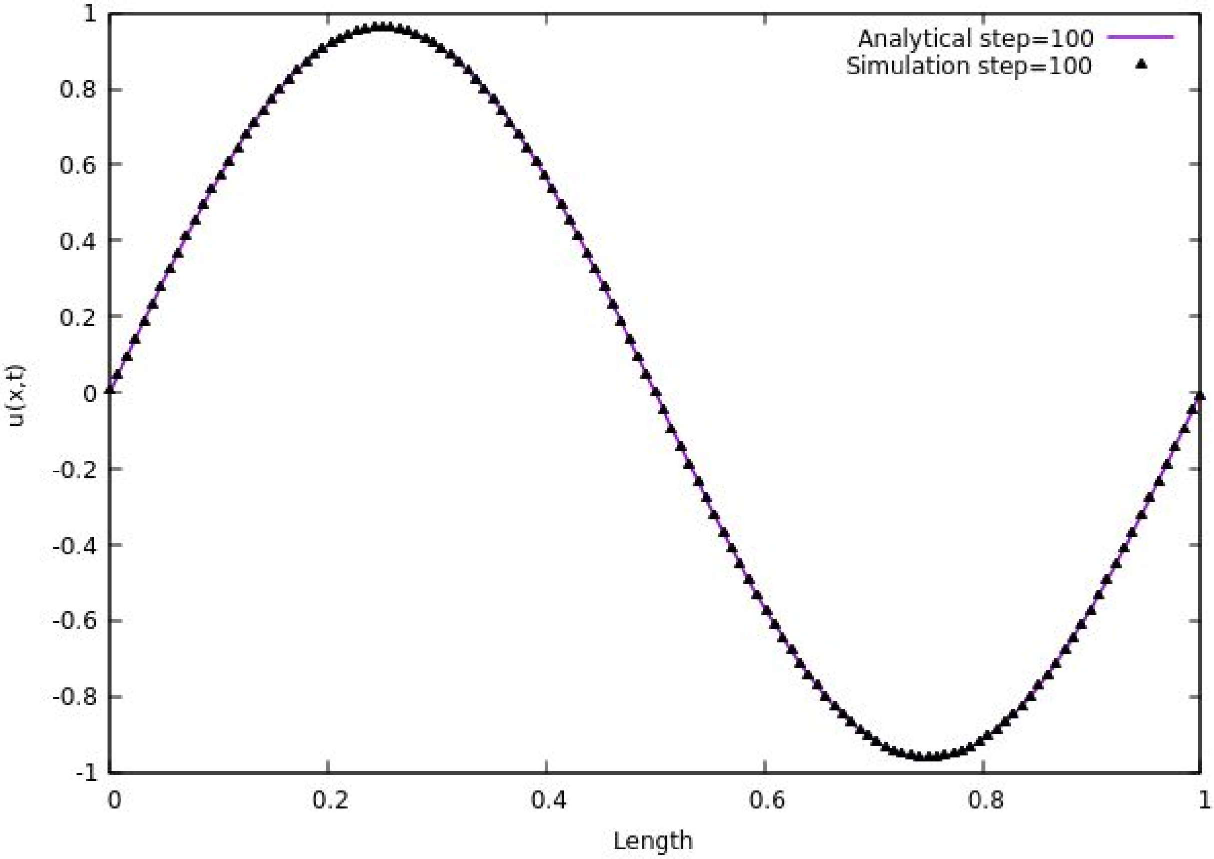
Sl No	Time Step	Average Error	Absolute Error
1	0	0.0000	0.000377
2	100	0.0000	0.000739
3	500	0.0000	0.001048
4	1000	0.0000	0.001301

1) Zeroth time step



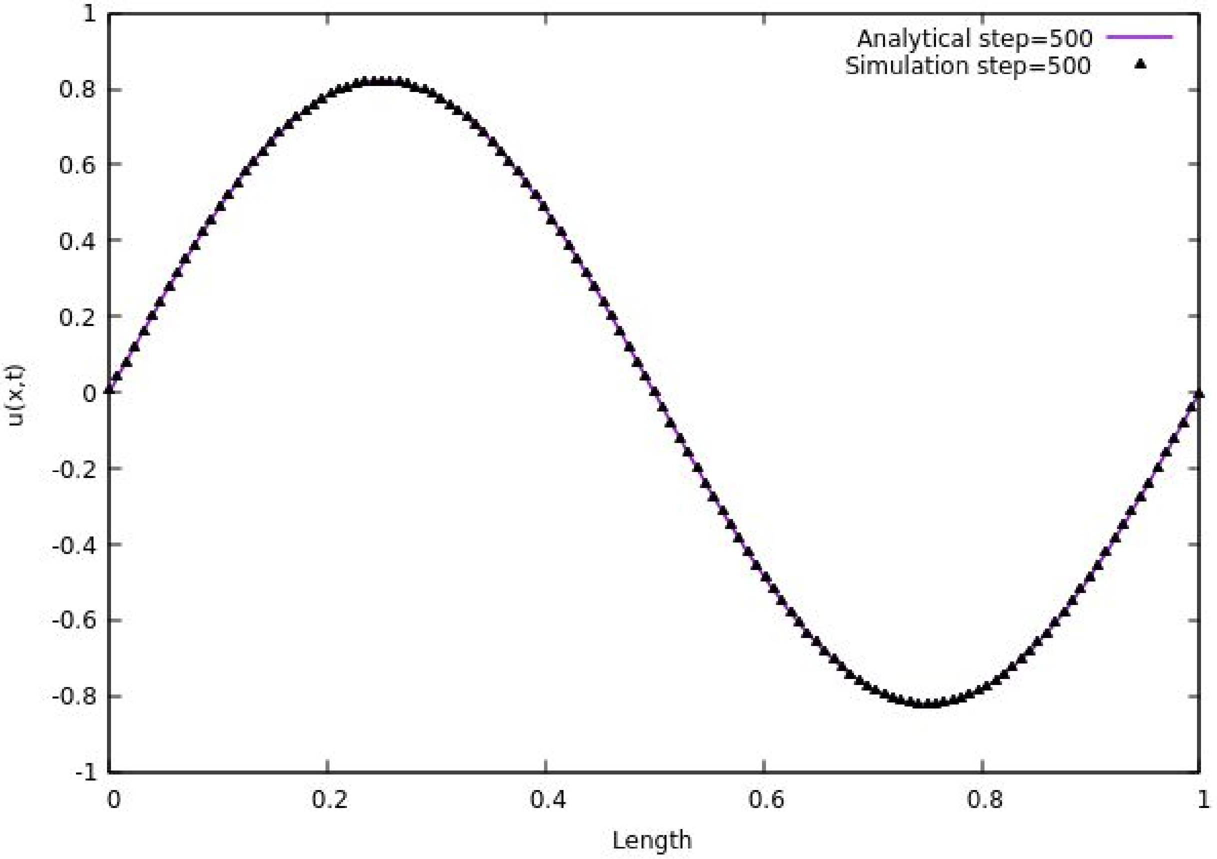
Average error= 0.0000
Absolut error = 0.000377

2) Time step of 100th sec



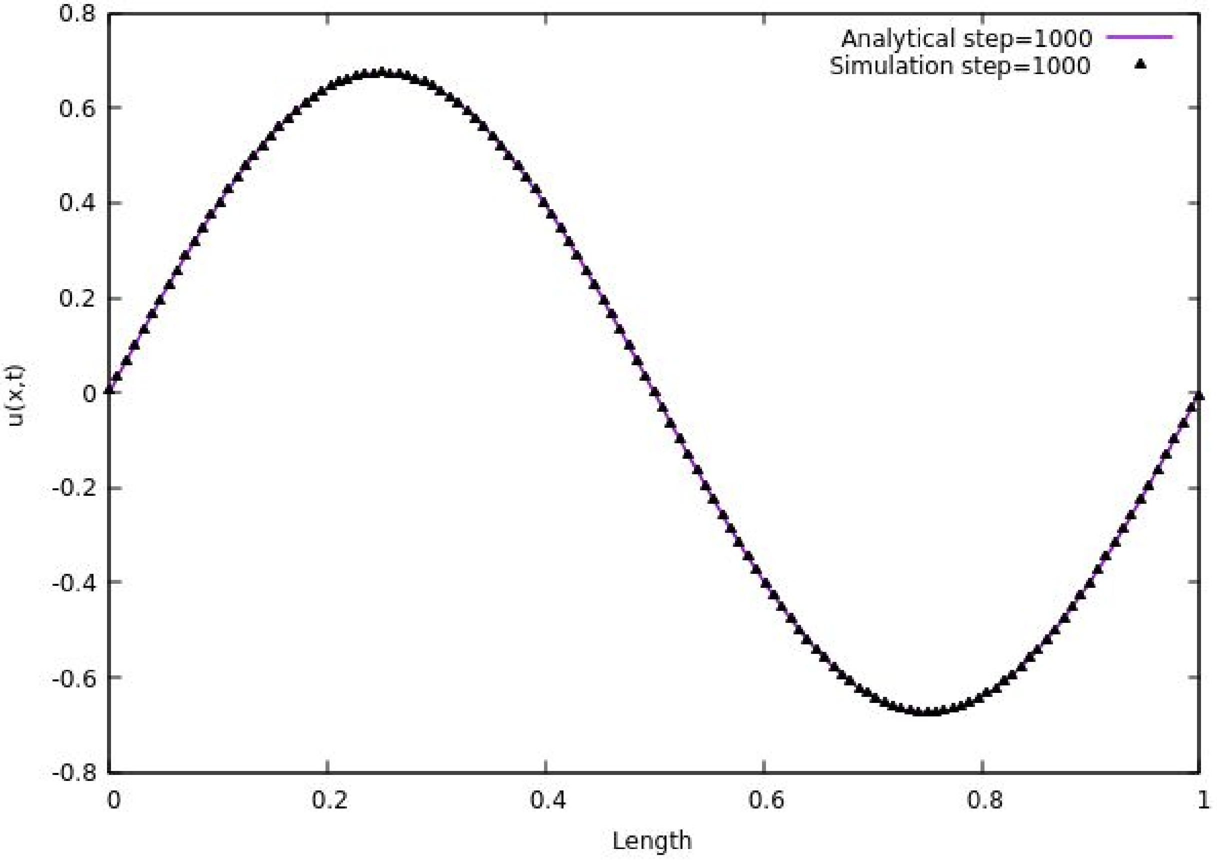
Average Error = 0.0000
Absolute Error=0.000739

3)Time step of 500th sec



Average error=0.0000
Absolute error=0.001048

4) Time step of 1000th sec



Average error=0.0000
Absolute error= 0.001301

Case II:

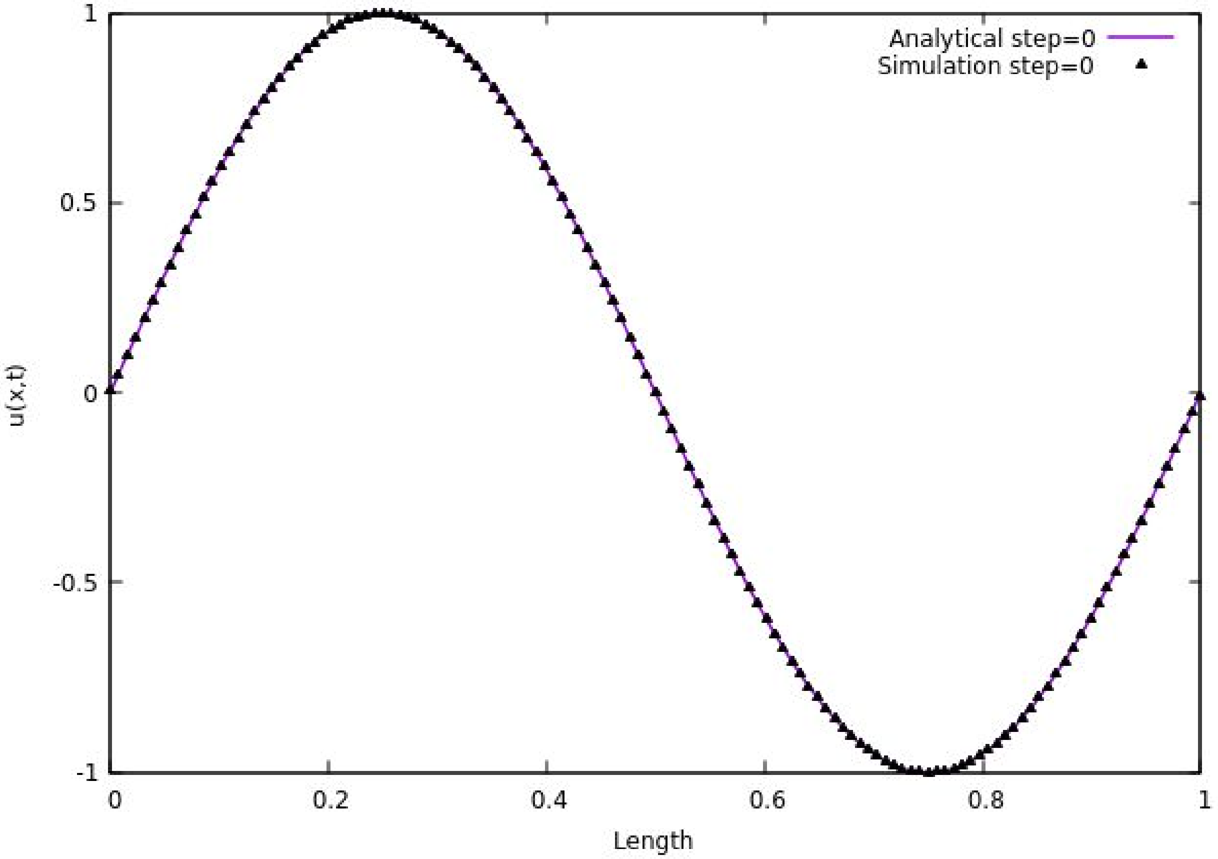
Periodic Boundary condition $u[0] = u[n]$
File name: diffusion_final2.c
Script file:plot_2.sh

To run the code in terminal

```
$ gcc  diffusion_final2.c  -o  diffusion_final2  -lm
$ ./diffusion_final2
$ ./plot_2.sh
```

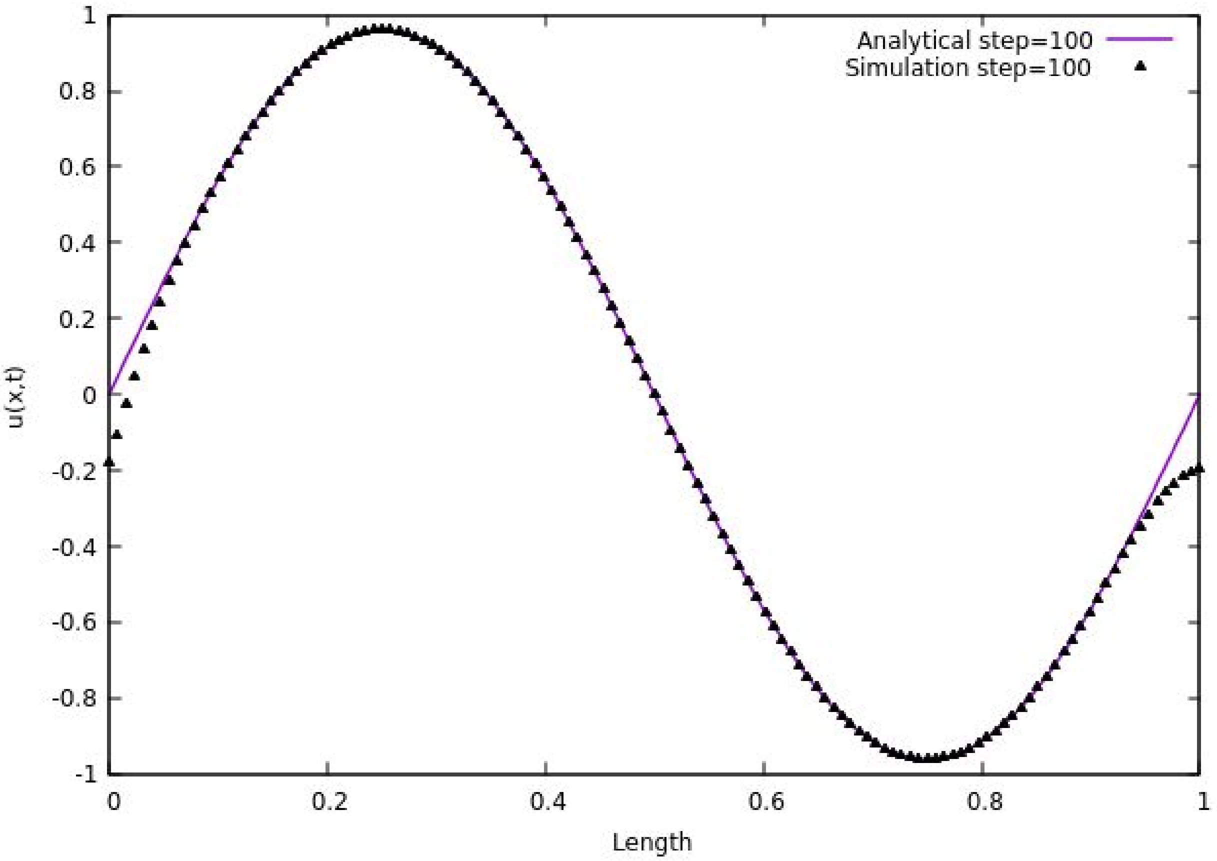
Sl No	Time Step	Average Error	Absolute Error
1	0	0.0000	0.000377
2	100	0.012319	0.012894
3	500	0.069327	0.069952
4	1000	0.173130	0.173755

1) Zeroth time step



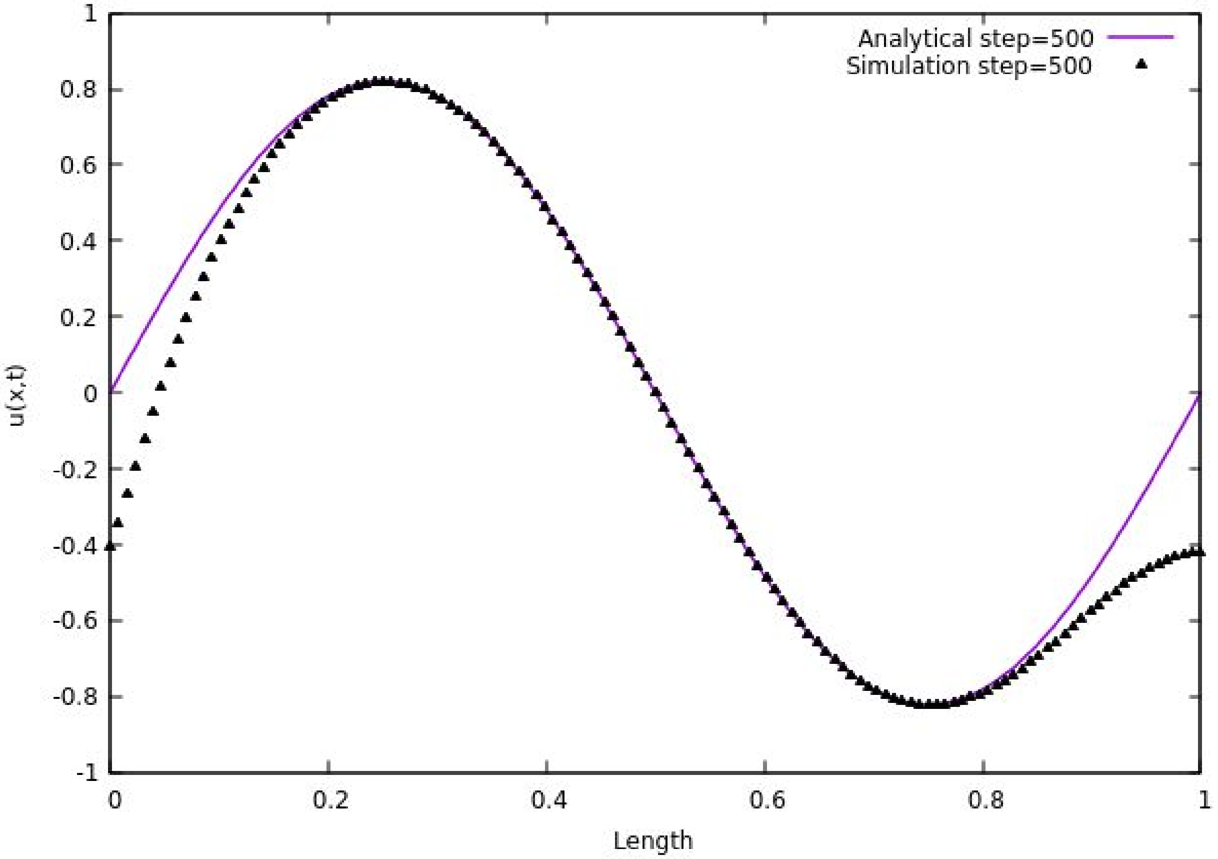
Average Error : 0.0000
Absolute Error : 0.000377

2) Time step of 100th sec



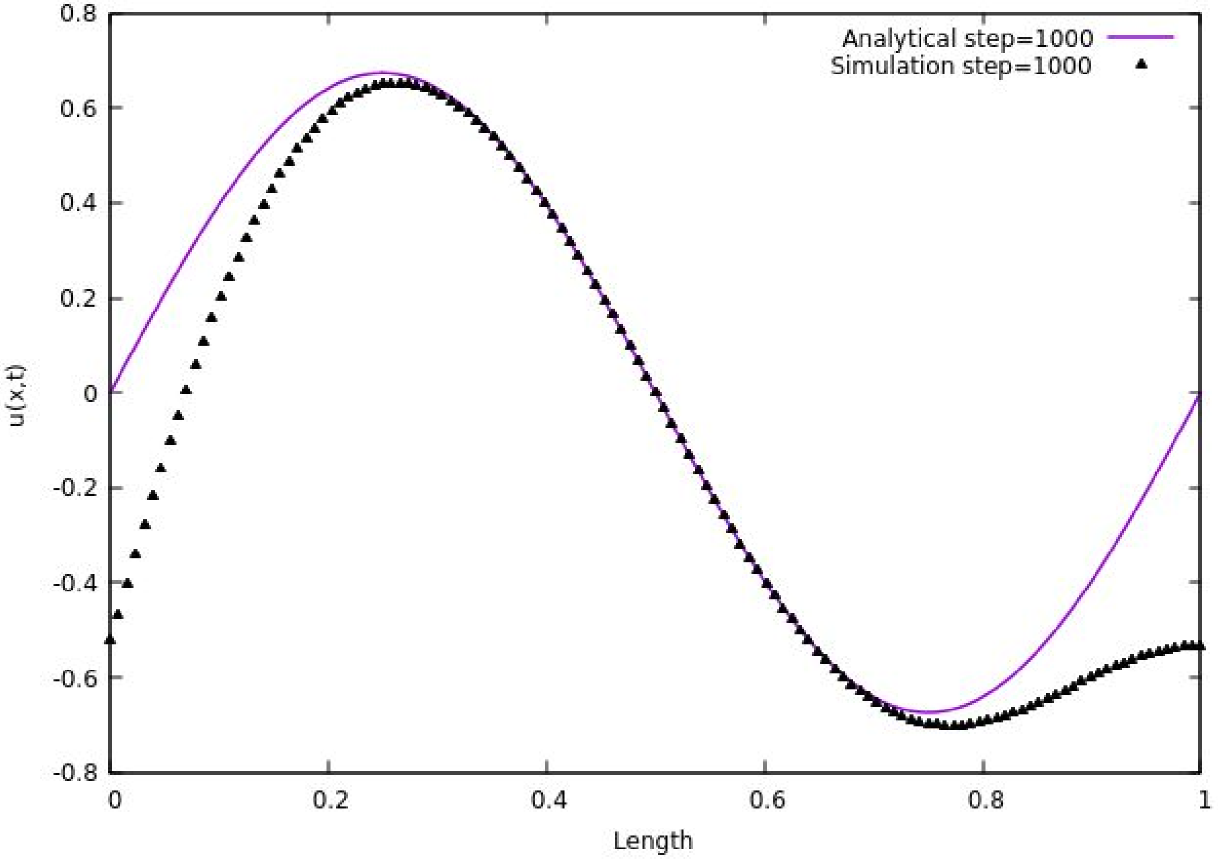
Average Error: 0.012319
Absolute Error: 0.012894

3) Time step of 500th sec



Average Error: 0.069327
Absolute Error: 0.069952

5) Time step of 1000th sec



Average Error : 0.173130
Absolute Error: 0.173755

Summary:

A code using “C” is developed to simulate 1D diffusion equation using explicit Euler and second order central difference scheme. Analytical solution of the problem is derived. I compared the analytical solution for the diffusion equation with that of numerical solution. Using the boundary condition $u[0] = u[n] = 0$ and from the plots I observe that both simulated and analytical solutions have the same phase and negligible error. Average error and absolute error for different time scales are calculated to know the variation of numerical solution from that of the analytical solution.Though the plots show that both simulated and analytical scheme matches well, sometimes with zero error, the values of absolute error shows that there is variation amongst them. I suggest this is due to the wave nature of the solution.The error in one half of the cycle is nullified by the other half of the cycle. Similarly using the periodic boundary condition $u[0] = u[n]$, I observe that both simulated and analytical solutions are differing by a greater degree of error. From the plots, it can be observed that the amplitude in the first half of the cycle is greater than the second half. This indicates that the principle of conservation of mass is violated for this boundary condition. Thus I assume that the boundary condition $u[0] = u[n] = 0$ suits well with the conditions under which analytical solution is derived.