Post_processing

April 30, 2023

1 Question (1)

1.1 Consider the viscous Burgers' equation $u_t + uu_x = \alpha u_{xx}$, where u(x, t) is the velocity component along the x−direction, and is the kinematic viscosity. Solve this 1D equation in a periodic domain of size 1.0 for the following cases. Use $\Delta t = 0.0004$, $t_{end} = 0.075$, and $u(x, 0) = sin(4\pi x) + sin(6\pi x) + sin(10\pi x)$.

1.2 (a) Use Euler and first order upwind schemes to solve the equation with $\alpha = 0$ for a grid resolution of 64 and 1024. Compare the two results by plotting the solution for a few timesteps, and comment on the errors.

2 Answer (1)(a):

3 Import the necessary libraries

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

4 Reading the name of the files generated

[2]:
```
Output_files = pd.read_csv("Output_file_names.csv",delimiter=",",header=None).
 ↪to_numpy()
Output_files = np.squeeze(Output_files)
Output_files = Output_files.tolist()
```

4.1 Output files generated

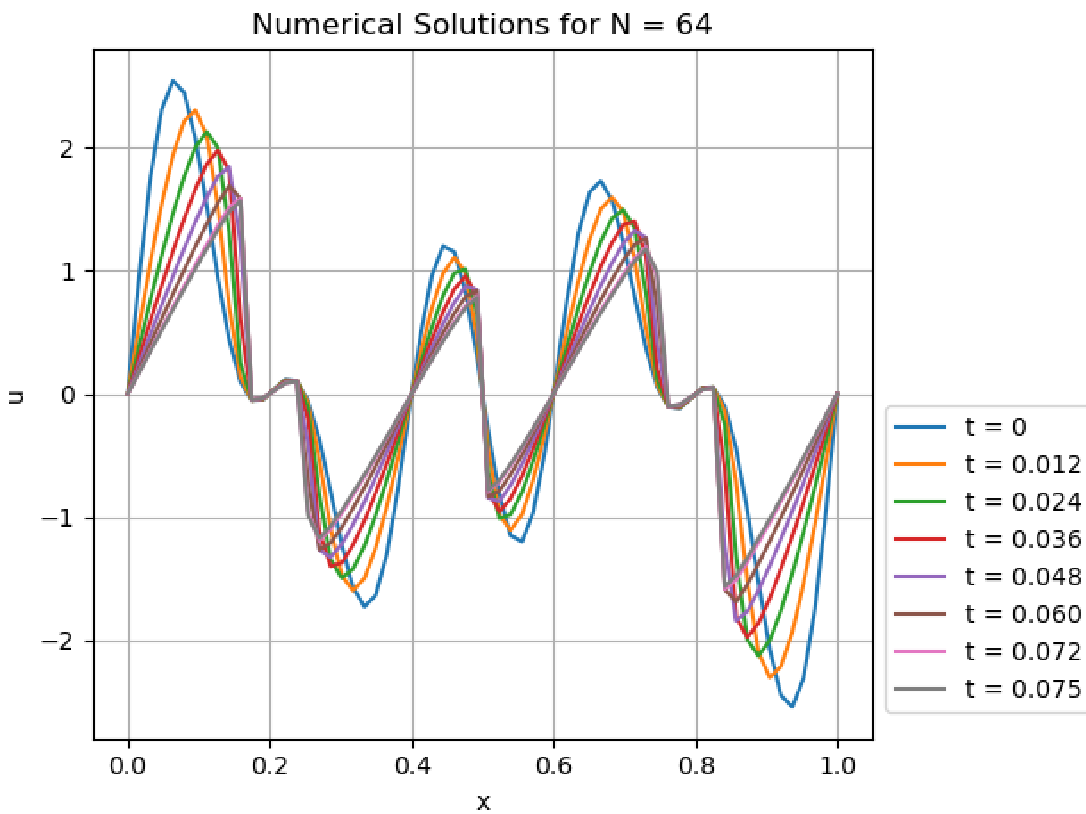
[3]:
```
Output_files
```

[3]:
```
['Question_1_a_u_n_t_0.000000_N_64_.csv',
 'Question_1_a_u_n_t_0.012000_N_64_.csv',
 'Question_1_a_u_n_t_0.024000_N_64_.csv',
 'Question_1_a_u_n_t_0.036000_N_64_.csv',
 'Question_1_a_u_n_t_0.048000_N_64_.csv',
```

```
 'Question_1_a_u_n_t_0.060000_N_64_.csv',
 'Question_1_a_u_n_t_0.072000_N_64_.csv',
 'Question_1_a_u_n_t_0.075000_N_64_.csv',
 'Question_1_a_u_n_t_0.000000_N_1024_.csv',
 'Question_1_a_u_n_t_0.012000_N_1024_.csv',
 'Question_1_a_u_n_t_0.024000_N_1024_.csv',
 'Question_1_a_u_n_t_0.036000_N_1024_.csv',
 'Question_1_a_u_n_t_0.048000_N_1024_.csv',
 'Question_1_a_u_n_t_0.060000_N_1024_.csv',
 'Question_1_a_u_n_t_0.072000_N_1024_.csv',
 'Question_1_a_u_n_t_0.075000_N_1024_.csv']
```

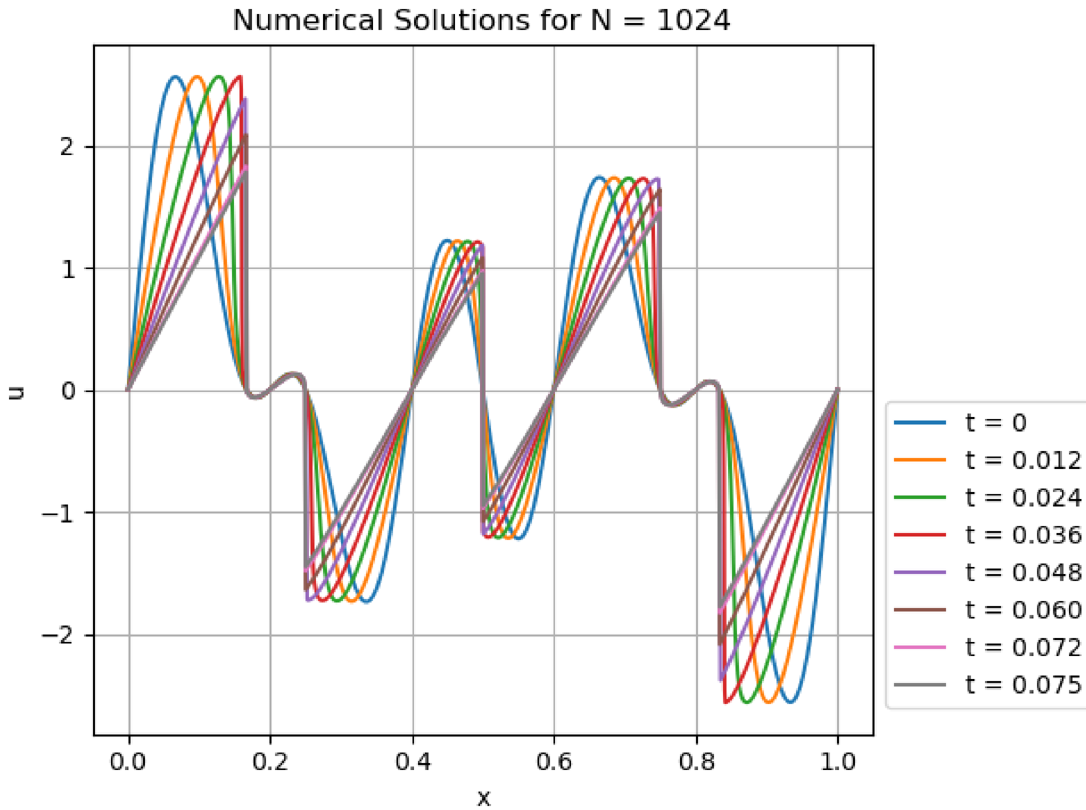
4.2 Numerical solution for grid resolution of 64

[4]:
```
for output_file in Output_files:
    if "_N_64_" in output_file:
        Numerical_Solution = pd.read_csv(output_file,delimiter=",",header=None).
 ↪to_numpy()
        x = np.linspace(0, 1,Numerical_Solution.shape[0])
        plt.plot(x,Numerical_Solution)
plt.xlabel("x")
plt.ylabel("u")
plt.legend(["t = 0","t = 0.012","t = 0.024","t = 0.036","t = 0.048","t = 0.
 ↪060","t = 0.072","t = 0.075"],bbox_to_anchor=(1,0.5))
plt.title("Numerical Solutions for N = 64")
plt.grid()
plt.tight_layout()
plt.savefig("Question_1_a_N_64.png",dpi = 500)
plt.show()
```



4.3 Numerical solution for grid resolution of 1024

[5]:
```
for output_file in Output_files:
    if "_N_1024_" in output_file:
        Numerical_Solution = pd.read_csv(output_file,delimiter=",",header=None).
 ↪to_numpy()
        x = np.linspace(0, 1,Numerical_Solution.shape[0])
        plt.plot(x,Numerical_Solution)
plt.xlabel("x")
plt.ylabel("u")
plt.legend(["t = 0","t = 0.012","t = 0.024","t = 0.036","t = 0.048","t = 0.
 ↪060","t = 0.072","t = 0.075"],bbox_to_anchor=(1,0.5))
plt.title("Numerical Solutions for N = 1024")
plt.grid()
plt.tight_layout()
plt.savefig("Question_1_a_N_1024.png",dpi = 500)
plt.show()
```



4.4 Comment on the errors:

4.4.1 1. From the above graph it is apparent that the error is dissipative in nature.

4.4.2 2. In case of grid resolution of 64 the dissipation is more than that of grid resolution of 1024.

4.4.3 3. We are also getting the same result about the nature of the error using the modified equation.