CFD Project 1

Bilal Abdul Halim

November 30, 2020

1 Introduction

Given a tube geometry filled with a fluid of uniform velocity, pressure and density. Using the Lax Wendroff scheme, I was able to solve the linear wave equation

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \tag{1}$$

through writing a computer code using Python. Here, c = 0.5, L = 1, and I took $Nx = 1001$, which yields $\Delta x = \frac{1}{Nx-1} = 0.001$. The scheme will also be running 10000 time steps.

2 Developing Lax Wendroff Scheme

The Lax-Wendroff method was named after P.Lax and B.Wendroff. Now we start off by defining u_i^{j+1} using half time step

$$u_i^{j+\frac{1}{2}} = u_i^j + \frac{\Delta t}{2}\Big(- c\frac{\partial u}{\partial x}\Big|_{(i,j)}\Big)$$

$$u_i^{j+1} = u_i^j + \Delta t\Big(- c\frac{\partial u}{\partial x}\Big|_{(i,j+\frac{1}{2})}\Big)$$

From there, the central difference is used to approxmate the derivative $u_x|_{(i,j+\frac{1}{2})}$, for example

$$u_i^{j+1} = u_i^j - c\frac{\Delta t}{\Delta x}\Big(u_{i+\frac{1}{2}}^{j+\frac{1}{2}} - u_{i-\frac{1}{2}}^{j+\frac{1}{2}}\Big)$$

From the two terms $u_{i+\frac{1}{2}}^{j+\frac{1}{2}}$ and $u_{i-\frac{1}{2}}^{j+\frac{1}{2}}$, we are able to get a two-step scheme as follows:

$$u_{i-\frac{1}{2}}^{j+\frac{1}{2}} = \frac{1}{2}\Big(u_i^j + u_{i-1}^j\Big) - \frac{c\Delta t}{2\Delta x}\Big(u_i^j + u_{i-1}^j\Big)$$

$$u_{i+\frac{1}{2}}^{j+\frac{1}{2}} = \frac{1}{2}\Big(u_i^j + u_{i+1}^j\Big) - \frac{c\Delta t}{2\Delta x}\Big(u_{i+1}^j + u_i^j\Big)$$

$$u_i^{j+1} = u_i^j - \frac{c\Delta t}{\Delta x}\Big(u_{i+\frac{1}{2}}^{j+\frac{1}{2}} - u_{i-\frac{1}{2}}^{j+\frac{1}{2}}\Big)$$

And this can also be rewritten as

$$u_i^{j+1} = b_{-1}u_{i-1}^j + b_0 u_i^j + b_1 u_{i+1}^j \tag{2}$$

Here, b_{-1}, b_0 and b_1 are constants such that:

$$b_{-1} = \frac{\alpha}{2}(\alpha + 1)$$

$$b_0 = 1 - \alpha^2$$

$$b_1 = \frac{\alpha}{2}(\alpha - 1)$$

and α is the Courant number.

2.1 Von Neumann Stability Analysis

For equation 2, the amplification factor $G(k)$ becomes

$$G(k) = \big(1 + \alpha^2(\cos(k\Delta x) - 1)\big) - i\alpha\sin(k\Delta x)$$

,

therefore

$$|G(k)|^2 = 1 - \alpha^2(1 - \alpha^2)\big(1 - \cos(k\Delta x)\big)^2$$

And so, we can conclude that the stability condition becomes the CFL condition itself,

$$1 - \alpha^2 \geq 0$$

$$\alpha = c\frac{\Delta x}{\Delta t} \leq 1$$

3 Results

Because I wanted to see how my code behaves and if it shows correct behavior, I went ahead and conducted two parametric studies. The first one comprised of changing the CFL number while keeping every parameter the same. The second one however, was comprised of changing the total run simulation time while keeping everything the same. One quick disclaimer before I get into the two studies is that the code I wrote behaves in such a way that there are infinitely many discontinuities in the velocity, every 0.5 m. What I mean by that is the exact solution is kind of like a step function ranging from 1 m/s to 0.5 m/s but keeps on changing every 0.5 m, and so the scheme follows that. I say this because there are finalillations at the far lef and right of the figure, which are caused by that reason.

3.1 CFL Parametric Study

As I mentioned previously, this study is comprised of me only changing the CFL and keeping all parameters the same. CFL number ranged from 0.1 to 0.99 with a step size of 0.1, excep for thet last measurement from 0.9 to 0.99, which had a step size of 0.01. Below are the parameters and their corresponding values used in this study:

- $N_x = 1001$; Number of grid points.
- $L = 1$; Length of pipe.
- $dx = \frac{L}{N_x - 1}$; Mesh size.
- $dt = dt$; Time step chosen in a way to change CFL.
- $c = 0.5$; Advection speed.
- $CFL = c\frac{dt}{dx}$; CFL number.
- $t = 4$; Simulation time.

By looking at all the figures, and observing the dominant trend, it can be seen that the solution converges to the exact solution as the CFL number increases. In layman's terms, the CFL number is basically a condition to restrict the information to only travel to it's adjacent node, and not any further. I also plotted the change in the error TE as a function of the CFL number, which can be see in figure 11 .

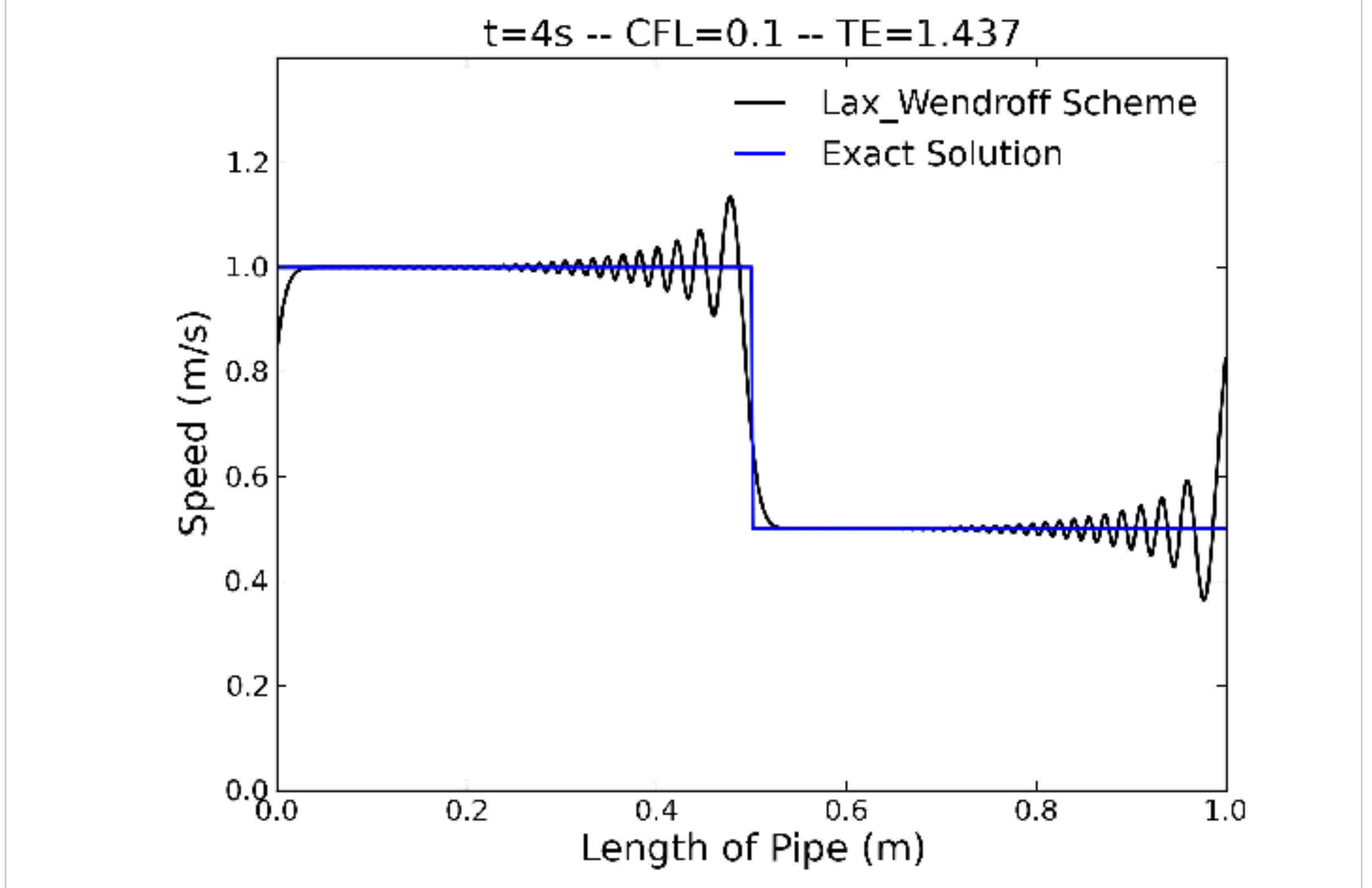

Figure 1: Solution for CFL number 0.1 and simulation time 4s

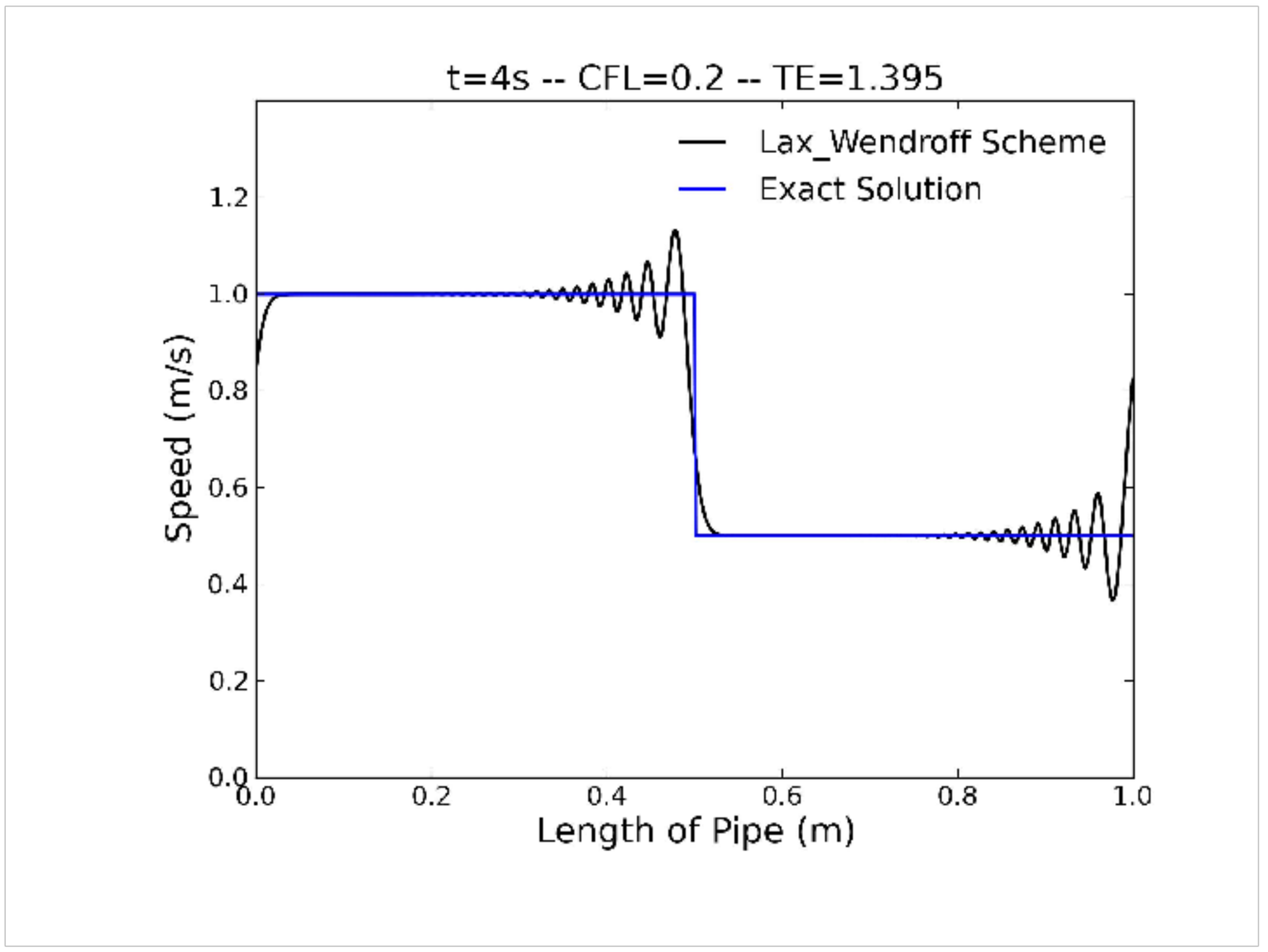

Figure 2: Solution for CFL number 0.2 and simulation time 4s

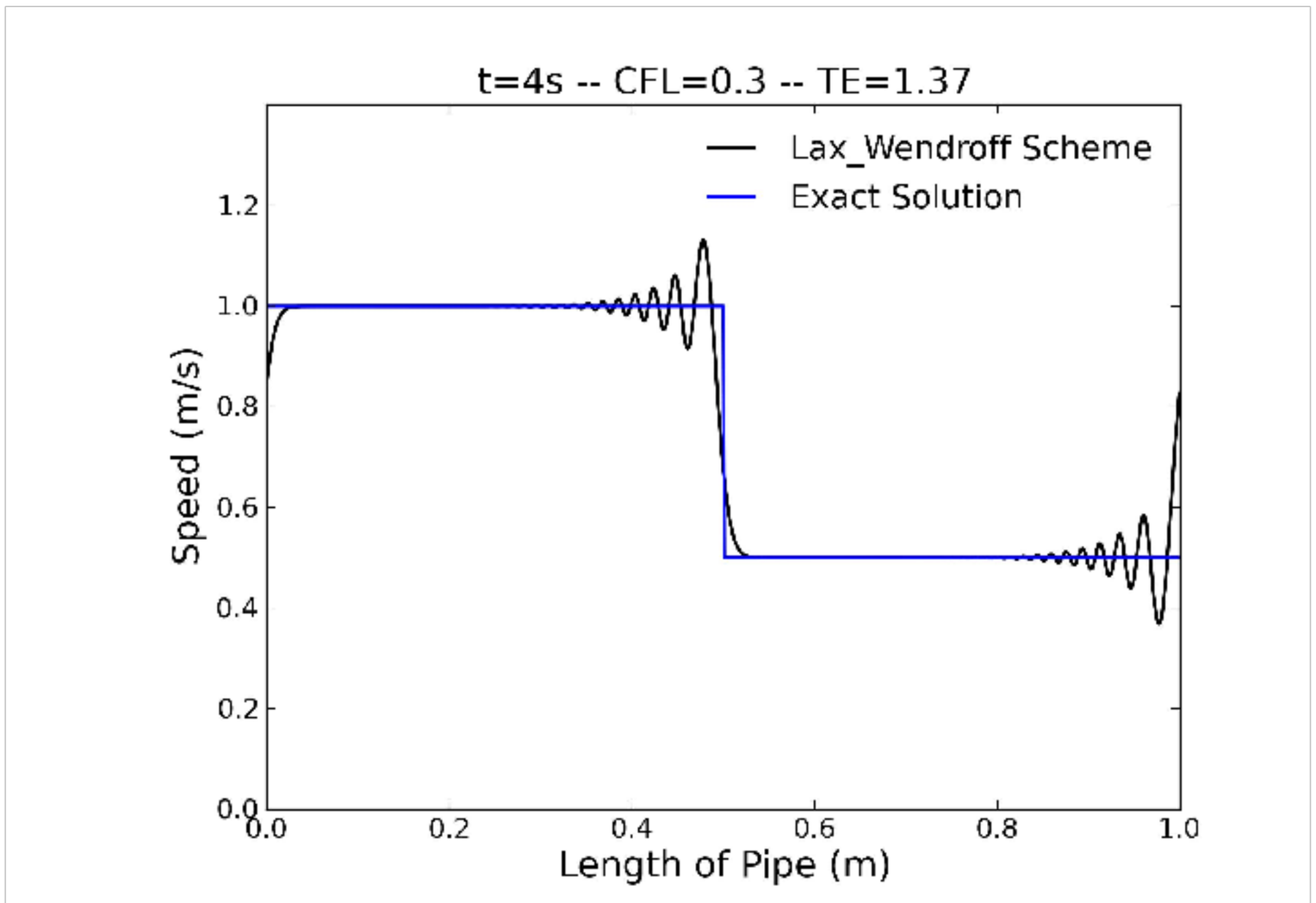

Figure 3: Solution for CFL number 0.3 and simulation time 4s

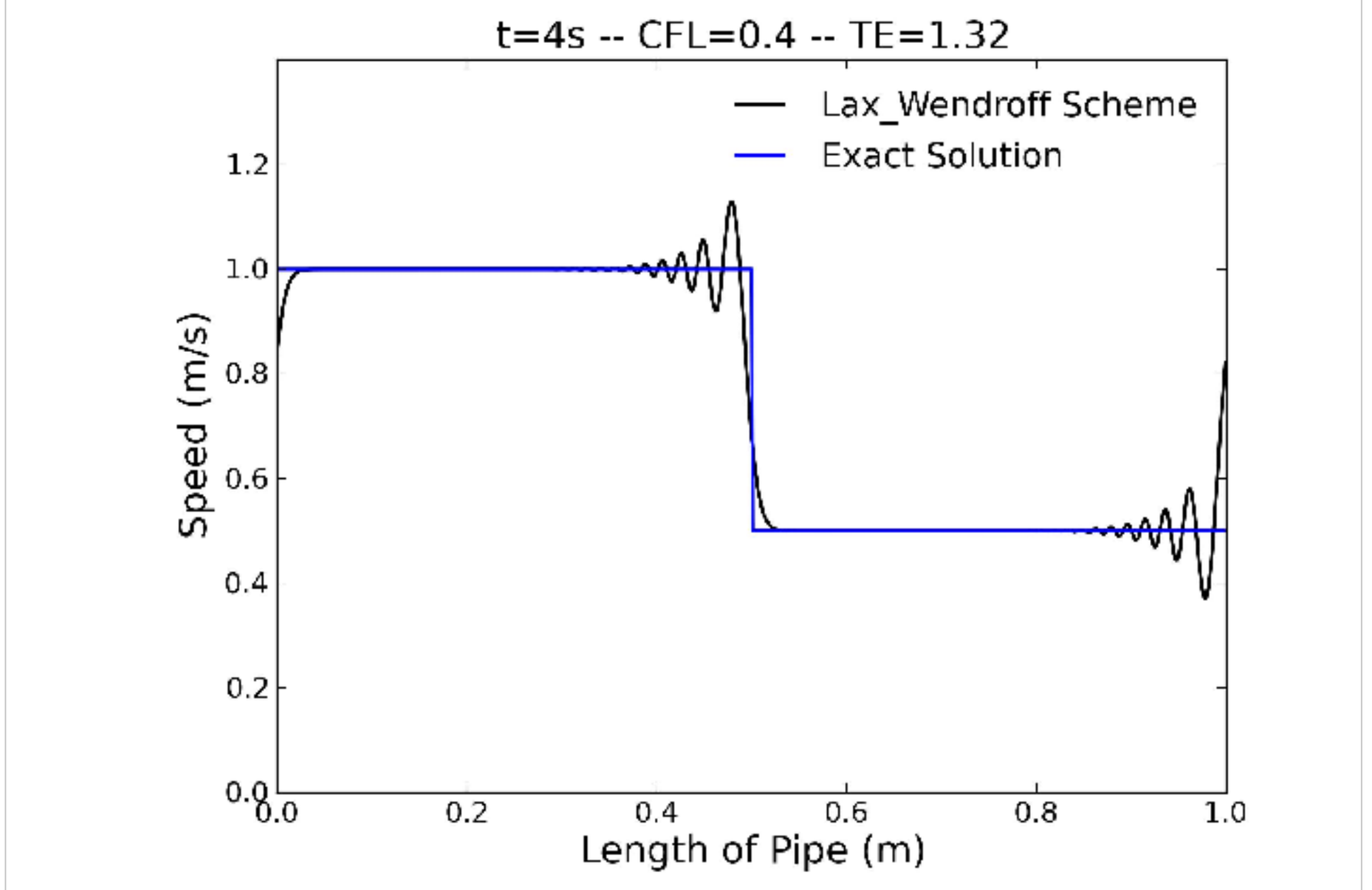

Figure 4: Solution for CFL number 0.4 and simulation time 4s

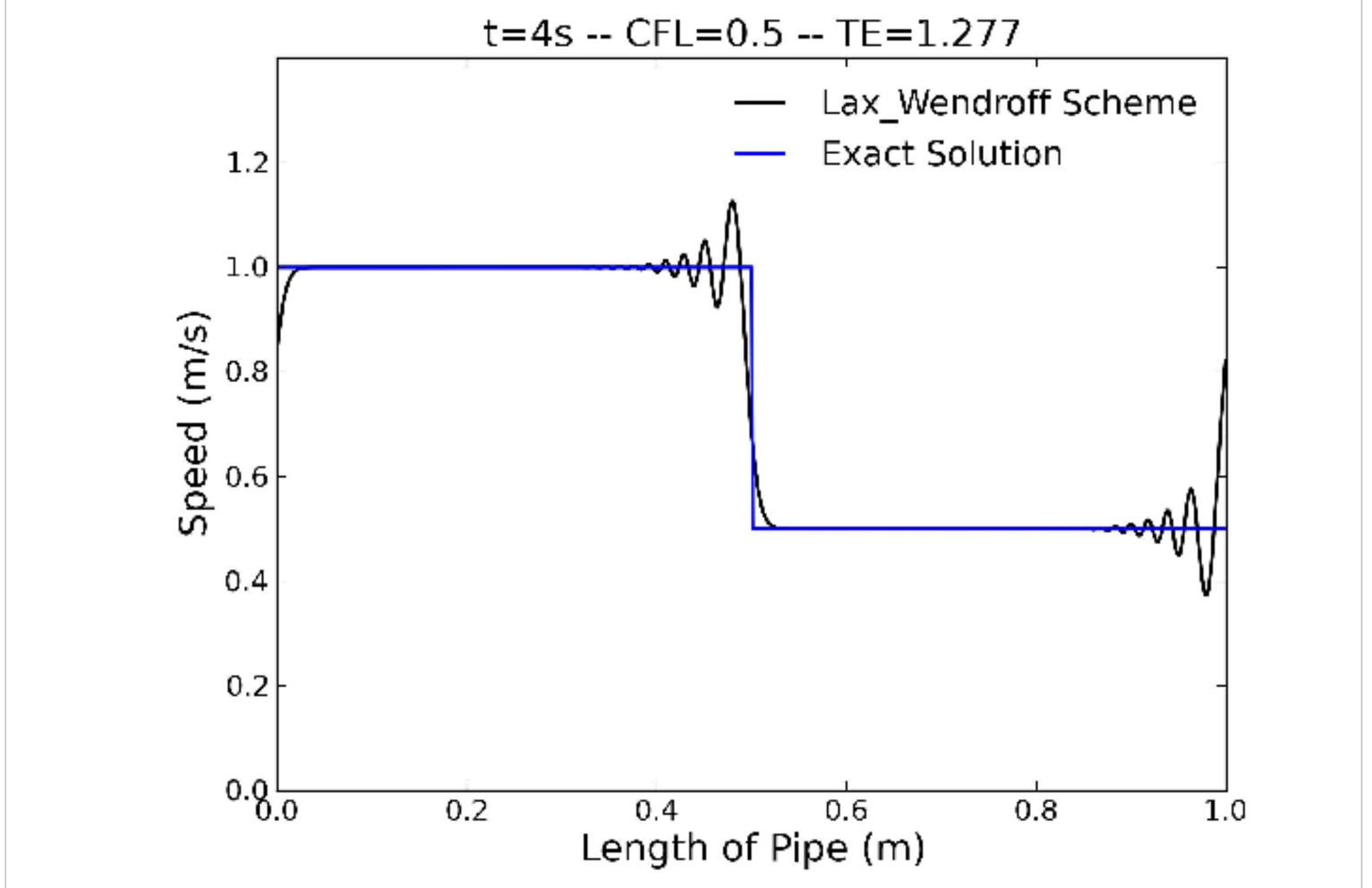

Figure 5: Solution for CFL number 0.5 and simulation time 4s

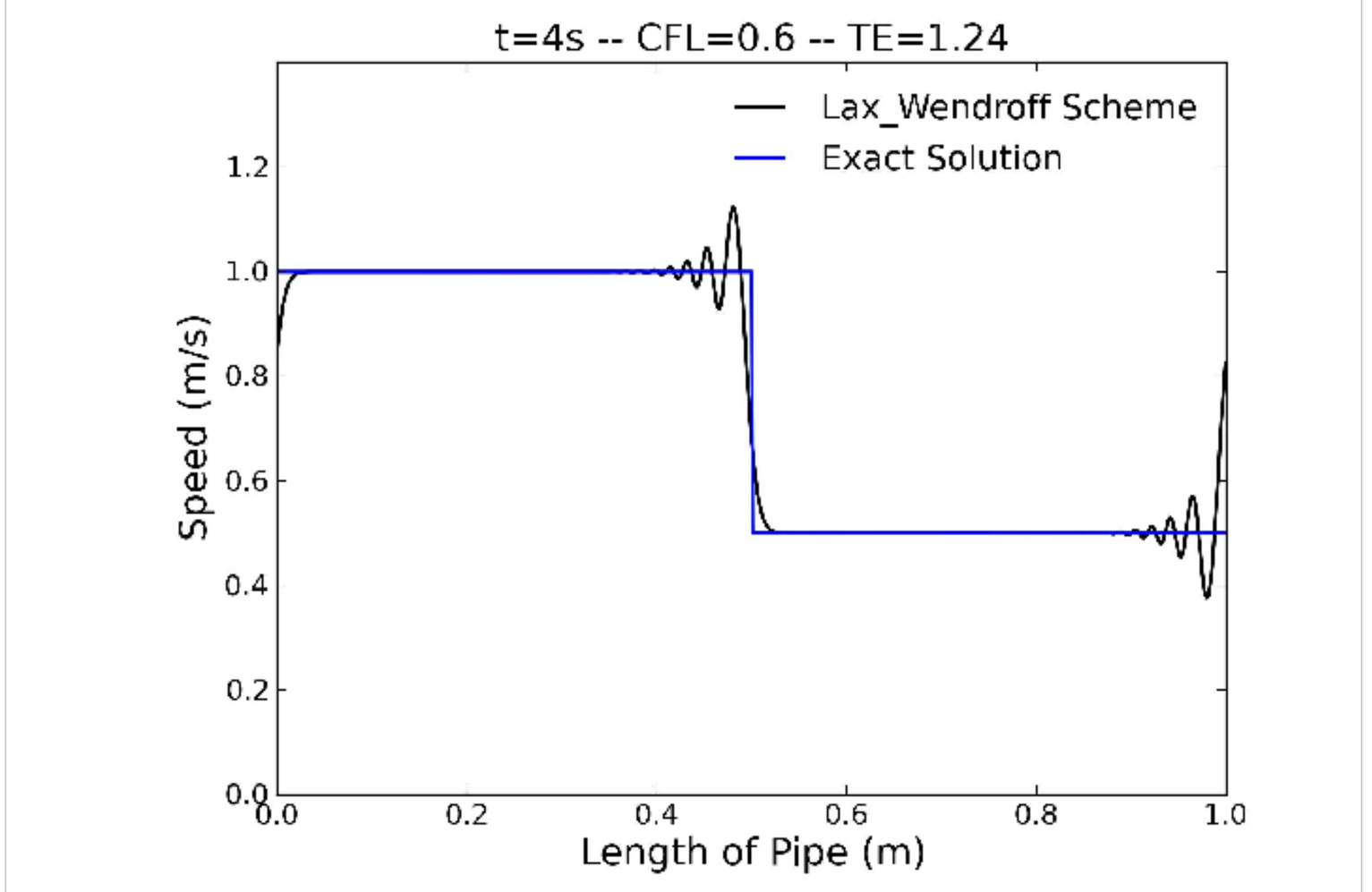

Figure 6: Solution for CFL number 0.6 and simulation time 4s

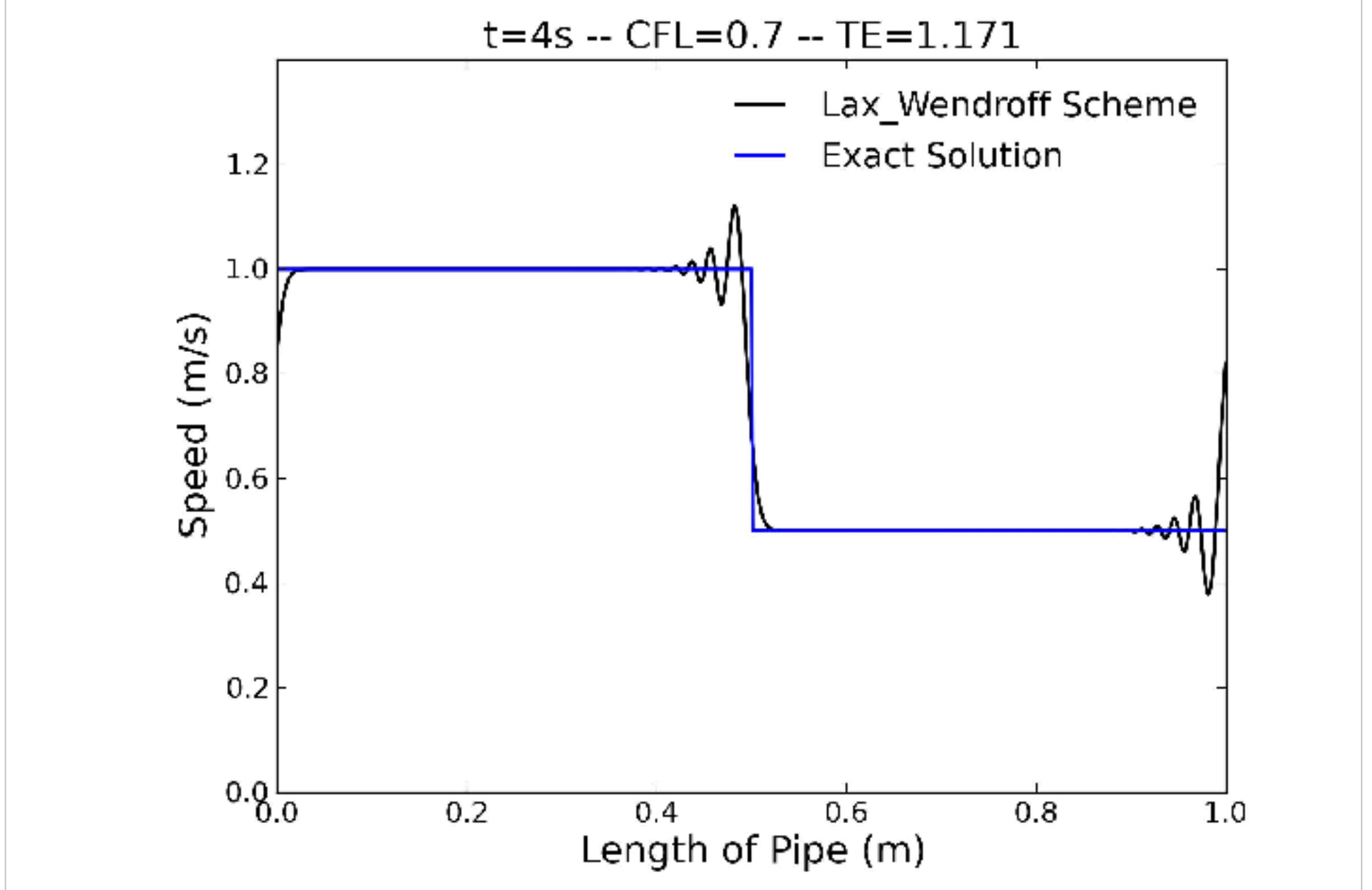

Figure 7: Solution for CFL number 0.7 and simulation time 4s

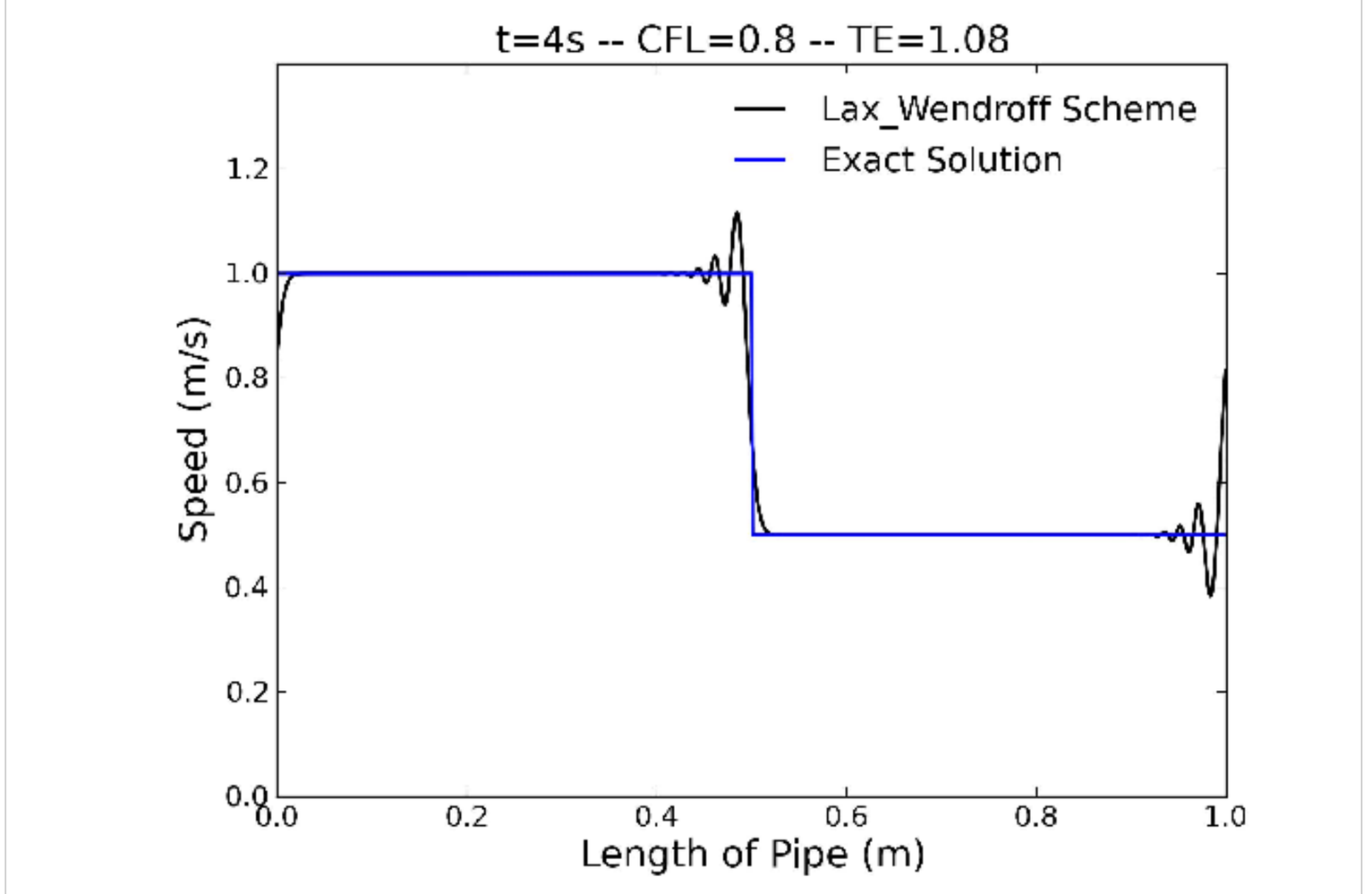

Figure 8: Solution for CFL number 0.8 and simulation time 4s

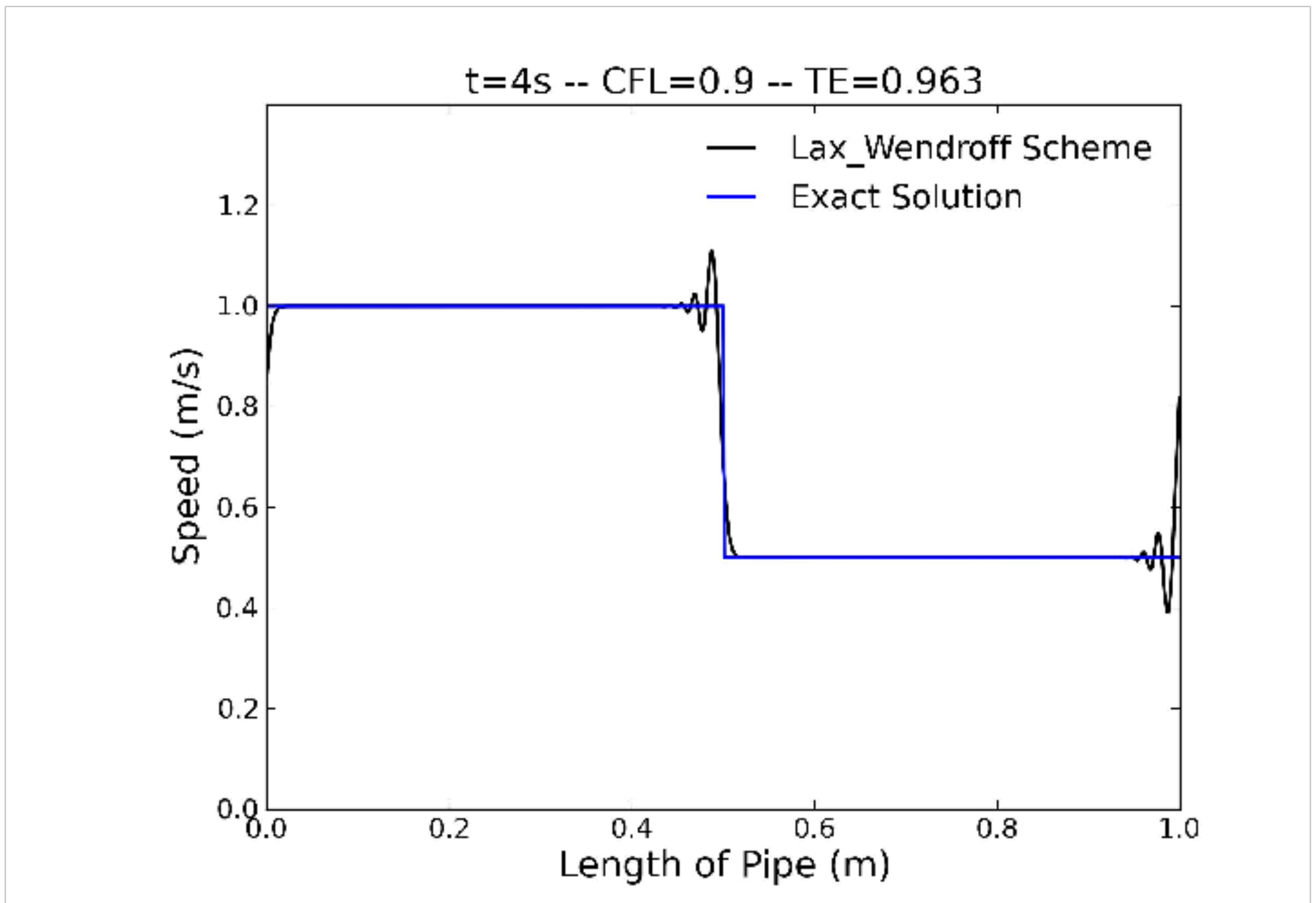

Figure 9: Solution for CFL number 0.9 and simulation time 4s

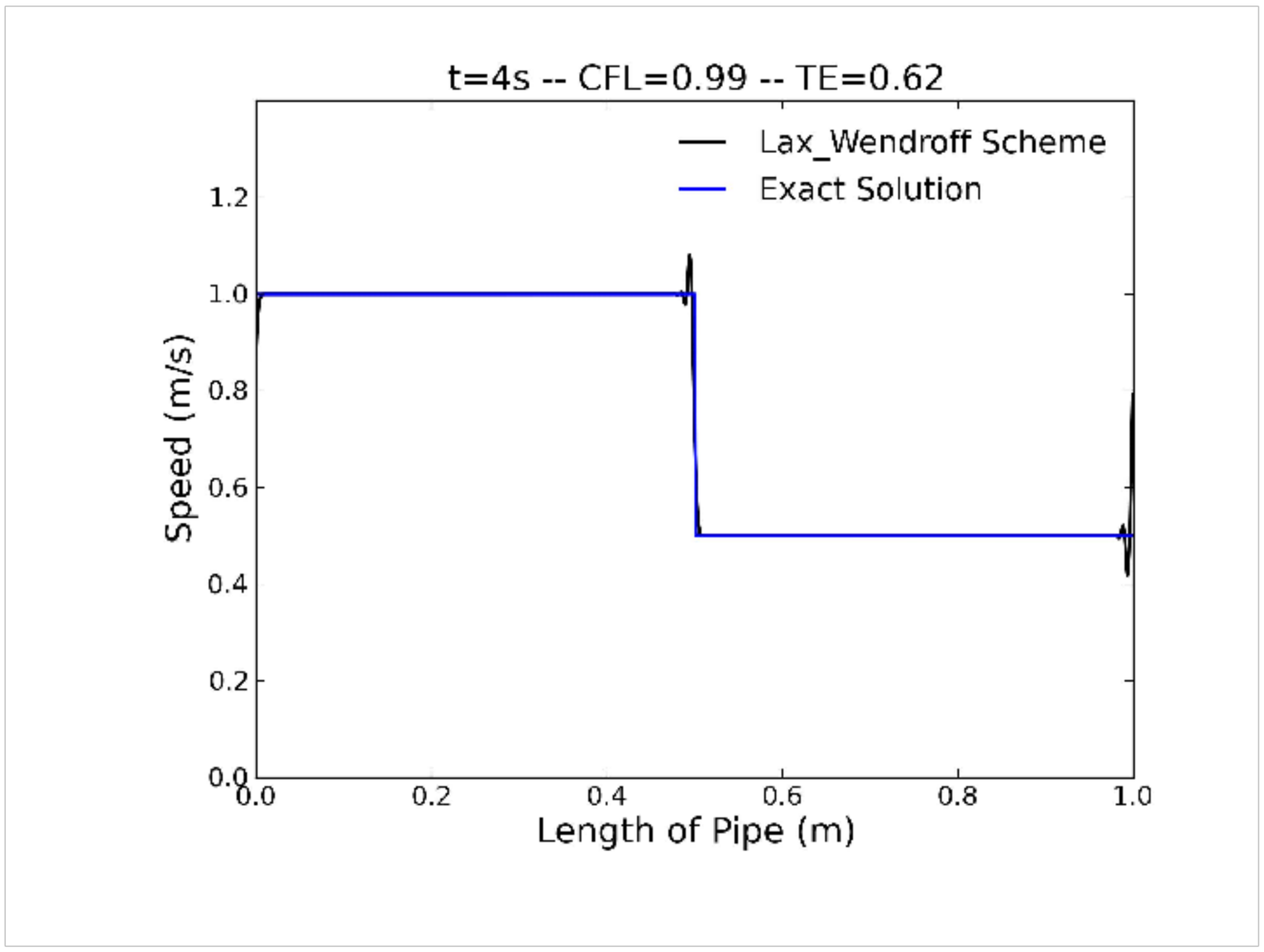

Figure 10: Solution for CFL number 0.99 and simulation time 4s

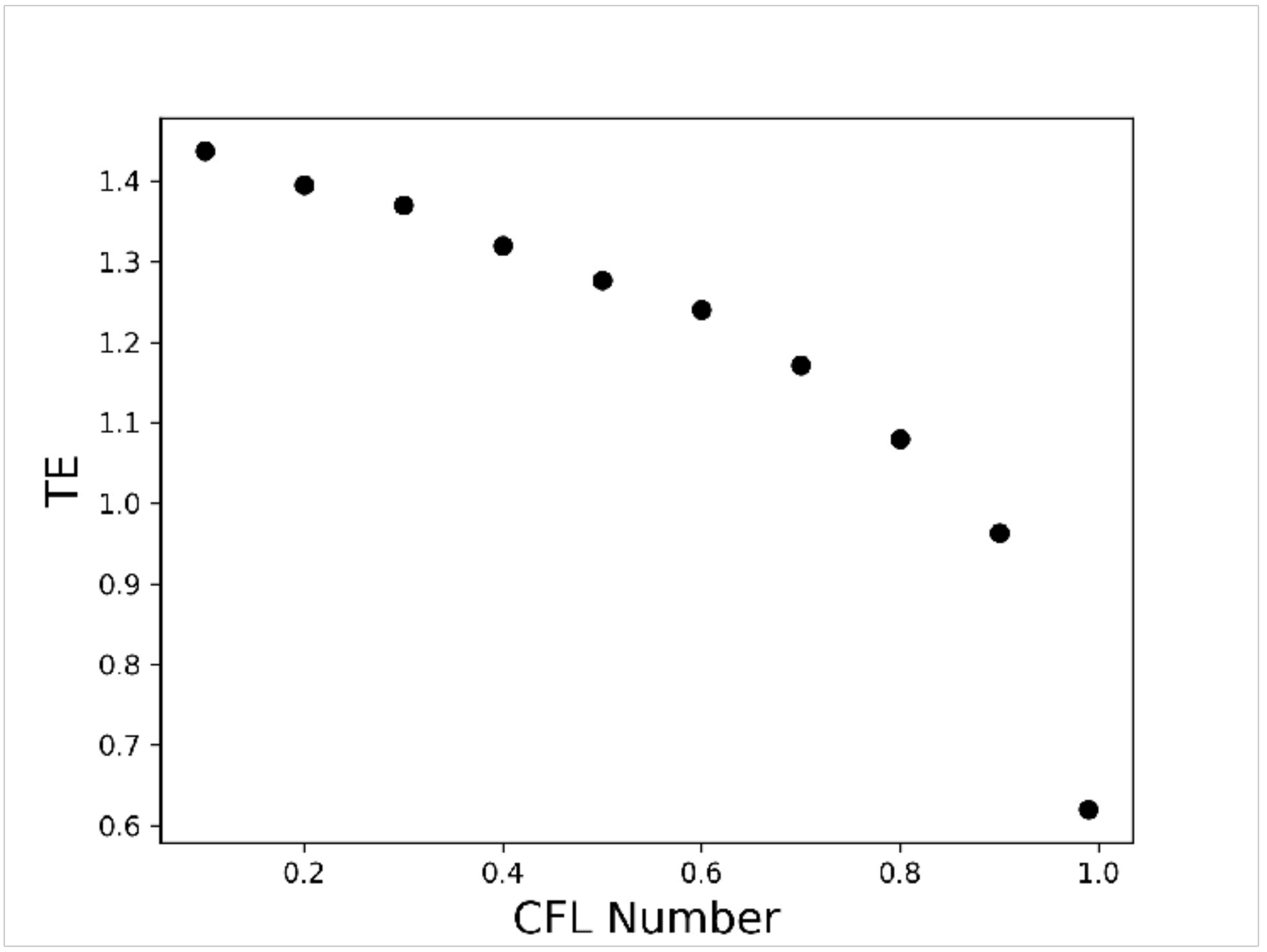

Figure 11: TE vs CFL number

3.2 Simulation Time Parametric Study

Another parametric study I ran was the total time parametric study. Basically letting the code to run for whatever time while keeping other parameters constant. Those parameters were mentioned before in a list, and I also used CFL number to be 0.9, as I got a decent solution with that. What I observed when running the code is that the longer I ran the code, the more it diverged from the solution and it's TE kept increasing more and more. I think that's the case because I considered fixed boundary conditions on both ends of the pipe.

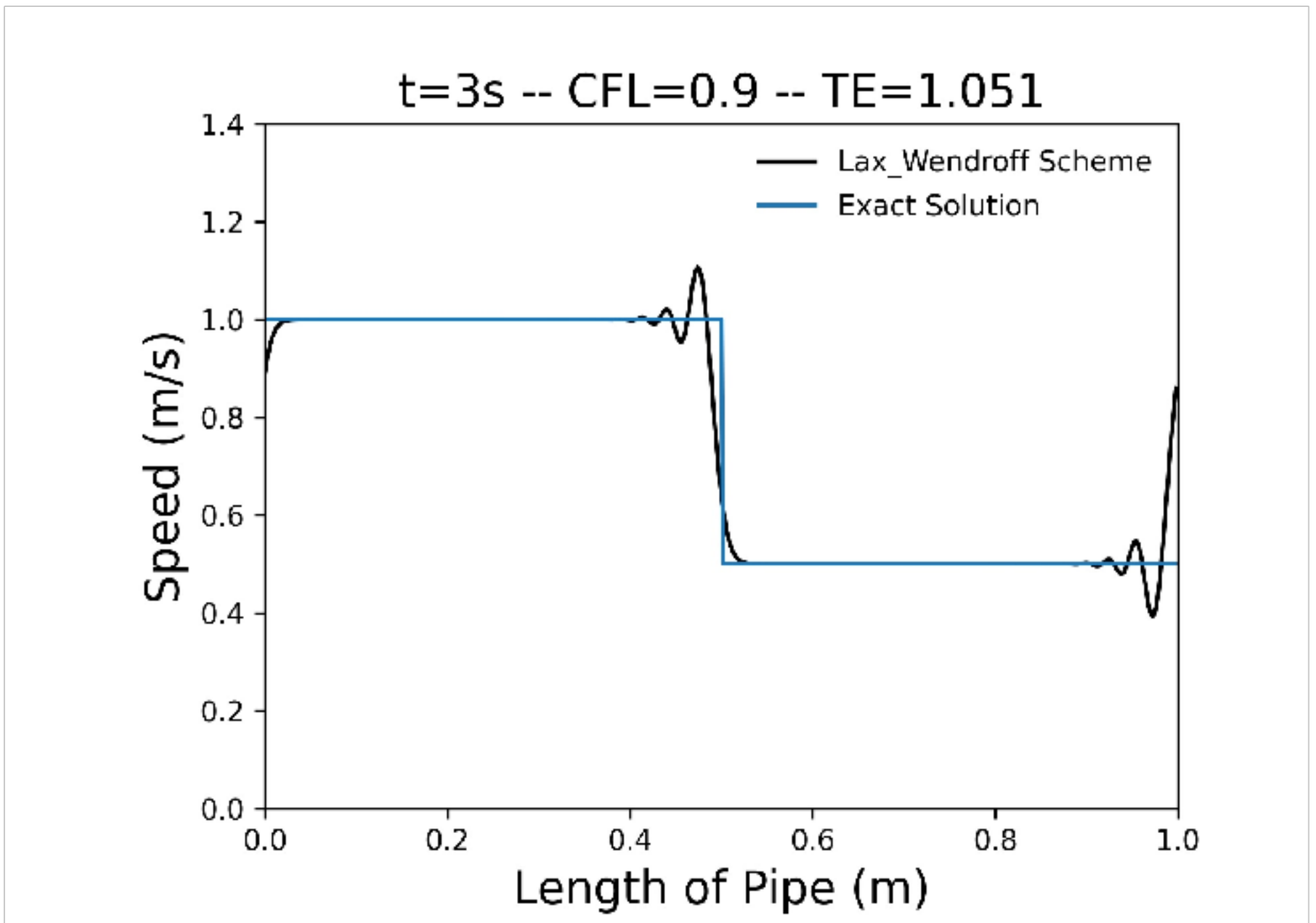

Figure 12: Total simulation time 3s and CFL 0.9

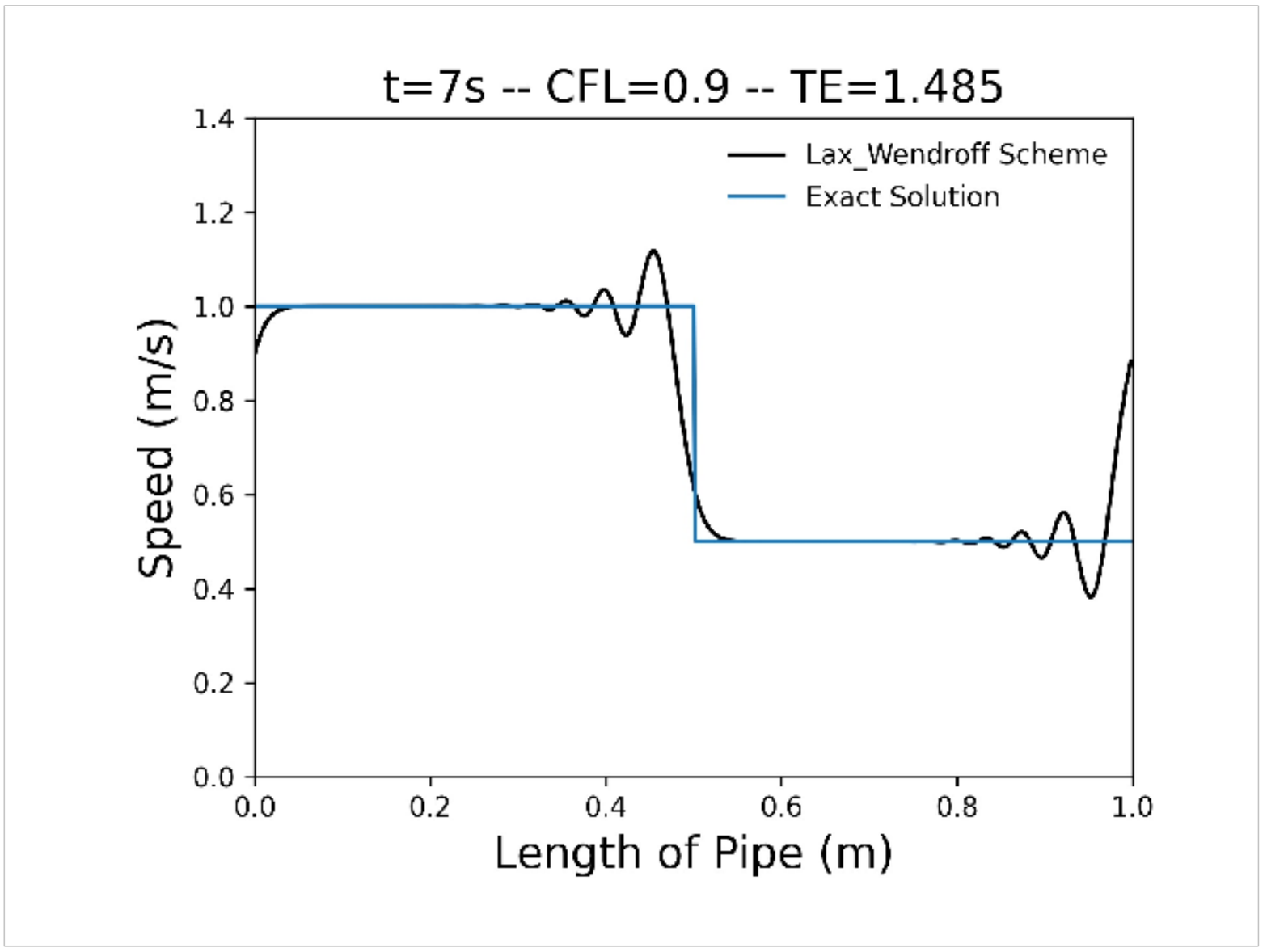

Figure 13: Total simulation time 7s and CFL 0.9

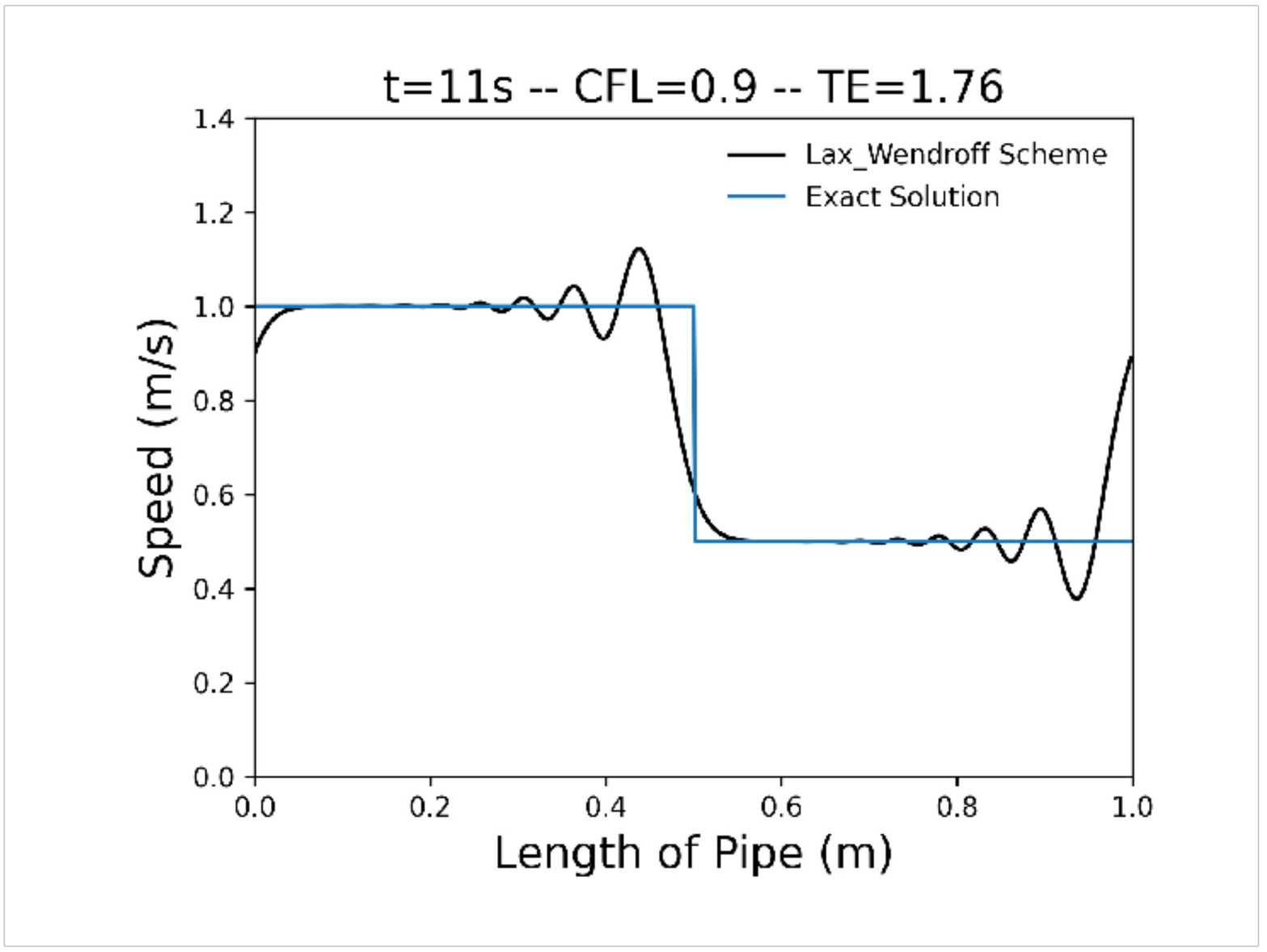

Figure 14: Total simulation time 11s and CFL 0.9

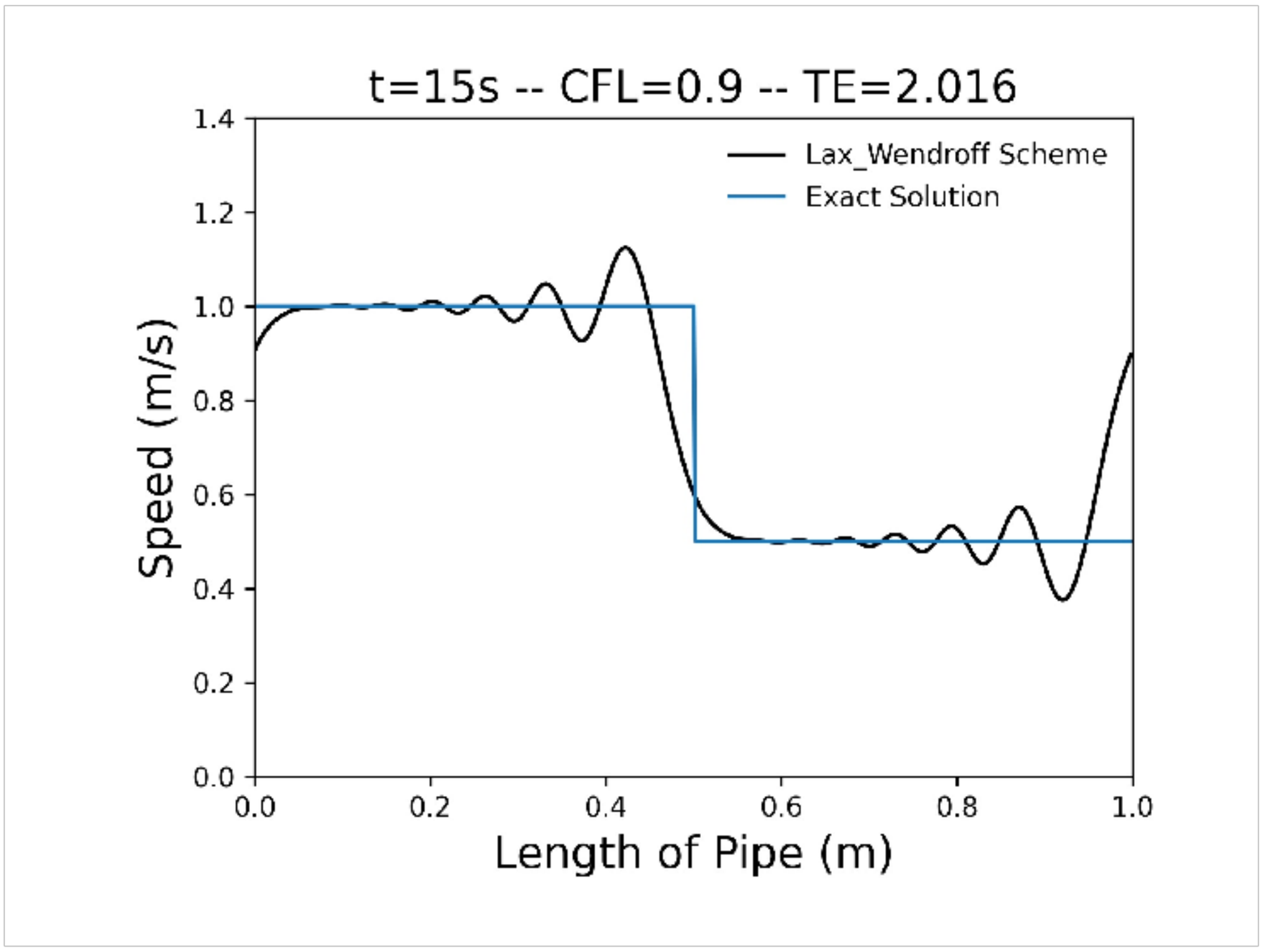

Figure 15: Total simulation time 15s and CFL 0.9

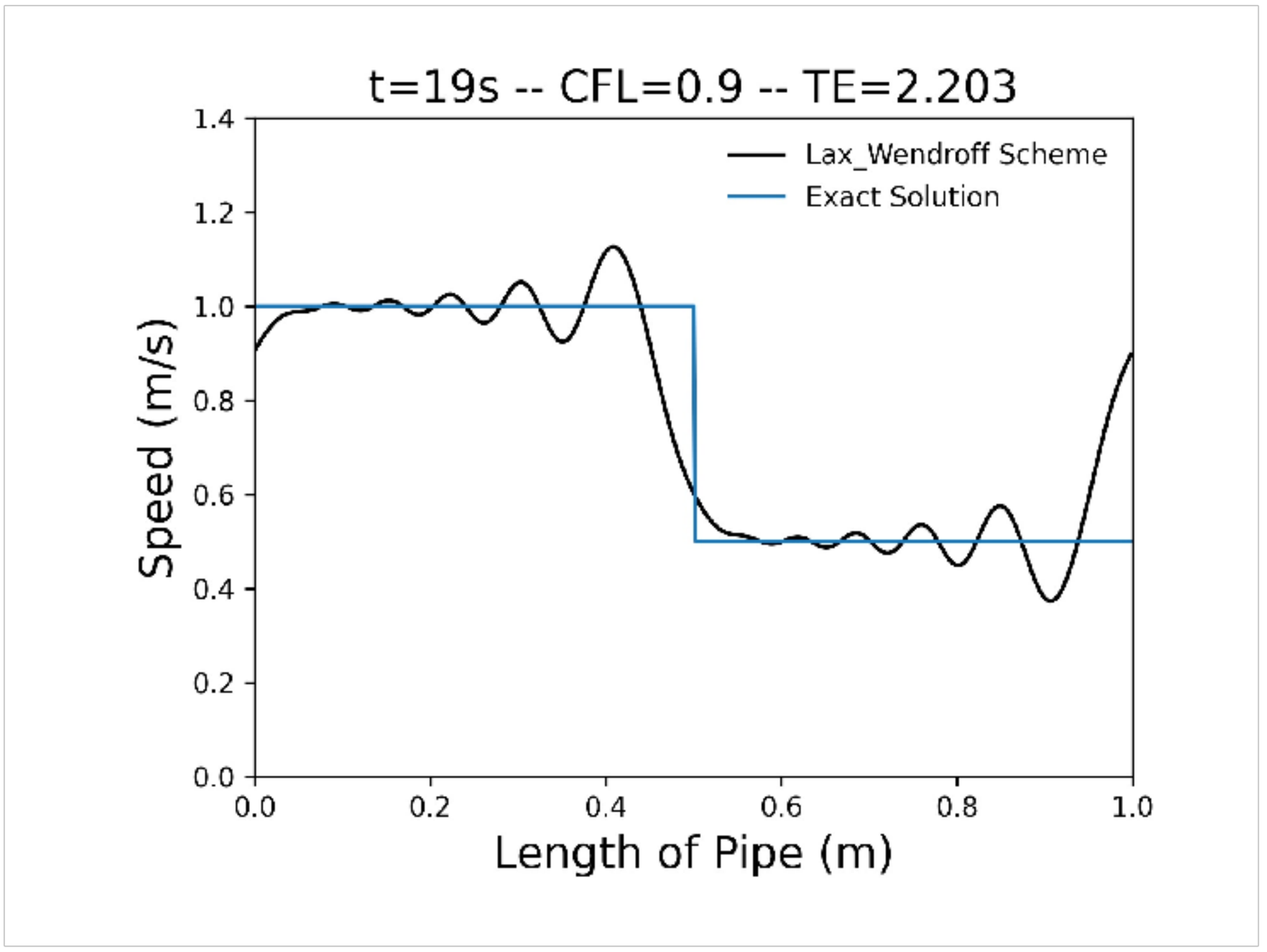

Figure 16: Total simulation time 19s and CFL 0.9

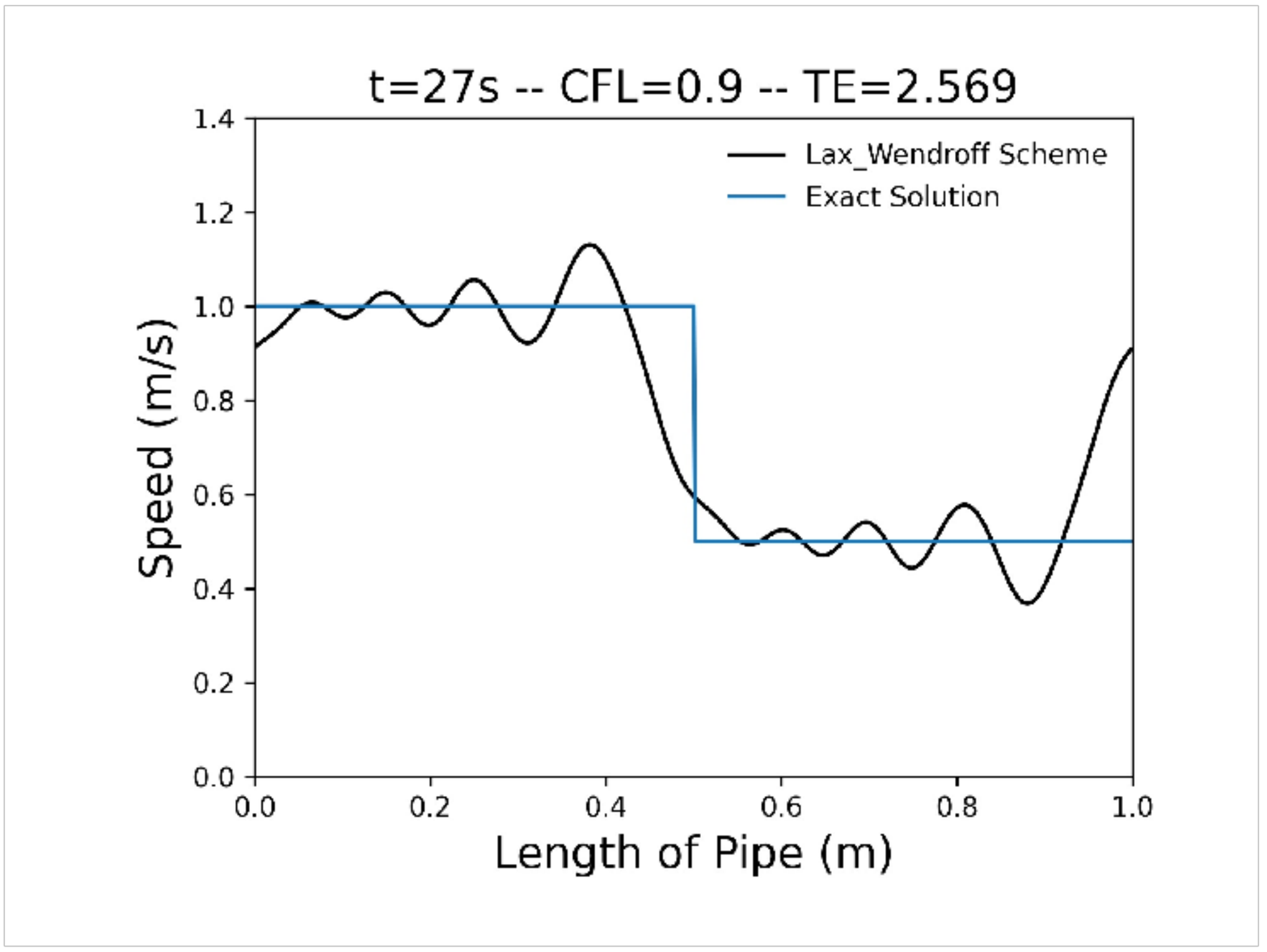

Figure 17: Total simulation time 27s and CFL 0.9

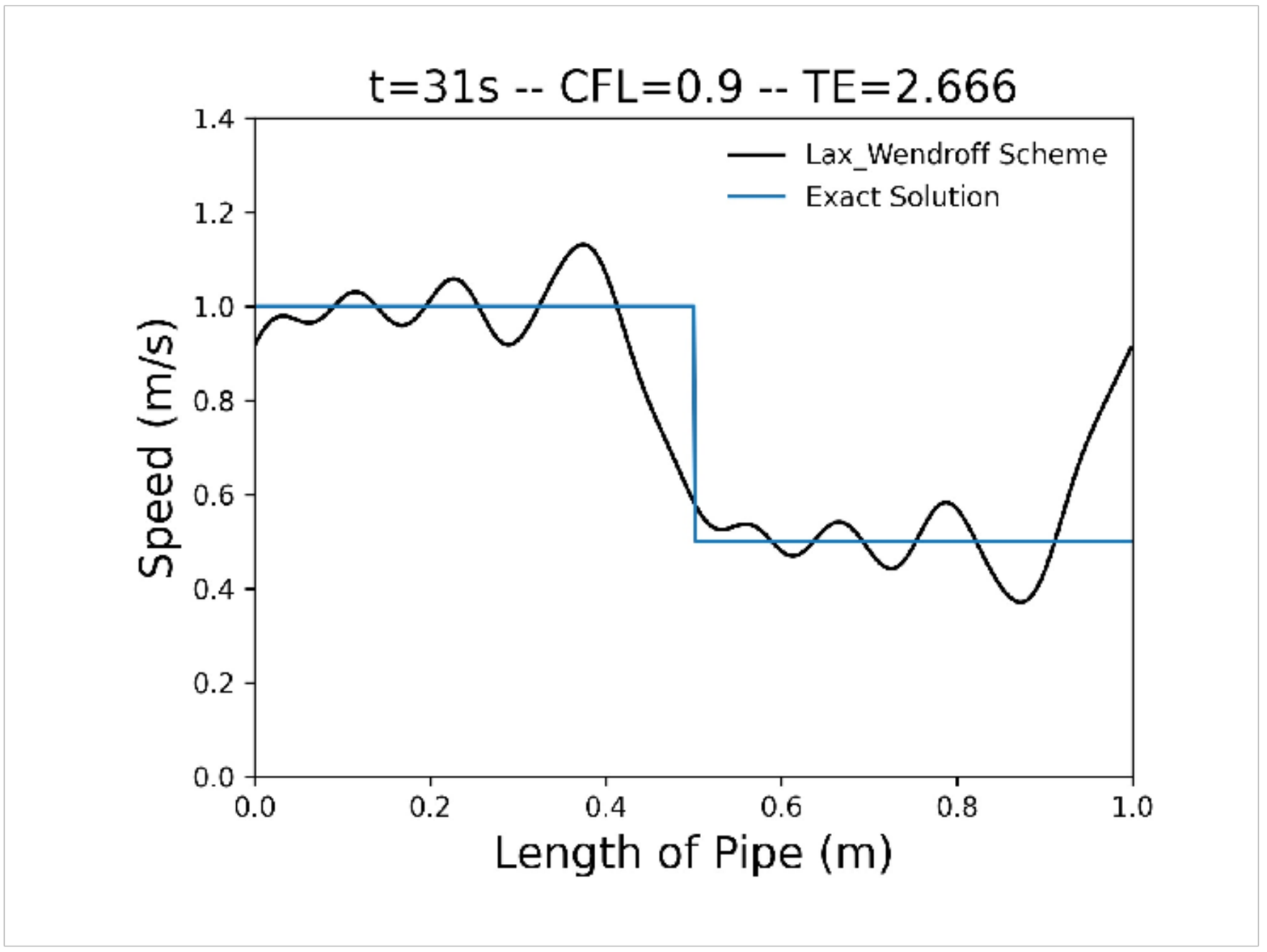

Figure 18: Total simulation time 31s and CFL 0.9

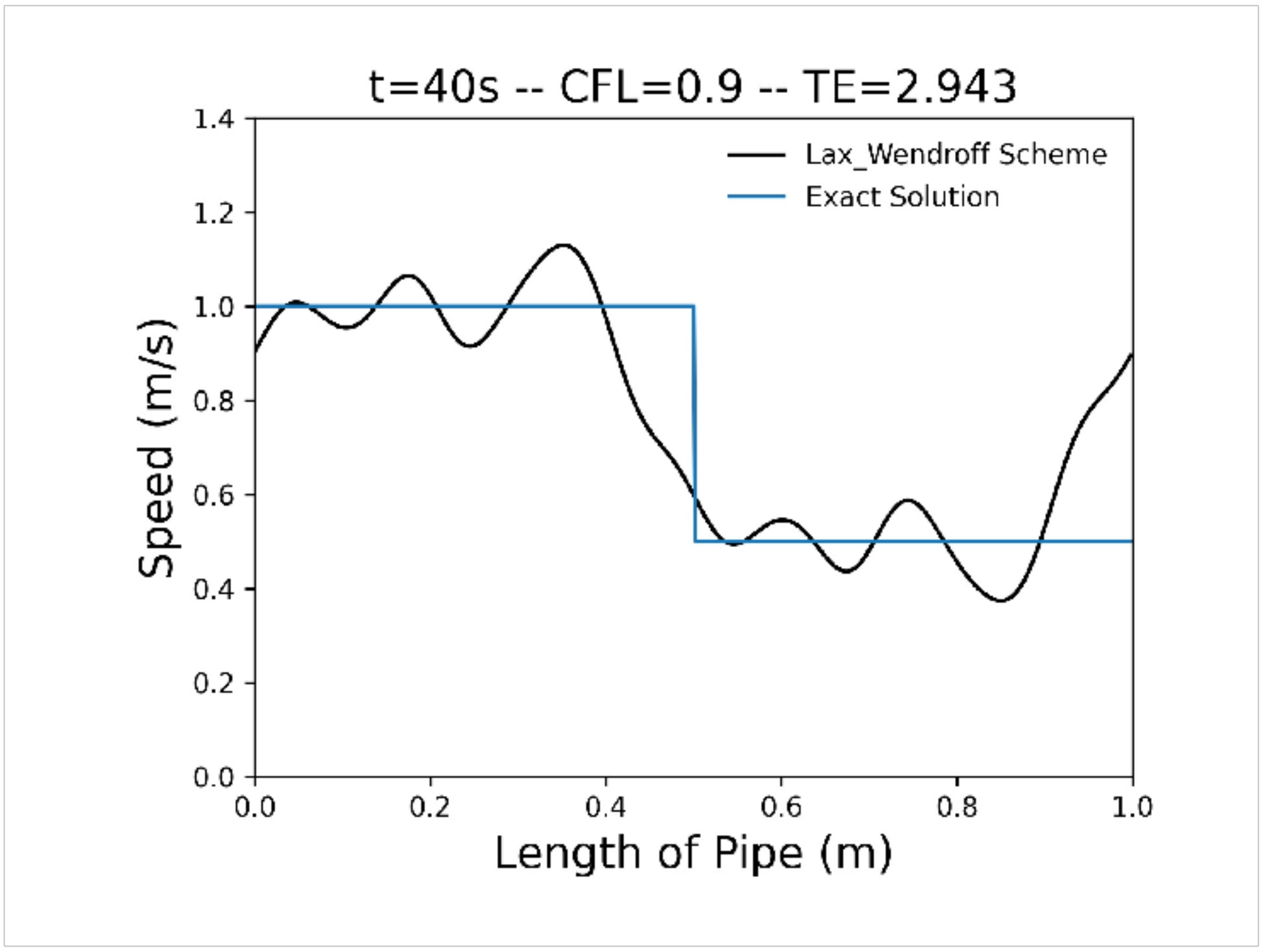

Figure 19: Total simulation time 40s and CFL 0.9

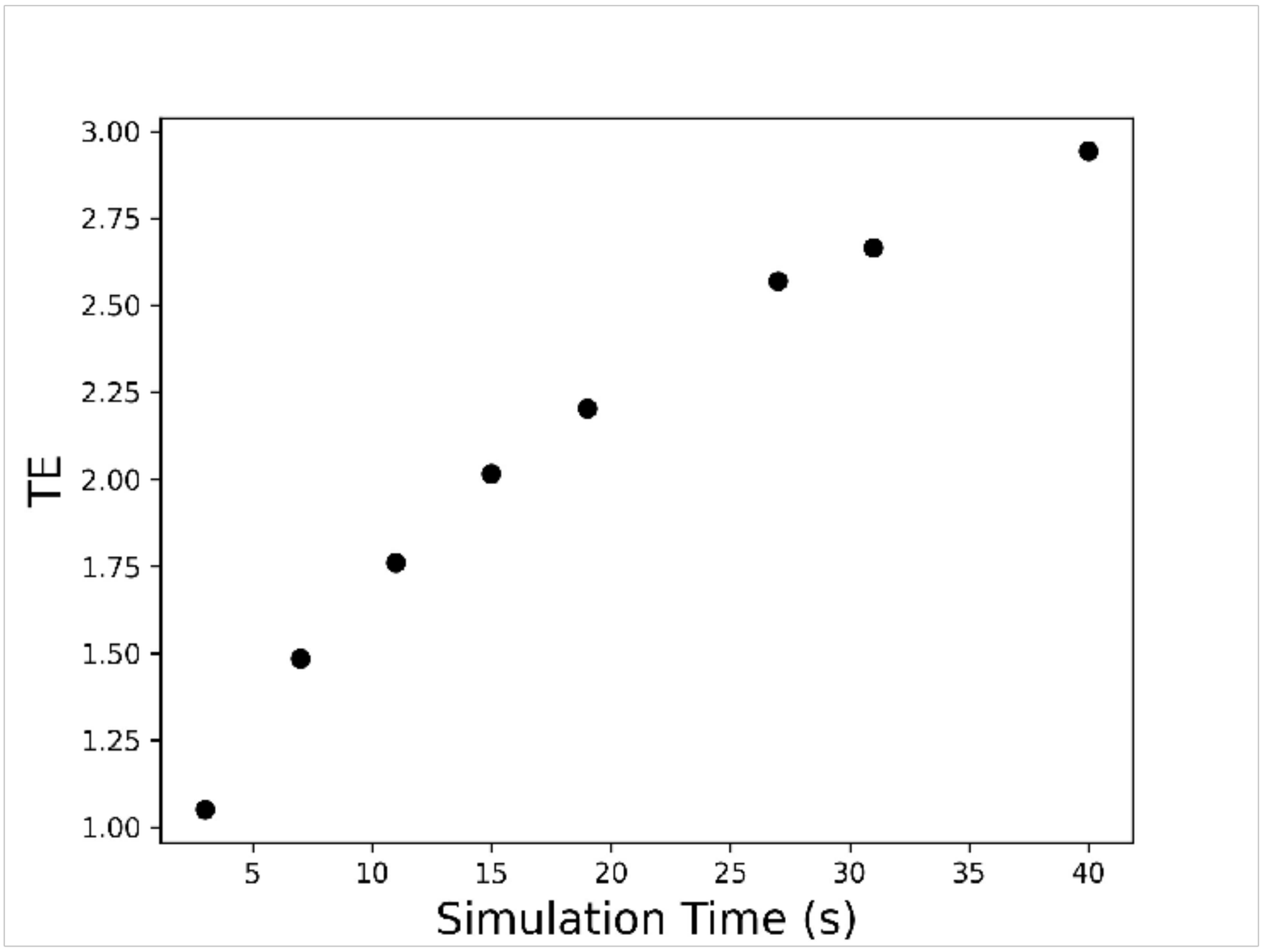

Figure 20: TE vs Simulation Time

4 Conclusion

For my project, I developed and used the Lax Wendroff scheme to solve the linear advection equation over a pipe with gas flowing through it. This gas has a discontinuity of velocity in the middle of the pipe. On the left side of the pipe, the gas flows at 1 m/s while on the right side of the pipe, after the discontinuity, the gas flows at 0.5 m/s. I coded the scheme using python and ran two parametric studies. The first one was a CFL number parametric study where I had to change the CFL number but keep all the other parameters the same. I observed that the scheme diverged from the exact solution as the CFL number decreased. Another parametric study I ran was a simulation time study, where I changed the total time of the code but kept all the parameters the same. I observed that the code diverges from the solution as the code ran longer.

I also appended my python code at the bottom of the report. Thank you for taking the time and reading through my report. This was a little hard for me to do as I struggled in the course and had my own research projects.

5 Python Code

```
from matplotlib import rc
from matplotlib import pyplot as plt
import numpy as np
import math as mt
import os
nx = 101;
xm = 1.;
xmm = 0.;
delta_x = (xm-xmm)/(nx-1)
dt = 0.0018
c = 0.5
cfl = c*dt/delta_x
x = np.arange(xmm,xm,delta_x)
u = np.zeros_like(x)
for i in range(0,len(u)):
if i <= (len(u)) / 2:
    u[i] = 1
else:
    u[i] = 0.5
uex = u
tf = 1
Nt = int(tf/dt)
t = np.linspace(0.,tf,Nt+1)
for n in range (1,len(t)):
    theta = (c*dt/delta_x)**2;
a = u
Um = np.roll(a,1)
gb = np.roll(a,-1)
u  = a - 0.5*cfl*(gb-Um) + 0.5*theta*(gb-2*a+Um)
d = c*n*dt
eb = u - uex
ec = np.linalg.norm(eb)
if (n==1): plt.figure()
plt.clf()
plt.plot(x,u,'k',label='Lax_Wendroff Scheme')
```

```
    plt.plot(x,uex,label='Exact Solution')
    plt.legend(frameon=False)
    plt.axis([xmm, xm, 0, 1.4])
    title="t="+str(tf)+"s -- CFL="+str(cfl)+" -- TE="+str(round(ec,3))
    plt.title('t='+str(tf),fontsize=12)
    plt.xlabel('Length of Pipe (m)',fontsize=12)
    plt.ylabel('Velocity (m/s)',fontsize=12)
plt.savefig("solution.jpg", dpi=300)
```