Computational Thermo-Fluid Dynamics
Winter Semester 2020-2021

2D wave propagation in different irregular geometries

Sebastian Egger[1]*, Adrian Salah[2]

Abstract
In this work the authors investigated the transmission of waves through different geometries using the wave equation. The geometries of interest are three variations of the popular 'Backstep' geometry that is often used in computational fluid dynamics to visualize turbulence phenomenon. The variations differ in the transition of the inlet to the outlet. Goal of this work is to find out how the difference in the transition affects the wave propagation. The input signal is a random noise signal generated at the inlet. The observation of the experiments showed that a sharp corner results in more waves being reflected while a smooth transition allows for a better wave transmission. The authors also used a spectral analysis of the wave amplitudes at specific probe positions in the domain to get insight over the resonance behaviour of the geometry.

Keywords
wave equation — finite volume method — wave propagation

[1] *Master Student, TUM, Munich, Germany*
[2] *Master Student, TUM, Munich, Germany*
***e-mails**: sebastian.egger@tum.de, adrian.salah@tum.de

Contents

1. Introduction

The studied phenomenon of interest is a wave propagating through different cone shaped geometries displayed in figure 1.

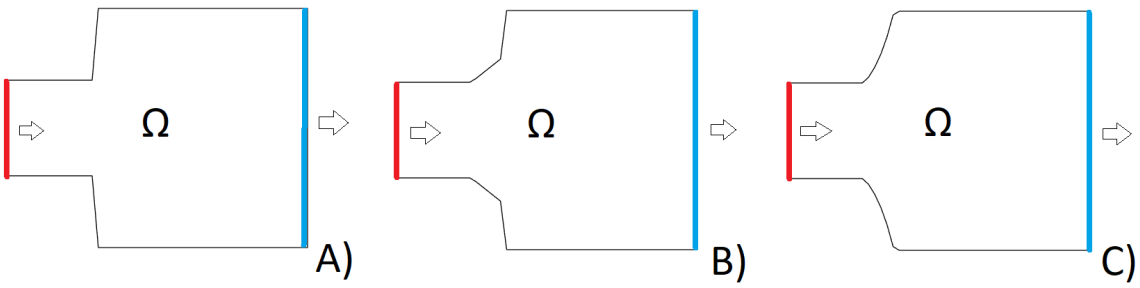


Figure 1. Outline of the studied geometries. The inlet is marked in red and the outlet in blue colour. The geometries are referred to as A) Backstep, B) angled Backstep and C) smooth Outlet

A white noise signal is fed into the inlet. It shall be investigated which frequency fractions are propagated through the inlet tube into the higher volume outlet chamber and which ones are reflected . Moreover it is examined if resonance is emerging and how those phenomena depend on the chosen geometry setup.

As a real world example one could imagine a wind instrument. Pressure differences created at the mouthpiece induce a longitudinal wave propagating through the instrument. The geometry of the instrument influences the created sound and therefore determines the frequencies leaving the outlet.
For this task a FVM solver for the two dimensional wave equation was developed. The wave equation is important for various branches of science and engineering such as fluid dynamics, acoustics, electro-magnetics or geophysics to name just a few. For this problem we consider the homogeneous wave equation. It is a second order partial differential equation (PDE) of the form:

$$\frac{\partial^2 u(\mathbf{x},t)}{\partial t^2} - c^2\nabla^2 u(\mathbf{x},t) = 0 \tag{1}$$

Therefore we have no source (or sink) terms in our simulation domain Ω. The scalar quantity u can generally represent different physical quantities. We consider u to be a scalar pressure field $p(\mathbf{x},t)$. In equation 1, c corresponds to the propagation speed or *phase velocity* of the wave. Evidently the phase velocity depends on the medium and its state through which the wave is propagating. From the general form of a linear second ordner PDE of two independent variables (x,t) [1]

$$A\frac{\partial^2\phi(x,t)}{\partial t^2} + B\frac{\partial^2\phi(x,t)}{\partial t\partial x} + C\frac{\partial^2\phi(x,t)}{\partial t^2} + ... = 0 \tag{2}$$

we can easily see that the expression $B^2 - 4AC$ is always bigger than zero as $B = 0$ and $C = -c^2 < 0$ in case of the wave equation. This argument can be extended to higher spatial dimensions $\mathbf{x} = (x, y, ...)$ as well. Thus, the wave equation is a *hyperbolic* PDE. We want to restrict the analysis to the two-dimensional case. The wave equation consequently takes the from $p_{tt} - c^2(p_{xx} + p_{yy}) = 0$.

2. Discretization

In order to solve this PDE on a computer, the computational domain needs to be discretized. The discretization needs to be in space as well as in time.

2.1 Spatial discretization

To discretize the spatial domain, the finite volume method was used. The domain is divided into cells with nodes at the corners. The PDE is integrated over each cell and using greens theorem we can find relations between each cell and neighbouring cells. This results in the discretized operator for the spatial discretization $Au_n = b$, which is a linear system of equations we can solve on the computer.

We used different geometries to solve the wave equation and compare the results. There are multiple ways of solving the PDE on a given geometry using a structured mesh. The

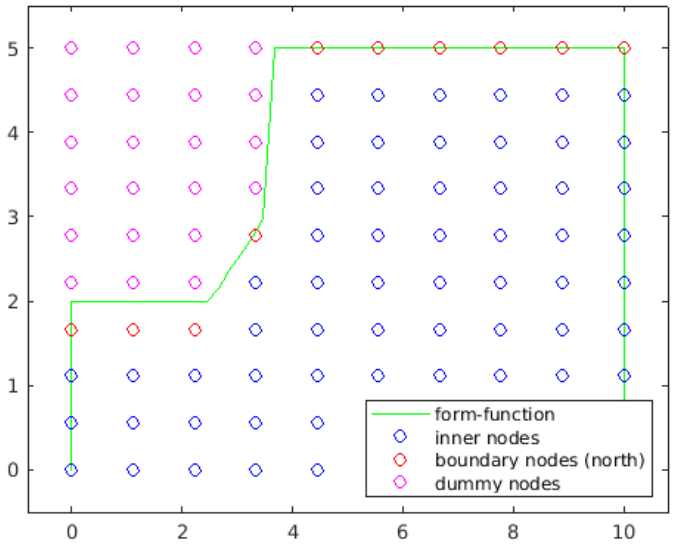


Figure 2. This figure shows the form-function of the given geometry and the nodes of the mesh

first option is to use one mesh which is non-uniform over the domain.
The second option is to use two meshes (two separate domains) and connect them via the boundary conditions (outlet values of one domain are used as inlet values for the second domain) and if necessary the values are interpolated.
The third option is to use a rectangular domain as big as necessary to cover the given geometry and then only use the nodes that lie within the geometry for computation.

In our case we chose the third option for our mesh. In order to realize this approach the dummy nodes (nodes that are not used in the computation) as well as the boundary nodes have to be determined. In our case only the boundary nodes on the north boundary had to be determined as the other boundary nodes are just at the boundary of the rectangular domain.

Figure 2 shows our mesh and the different nodes in different colours. The domain is a rectangular domain with a Cartesian mesh. The green line indicates the form-function which shows the given geometry. The purple nodes show the dummy nodes, which are not used in the computation. The red nodes are the nodes that are assigned to the north boundary and the blue nodes are the inner nodes (and also nodes belonging to other boundaries than the north boundary).

The advantage of our approach is that we can avoid stability issues that are introduced with cells that have different sizes as all of our cells have the same size but we do not use all of the cells for the computation.

2.2 Time discretization

The wave equation is an unsteady problem which means we have an evolution in time. For the computation we use a temporal domain $t \in [0, t_{end}]$ with equidistant time-step sizes $\Delta t = t_1 - t_0 = t_n - t_{n-1}$. The discretization of the PDE in time is handled via finite differences. For the wave equation we have a second-order time derivative, which allows the use of the centered difference approximation:

$$\frac{\partial^2}{\partial t^2} u \approx \frac{u_{n-1} - 2u_n + u_{n+1}}{\Delta t^2} \tag{3}$$

The centered difference scheme is second order accurate in time so $\mathcal{O}(\Delta t^2)$. The update rule to compute the solution of the wave equation for each time step can be written as follows,

$$u_{n+1} \approx 2u_n - u_{n-1} + c^2 \Delta t^2 (Au_n - b) \tag{4}$$

with $Au_n - b$ being the discretized version of the Laplacian operator.

One might recognize that for the solution of the next time step, the solution of the current as well as the past time-step are needed. Then the question arises how we can compute the first time step with only the current (zeroth) time-step. As this is not possible we find that we need not only an initial condition for the value u_n at time $t = 0$, but also for the derivative u'_n at time $t = 0$. Using the initial condition $u'_n = 0$ and a forward difference scheme for the approximation of the derivative we get the following expression:

$$\frac{u_0 - u_{-1}}{\Delta t} = 0 \tag{5}$$

with the timestep size Δt. This results in the following condition $u_0 = u_{-1}$. Using this discretization scheme we only get first order accuracy in time $\mathcal{O}(\Delta t)$. As we mix a second order discretization scheme with a first order discretization scheme we can only expect the solution to be first order accurate in time.

2.3 Numerical stability

The numerical stability of this method shall be investigated in the following. As our computational domain Ω is a rectangle with fixed cell-lengths Δx and Δy we can derive a stability criterion by means of the von Neumann stability analysis. The general solution of the wave equation 1 can be obtained by adding complex Fourier components

$$u(\mathbf{x}, t) = \sum_{n=0}^{\infty} a_n e^{i(k_x \Delta x + k_y \Delta y - \omega \Delta t)} \tag{6}$$

We use uniform timesteps: $t = n\Delta t$ for $n = \{0,1,2,3,...\}$ as well as equidistant grid points $x = l\Delta x$ and $y = m\Delta y$ with $m = \{0,1,2,3,...\}$ and $l = \{0,1,2,3,...\}$.
Therefore we can express the discretized solution as:

$$u_{l,m}^{n} = \sum_{n=0}^{\infty} a_n e^{ik_x l\Delta x} e^{ik_y m\Delta y} e^{-i\omega n\Delta t} \tag{7}$$

The discretized wave equation can now be written as:

$$D_t^2 u_{l,m}^{n} = c^2 [D_x^2 + D_y^2] u_{l,m}^{n} \tag{8}$$

The error between the exact solution at the grid points $u(x_l, y_m, t_n)$ and the discretized solution $u_{l,m}^{n}$ is defined as $\epsilon_{l,m}^{n} = u_{l,m}^{n} - u(x_l, y_m, t_n)$. As $u_{l,m}^{n}$ and $u(x_l, y_m, t_n)$ satisfy the discretized wave equation exactly, the error also satisfies equation 8. For stability it is required that the error amplification factor:

$$G := \frac{\epsilon_{l,m}^{n+1}}{\epsilon_{l,m}^{n}} \overset{!}{<} 1 \tag{9}$$

That essentially means that the error terms do not grow with time. Applying the discrete operators D_t^2, D_x^2 and D_y^2 on one single component of $u_{l,m}^{n}$ we obtain after some calculation:

$$sin^2\left(\frac{\omega\Delta t}{2}\right) = \left(\frac{c\Delta t}{\Delta x}\right)^2 sin^2\left(\frac{k_x\Delta x}{2}\right) + \left(\frac{c\Delta t}{\Delta x}\right)^2 sin^2\left(\frac{k_y\Delta y}{2}\right) \tag{10}$$

To full-fill the stability criterion in Eq. 9 we need to satisfy following relation:

$$\Delta t < \frac{1}{c}\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right)^{-1/2} \tag{11}$$

This condition is checked in the code for the specified parameters after the initialisation in the file `InitFVM.m` by the function `isWaveStable()`. If the condition 11 evaluates to `False` an error is thrown. A message that the CFL condition is not met is displayed to the user. This ensures robustness of the provided code.

In figure 3 we provide an example of a developing instability of the solution after a certain time-step. Note that the phase velocity c also influences the stability of the algorithm. The analysis was restricted for a constant phase velocity $c = const$. However it can be in general be extended to variable phase velocities $c(\mathbf{x})$. For non-constant phase

velocities the worst-case absolute velocity $\tilde{c}$ value has to be considered.

$$\tilde{c} = \sqrt{\max_{\mathbf{x}\in\Omega}(c^2(\mathbf{x}))}$$

For this case an additional safety factor $\beta < 1$ is often used. This was not done in the developed code as the phase velocity c is fixed. This loosens additional restrictions on possible parameter choices. The presented stability analysis can be conveniently extended to three dimensions.

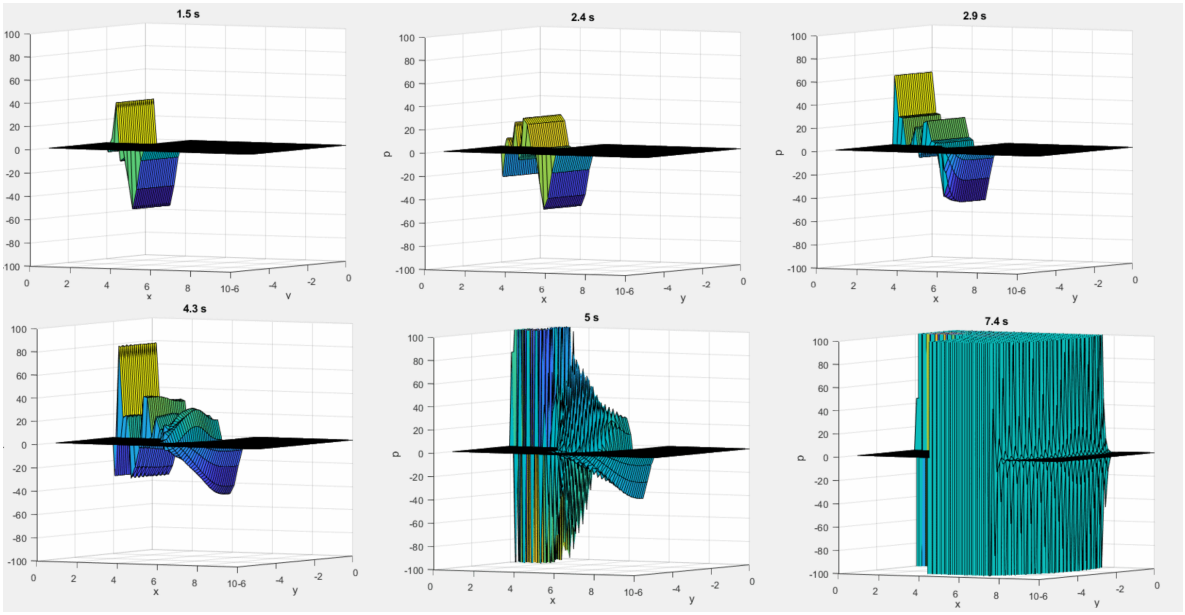


Figure 3. Development of a numerical instability for parameter choice $\Delta x = 0.2$, $\Delta y = 0.1$, $c = 1$ and $dt = 0.1$. The stability criterion in equation 11 is not fullfilled. Therefore the solvercode would throw an error in this case

3. Boundary Conditions

The process of solving a given PDE has to be handled differently at the boundary than at the inner nodes of the computational domain. To specify the solution behaviour at the boundary we use boundary conditions.

There are three different types of boundary conditions, homogeneous Dirichlet, homogenous Neumann and Robin boundary conditions. Dirichlet BCs use a fixed value of the solution e.g. $u = 0$ on the boundary, Neumann BCs use a fixed gradient of the solution e.g. $\frac{\partial}{\partial x}u = 0$ and Robin BCs are a combination of Dirichlet and Neumann BCs.

3.1 Physical meaning of boundary conditions

Regarding the wave equation there are different physical phenomenons related to the different types of boundary conditions. The homogeneous Dirichlet boundary condition

applied on a boundary, will result in the wave being reflected off the boundary but with the opposite sign and a fixed amplitude value of zero on the boundary. The homogeneous Neumann boundary condition applied on a boundary will result in the wave being reflected off the wall with the same sign. One special Robin boundary condition is the so called open boundary condition. It allows a wave to exit the computational domain without any other interaction. Depending on the direction of the wave the open boundary condition can be formulated as follows:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \tag{12}$$

It can be shown that the above equation accepts a solution $u = g_R(x - ct)$ (right-going wave), but not $u = g_L(x + ct)$ (left-going wave). This means the above equation will only allow right-going waves to pass through the boundary undisturbed.

3.2 Boundary Conditions used in our case

With the physical meaning of the boundary conditions explained we can discuss the boundary conditions used in our case setup. We used the west boundary as our inlet and the east boundary as our outlet. Because we exploited symmetry to save computational costs we use the south boundary as our symmetry boundary. The north boundary governs the shape of our geometry.

To simulate an incoming signal at the inlet, we use a time-dependent Dirichlet boundary condition on the west boundary. To make the signal travel through the geometry and out into the open without reflecting into the domain, we used the open boundary condition on the east boundary. For the symmetry on the south boundary we used a homogeneous Neumann boundary condition. We used the same boundary condition for the north boundary but instead of simulating a symmetry boundary we simulate a reflecting wall. Figure 4 gives a visual overview over the boundary conditions that are used in our case.

4. Implementation

In this section the essential components of the implemented code and solver steps are outlined. In the script file `InitFVM.m` all relevant physical parameters are defined. This includes mesh dimensions, time-step size, simulation time and choosen geometry among other parameters. It can be also specified how long the time-depended (white noise) input signal is present during the simulation time.

A setup file is stored to the project directory to store all the relevant parameters of the simulation. In the function `setupMesh()` the mesh nodes are initialized. Depending on the chosen geometry based on the form-function nodes outside of the simulation domain Ω are labeled as 'dummy nodes'.

The main solver loop consists of one loop to set up the coefficient matrix A of the spatial discretization according to the finite volume method as presented above. As A is a static

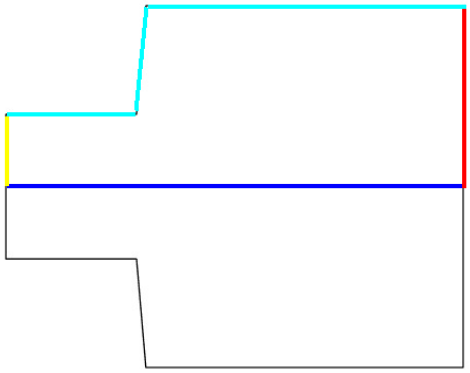
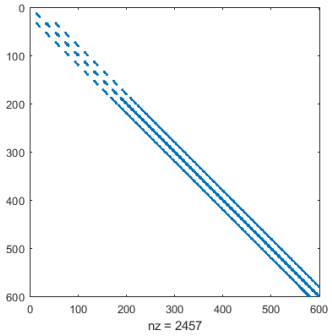
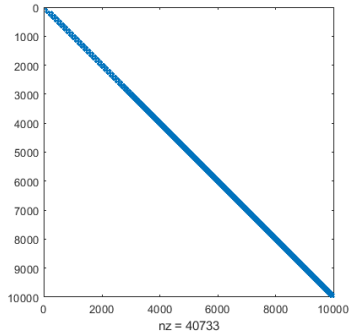


Figure 4. This figure shows the boundaries of the domain. **Blue:** Symmetry boundary with homogeneous Neumann BC. **Yellow:** Inlet boundary with time-dependent Dirichlet BC. **Red:** Outlet boundary with open-boundary condition. **Cyan:** North boundary specifying the shape of the geometry with homogeneous Neumann BC



(a) $dimX = 20,\ dimY = 30$



(b) $dimX = 100,\ dimY = 100$

Figure 5. Coefficient matrix A for different spatial resolutions

matrix it should be avoided to initialize A at every solution time-step for memory and performance reasons. For this matter the `stamp()` function, which previously calculated A and right hand side vector b was split up in two separate functions.

In line 1-8 the coefficient matrix is set up. If the current mesh node is a 'dummy node' as defined in the function `setupMesh()` it results in a zero line in the coefficient matrix A. Fig. 5 displays the structure of A for a different number of mesh nodes. The highly sparse tridiagonal structure of A suggests storing it in a sparse format.

In the next loop the right hand side vector is calculated according to the defined boundary conditions. If the current mesh node is a 'dummy node' as defined in the function `setupMesh()` it is ignored for the calculation. For the east side open boundary condition the first time derivative has to be calculated and passed to the `rhs()` function. The `rhs()` function is responsible for calculating the right hand side vector of the system $Ax = b$ based

on the imposed (and now time depended) boundary conditions. Likewise the current time-step has to be passed to `rhs()` to take account for the time-depended Diriechlet Boundary condition on the west-side. In the following the main solution step is executed and the current solution is stored in the solution array. Finally the variables and boundaries of the solution are updated.

```
% set up coefficient matrix A
for i=1:dimX
    for j=1:dimY
        if dummyNodes(index(i,j))==1
            continue;
        else
            A(index(i,j)) = stamp();
        end
end
% calculate right hand side
for timestep = 1:tend/dt
     for i = 1:dimX
         for j=1:dimY
             if dummyNodes(index(i,j))==1
                continue;
             else
                dp_dt = (p_current(i,j) - p_past(i,j))/dt;
                B(i,j) = rhs(..., timestep, dp_dt);
             end
      end
end
% main solution step
p_future(:) = c^2*dt^2*(A*p_current(:) - B(:)) ...
 + 2*p_current(:) - p_past(:);
% store solution of current timestep in solution array
p_solution(:,:,timestep) = p_current;
% update variables
p_past = p_current;
p_current = p_future;
p_current = updateBC(...,p_current,timestep);
end
```

Listing 1. Pseudo-Matlab code for the main solver loop

After the a completed simulation run a result file is written to save the simulation data. This file can later be re-loaded for post-processing. In `mainFVM.m` a basic statistical and

spectral analysis can be performed. Transient visualisations can be created and exported as GIF-files.

5. Results and Discussion

5.1 Visual Analysis

For a qualitative visual comparison the transient solution is displayed in Fig. 6-8 for different time-steps for the respective geometry. From the random white noise signal modulated at the west boundary, one can observe the formation a wave of a certain frequency. This generated wave is propagated till the opening to the outlet chamber.

One can clearly observe that there is a reflection occurring on the transition between the inlet pipe and the outlet chamber. Several fractions of the reflected wave get reflected back into the inlet tube and some get transported into the outlet chamber. This reflections lead to interference with the wave travelling in horizontal direction to the outlet. The geometric shape of the transition element between inlet and outlet chamber influences the direction of propagation of the reflected wave.

Furthermore, the expanding wave in the outlet chamber gets reflected back on the north and south boundary. The superposition of the reflected components with the other waves forms a complex pattern in the outlet chamber. It was also observed that no waves are propagating backwards to the left in horizontal direction. Therefore the implemented open boundary condition ensures that waves leave the simulation domain Ω.

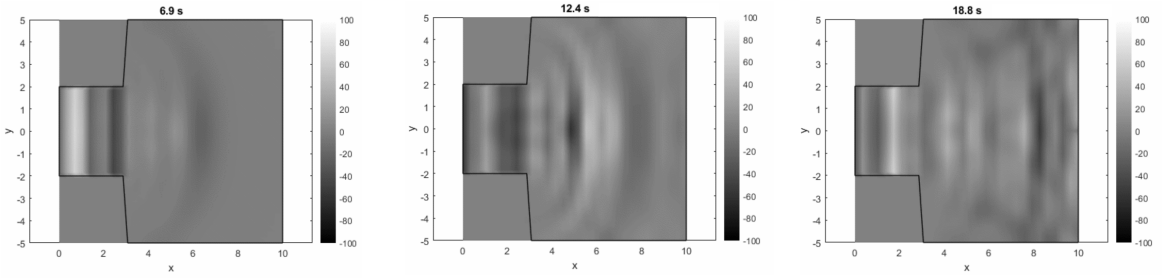


Figure 6. Time evolution of the propagating wave in the ‘Backstep’ geometry

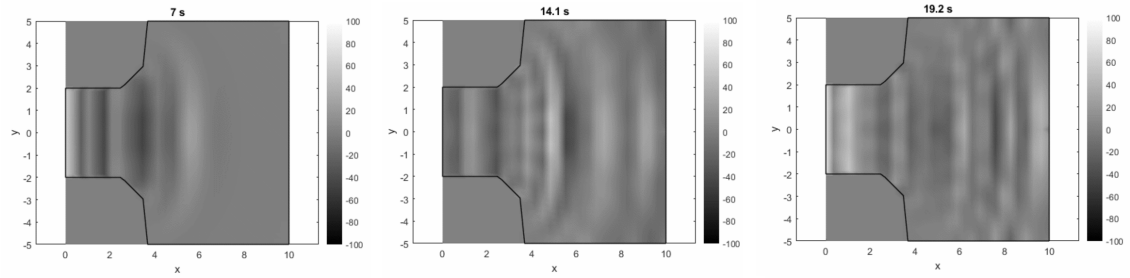


Figure 7. Time evolution of the propagating wave in the ‘angled Backstep’ geometry

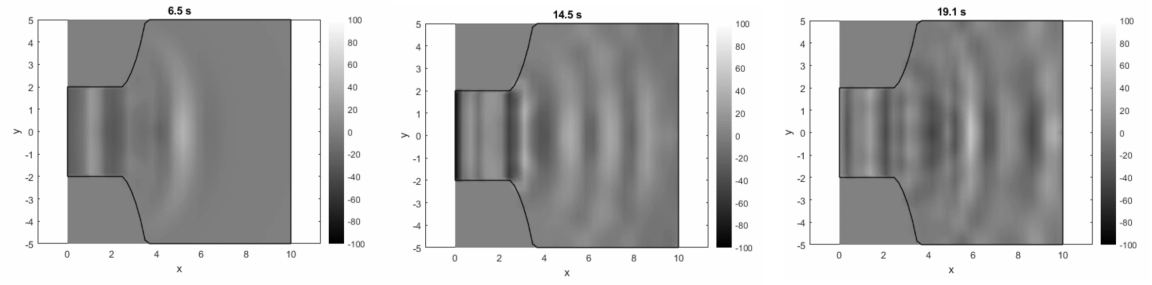


Figure 8. Time evolution of the propagating wave in the 'smooth Outlet' geometry

The time needed for the signal to reach the east (open) boundary as well as the the time needed for the signal to reach the higher volume outlet chamber is given in Tab. 1 and 2.

Table 1. Time needed to reach outlet

Parameter choice: $dt = 0.1$, $tend = 20$, $c = 1$	
Geometry	time in $[s]$
Backstep	6.5
angled Backstep	6.5
Smooth Outlet	6.4

Table 2. Time needed to leave inlet tube

Parameter choice: $dt = 0.1$, $tend = 20$, $c = 1$	
Geometry	time in $[s]$
Backstep	1.0
angled Backstep	1.2
Smooth Outlet	1.2

5.2 Probe Analysis

Two probes where placed in the inlet tube and the outlet chamber to alow a quantitative analysis of the time evolution of the wave and its frequency spectrum. The probe positions are depicted in Fig. 9. The instationary pressure probes are analysed for the time interval $t \in [0,500]$. This rather long time intervall ensures that the wave can propagrate throughout the geometry and one takes account for the interference effects occuring due to the geometrical shape.

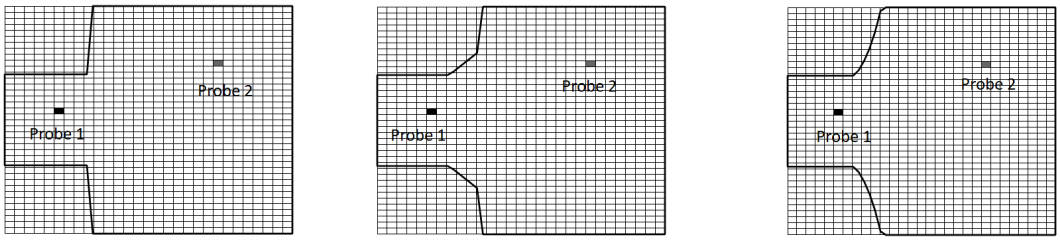
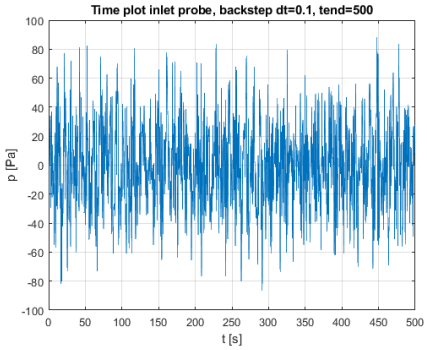


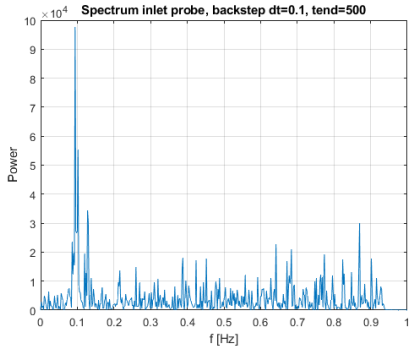
Figure 9. Probe positions for time and spectral analysis

In Fig. 10 one can see that a predominant frequency of approximately 0.1 Hz is monitored at the inlet probe. Due to the low angular frequency of the propagating wave the frequency range is depicted from $0-1$ Hz. The spectrum for the outlet probe in Fig. 11 shows that other frequencies parts are developed in the outlet chamber. By comparison of both time signals in Fig. 10 and 11 it is obvious that the amplitude is damped. This result is also reflected in the mean absolute amplitude values in Tab. 3 and 4.

For the 'angled Backstep' geometry multiple frequency components are found at the inlet probe. Likewise multiple frequency components are observed at the outlet chamber probe. However the spectra of inlet and outlet probe vary significantly. Therefore we conclude that for geometries 'Backstep' and 'angled Backstep' different frequencies are developed in the inlet tube as emitted through the outlet chamber. For the spectrum of the inlet probe of the 'smooth Outlet' Geometry shows again the predominant frequency component is again found at approx. 0.1 Hz. This frequency is also predominant in the spectrum at the outlet probe.

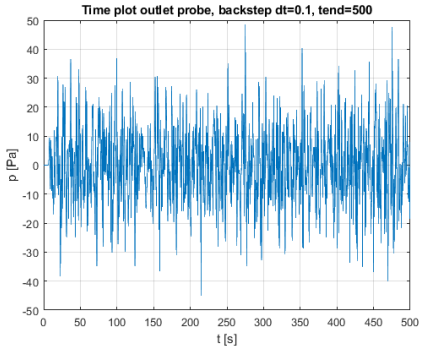


(a) Plot of the amplitude over time

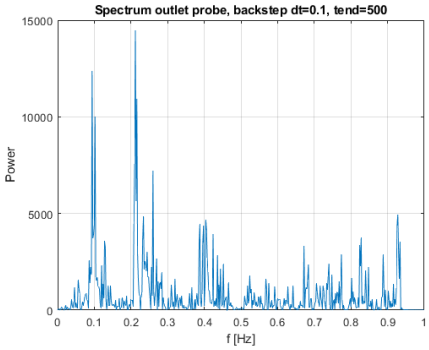


(b) Plot of the spectrum over the frequency

Figure 10. Spectral analysis of the inlet probe of the 'Backstep' geometry

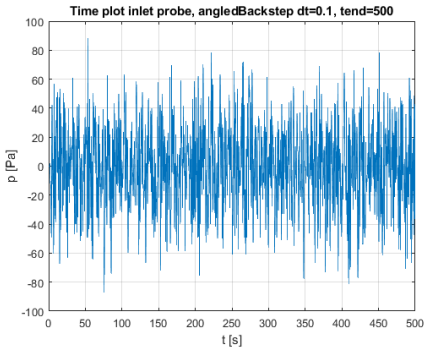


(a) Plot of the amplitude over time

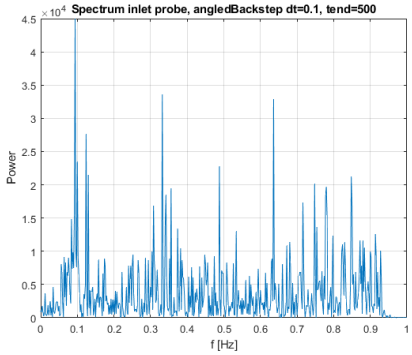


(b) Plot of the spectrum over the frequency

Figure 11. Spectral analysis of the outlet probe of the 'Backstep' geometry

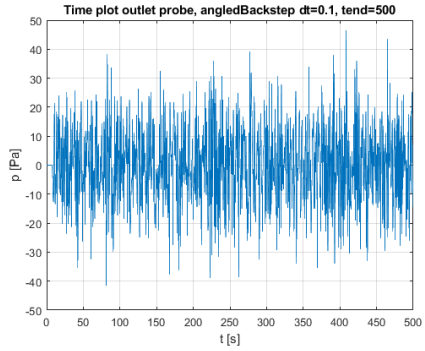


(a) Plot of the amplitude over time

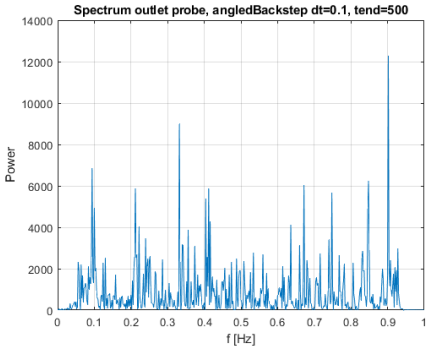


(b) Plot of the spectrum over the frequency

Figure 12. Spectral analysis of the inlet probe of the 'angled Backstep' geometry

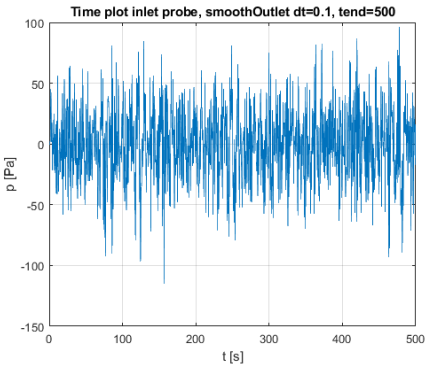


(a) Plot of the amplitude over time

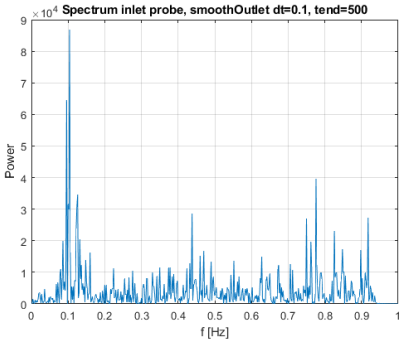


(b) Plot of the spectrum over the frequency

Figure 13. Spectral analysis of the outlet probe of the 'angled Backstep' geometry

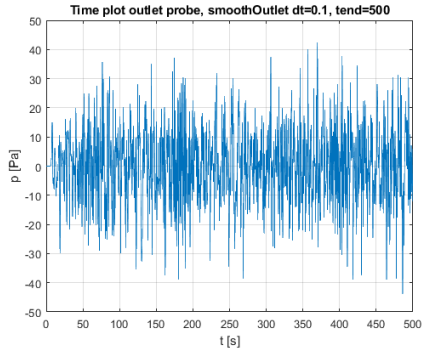


(a) Plot of the amplitude over time

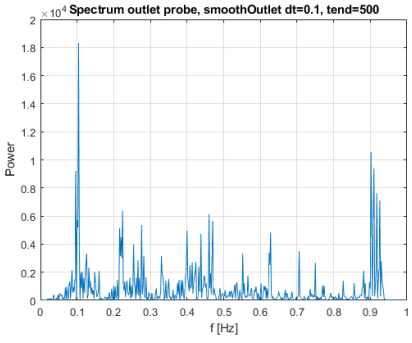


(b) Plot of the spectrum over the frequency

Figure 14. Spectral analysis of the inlet probe of the 'smooth Outlet' geometry



(a) Plot of the amplitude over time



(b) Plot of the spectrum over the frequency

Figure 15. Spectral analysis of the outlet probe of the 'smooth Outlet' geometry

Table 3. Maximum and mean absolute amplitudes for inlet probe

Inlet Probe		
Geometry	Mean value	Max value
Backstep	23.4877	102.0925
angled Backstep	21.1267	84.8887
Smooth Outlet	21.3841	96.4503

Table 4. Maximum and mean amplitudes for outlet probe

Probe Outlet		
Geometry	Mean value	Max value
Backstep	10.1596	45.1562
angled Backstep	9.8462	43.1932
Smooth Outlet	9.4434	59.2770

6. Conclusions

The overall solution of the 2d wave equation solver provides reasonable physical results, yet no concrete conclusions can be drawn regarding the physical reliability of the obtained results. As this study was performed without available experimental data, experimental evidence need to verify the observed behaviour in the studied geometries.

Moreover, a more in-depth quantitative analysis is needed to monitor the solution of the solver. One could introduce multiple probes in the domain and also study excitations of fixed frequencies at the west boundary and their effect on the observed spectra. The possibility of a single frequency excitation is already provided in the code.

To cover more realistic physical behaviour the code could be extended for variable phase velocities $c(\mathbf{x})$. This would account for a wave travelling in different media or a medium of differing spatial physical properties.

Acknowledgments

We thank Dr. Camilo Silva for his support and valuable input throughout the project as well as for the insights in the theoretical and numerical aspects of computational fluid dynamics.

References

[1] Camilo F. Silva, Ph. D., Ph. D. Kilian Förner, and Prof. Wolfgang Polifke, Ph. D. *Notes on Computational Thermo-Fluid Dynamics (Winter 2020)*. 2020.