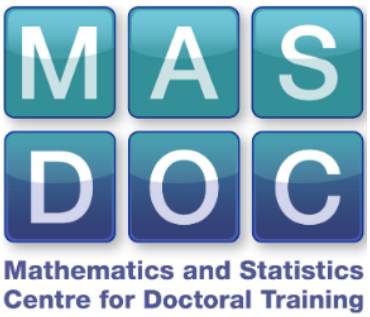
Transparent Boundary Conditions Applied to Atmospheric Problems

Lloyd Connellan

August 25, 2015

Submitted for the degree of Master of Science at the University of Warwick

Supervisor: Dr Andreas Dedner



Abstract

The compressible Euler equations are considered, and are linearised by assuming relatively small perturbations in the atmosphere. An equation for the pressure is derived, which is solved using Bessel functions, and this is used to derive a transparent boundary condition for the Euler equations. This BC is tested by numerical implementations in 1D and 2D, by moving the location of the top boundary and comparing the results. Finally we rerun the tests using assumptions for the background atmosphere taken from COSMO (a model used by weather services).

Contents

1 Introduction

A natural problem that arises when constructing any atmospheric model is the implementation of effective boundary conditions. Amongst the two kinds of domains commonly used in models, *global* ones that cover the entire Earth, and *regional* ones that cover a much smaller part, both feature a need for an upper boundary. Generally speaking this is because the atmosphere is very large, and the region we are interested in looking at is relatively small, so for the purposes of computations, it makes little sense to waste our efforts trying to simulate the whole thing. Consequently, we desire to truncate our *computational domain* Ω at a boundary, which requires us to set values at this boundary. The issue with using conventional boundary conditions in this case (such as Dirichlet or Neumann ones) is that for equations involving wave propagation (such as the Euler equations), these boundary conditions reflect waves back into the domain, interfering with our computations. Thus our aim becomes trying to derive boundary conditions for Ω that do not reflect waves, for which there has already been a good deal of work, see for example [4] and [10].

In particular one approach taken by Berenger [6] called a *Perfectly matched layer* (PML), which was originally used in computational electromagnetics, has been applied to the linearised Euler equations which we are interested in (see [2]). This method uses the concept of an additional damping layer around Ω, which is matched with the shape the waves take, in order to absorb them as they approach the boundary. However the downside of the PML method is that it is only reflectionless for the exact wave equations, and for real-life applications, the discretised problem will only be an approximation to the wave equation. In addition the direction of the waves must be accounted for, since this approach was originally designed for just 1D.

In this paper we will consider the concept of *transparent boundary conditions* (TBCs), whose name comes from the fact that they are designed to be *transparent* to outgoing perturbations, which give them the attribute of being non-reflecting. TBCs were originally designed for the Schrödinger equations (see [3]) but have also been applied by Sofronov to the magnetohydrodynamical (MHD) equations in [7] and the wave equation in [12]. The TBCs used in [7] are derived by analytically solving an equation for the pressure, which comes directly from the linearised Euler equations, after making some assumptions of the background atmosphere.

Our main focus we will be on taking the derivation of the TBCs from [7] and examining how they work in the context of a weather prediction model. In particular we will see whether we can modify the assumptions of the background atmosphere to more *realistic* ones. In section 2, we first introduce the linearised Euler equations and in section 3 we go through a derivation of the TBCs which follows [7] quite closely. Then in sections 4 and 5, we look at numerical implementations of this in 1D and 2D respectively.

2 Euler equations

In this section we begin by presenting the general problem we want to solve and the basic assumptions we make to simplify it. Here we will consider the Euler equations that are well-known for governing the dynamics of atmospheric processes. The Euler equations are essentially a simplification of the full Navier-Stokes equations made by assuming the fluid in question is inviscid (i.e. has no viscosity). Because of this reason, they primarily see use in situations where viscous effects are negligible, or in the development of algorithms that can eventually be extended to the Navier-Stokes equations [11].

The **compressible Euler equations** we use take the following form

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{1}$$

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot \rho \mathbf{u}\mathbf{u}^T + \nabla p = \rho \mathbf{g}, \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (\mathbf{u}(E+p)) = \rho \mathbf{g} \cdot \mathbf{u}. \tag{3}$$

where ρ is the **density**, $\mathbf{u} = (u, v, w)^T$ is the **velocity**, E is the **energy**, $\mathbf{g} = (0, 0, g)^T$ is the **gravitational force** and p is the **pressure**. We also use the equation of state

$$p = (\gamma - 1)(E - \frac{1}{2}\rho |\mathbf{u}|^2),$$

where γ is the **adiabatic exponent**. This allows us to write these equations in terms of the following prognostic variables: $\rho, \mathbf{u}$ and p.

For the **domain** for our equations, we will consider a 2-dimensional well in the x-z plane: $\Omega = [0, x_r] \times [0, \infty)$. The kind of domain we are trying to model is one similar to an atmosphere, i.e. with a solid bottom, that continues upwards until we decide to truncate it. In order to incorporate the upper boundary conditions, we separate the domain into our computational domain Ω_{comp} (the actual region of interest) and the rest of the domain Ω_{top} which continues upwards (this is where we will set boundary values). The top boundary is thus defined as the intersection of these two domains, which we will call Γ.

2.1 Linearisation

In ordinary circumstances there is no way to *solve* the Euler equations directly. Instead we will make a common assumption in atmospheric problems, by supposing that there is a stable background solution, and that perturbations to this solution are relatively small. To this end we assume that we can split our variables $\rho, \mathbf{u}$ and p into **background states** (which depend on z only) and **perturbations** (which are assumed to be small, so $O(2)$

terms are considered negligible). i.e. we use the following substitutions

$$\begin{aligned}\rho(t,x,z) &= \rho_0(z) + \tilde{\rho}(t,x,z),\\ \mathbf{u}(t,x,z) &= \mathbf{u}_0(z) + \tilde{\mathbf{u}}(t,x,z),\\ p(t,x,z) &= p_0(z) + \tilde{p}(t,x,z).\end{aligned}$$

We also assume that the energy can be split in the same way, i.e. $E = E_0 + \tilde{E}$, and thus the equation of state takes the form

$$p_0 = (\gamma - 1)E_0, \quad \tilde{p} = (\gamma - 1)\tilde{E}. \tag{4}$$

Additionally we make assumptions for the values of the background states (we will label these assumptions so we can refer to them later)

Assumption 1.

$$\frac{d}{dz}p_0 = g\rho_0, \tag{5}$$

$$\boldsymbol{u}_0 = 0. \tag{6}$$

Here equation (5) is the hydrostatic balance equation, and (6) sets the background velocity to zero.

Now we will put our split equations into (1) - (3), and as stated exclude the time derivatives of our background states, and second order perturbation terms. For (1) we have

$$\begin{aligned}&\frac{\partial(\rho_0+\tilde{\rho})}{\partial t} + \nabla\cdot\begin{pmatrix}(\rho_0+\tilde{\rho})\tilde{u}\\(\rho_0+\tilde{\rho})\tilde{w}\end{pmatrix} = 0,\\ \Rightarrow\quad &\frac{\partial\tilde{\rho}}{\partial t} + \rho_0\frac{\partial\tilde{u}}{\partial x} + \frac{\partial(\rho_0\tilde{w})}{\partial z} = 0.\end{aligned}$$

For (2) we have two equations, one from the x component, and one from the z component. Note that the term $\nabla\cdot\rho\mathbf{u}\mathbf{u}^T$ vanishes due to being of $O(2)$ for the perturbations. For the x component:

$$\begin{aligned}&\frac{\partial((\rho_0+\tilde{\rho})\tilde{u})}{\partial t} + \frac{\partial}{\partial x}(p_0+\tilde{p}) = 0,\\ \Rightarrow\quad &\rho_0\frac{\partial\tilde{u}}{\partial t} + \frac{\partial\tilde{p}}{\partial x} = 0.\end{aligned}$$

For the z component:

$$\begin{aligned}
&\frac{\partial((\rho_0+\tilde{\rho})\tilde{w})}{\partial t}+\frac{\partial}{\partial z}(p_0+\tilde{p})=g(\rho_0+\tilde{\rho}),\\
\Rightarrow\quad &\frac{\partial(\rho_0\tilde{w})}{\partial t}+\frac{\partial p_0}{\partial z}+\frac{\partial\tilde{p}}{\partial z}=g(\rho_0+\tilde{\rho}),\\
\Rightarrow\quad &\rho_0\frac{\partial\tilde{w}}{\partial t}+\frac{\partial\tilde{p}}{\partial z}-g\tilde{\rho}=0. \quad \text{(using (5))}
\end{aligned}$$

Finally for (3) we have

$$\begin{aligned}
&\frac{\partial(E_0+\tilde{E})}{\partial t}+\frac{\partial(\tilde{u}(E_0+\tilde{E}+p_0+\tilde{p}))}{\partial x}+\frac{\partial(\tilde{w}(E_0+\tilde{E}+p_0+\tilde{p}))}{\partial z}\\
&=g(\rho_0+\tilde{\rho})\tilde{w}\\
\Rightarrow\quad &\frac{\partial\tilde{E}}{\partial t}+\frac{\partial(\tilde{u}(E_0+p_0))}{\partial x}+\frac{\partial(\tilde{w}(E_0+p_0))}{\partial z}=g\rho_0\tilde{w}\\
\Rightarrow\quad &\frac{1}{\gamma-1}\frac{\partial\tilde{p}}{\partial t}+\left(\frac{1}{\gamma-1}+1\right)\left(\frac{\partial(\tilde{u}p_0)}{\partial x}+\frac{\partial(\tilde{w}p_0)}{\partial z}\right)=g\rho_0\tilde{w} \quad \text{(using (4))}\\
\Rightarrow\quad &\frac{\partial\tilde{p}}{\partial t}+\gamma\left(\frac{\partial(\tilde{u}p_0)}{\partial x}+\frac{\partial(\tilde{w}p_0)}{\partial z}\right)+(1-\gamma)g\rho_0\tilde{w}=0\\
\Rightarrow\quad &\frac{\partial\tilde{p}}{\partial t}+\gamma p_0\left(\frac{\partial\tilde{u}}{\partial x}+\frac{\partial\tilde{w}}{\partial z}\right)+g\rho_0\tilde{w}=0. \quad \text{(using (5))}
\end{aligned}$$

Thus putting these together, we have the following **linearised system** of equations

$$\frac{\partial\tilde{\rho}}{\partial t}+\rho_0\frac{\partial\tilde{u}}{\partial x}+\frac{\partial(\rho_0\tilde{w})}{\partial z}=0, \tag{7}$$

$$\rho_0\frac{\partial\tilde{u}}{\partial t}+\frac{\partial\tilde{p}}{\partial x}=0, \tag{8}$$

$$\rho_0\frac{\partial\tilde{w}}{\partial t}+\frac{\partial\tilde{p}}{\partial z}-g\tilde{\rho}=0, \tag{9}$$

$$\frac{\partial\tilde{p}}{\partial t}+\gamma p_0\left(\frac{\partial\tilde{u}}{\partial x}+\frac{\partial\tilde{w}}{\partial z}\right)+g\rho_0\tilde{w}=0. \tag{10}$$

Keeping in mind we want to eventually derive TBCs for the boundary Γ, we set the following **initial** and **boundary conditions** for the equations in Ω_{top} (i.e. $[0,x_r]\times[z_\Gamma,\infty)$)

$$\tilde{\rho},\tilde{u},\tilde{w},\tilde{p}|_{t=0} \tag{11}$$

$$\tilde{p}|_{z=\infty}=0 \tag{12}$$

in other words the perturbations are initially set to zero in the extended domain, and the pressure perturbation decays to zero over distance. In section 3.2 we will see that we can essentially interchange these initial and boundary conditions for Ω_{top} with ones at Γ instead.

3 Equation for the pressure perturbation

In section 2 we introduced the Euler equations, and we linearised them in 2.1 to put them into a form (7) - (10) which is more easily solvable. Now in order to derive our transparent boundary conditions for these equations, we want to be able to solve them analytically. Hence our next step is to reduce the linearised system of Euler equations into one single equation for the pressure perturbation that we can solve.

To do this we need to remove the other terms, the background states p_0 and ρ_0, and the perturbations $\tilde{u}, \tilde{w}$ and $\tilde{\rho}$, from the equation. To that end we must make an additional assumption of the background atmosphere.

Assumption 2.

$$p_0 = \rho_0^\gamma \tag{13}$$

This equation assumes that the atmosphere is **adiabatic**. It should be noted that this is a significant assumption to make, and in fact means that the gravitational force $\mathbf{g}$ is in general no longer constant, and instead becomes a function $\mathbf{g}(z)$[1].

From this equation we can see that

$$\begin{aligned} & & \log p_0 &= \gamma \log \rho_0 \\ \frac{\partial}{\partial z}: \quad & \Rightarrow & \frac{1}{p_0}\frac{dp_0}{dz} &= \frac{\gamma}{\rho_0}\frac{d\rho_0}{dz} \\ & \Rightarrow & g\rho_0 &= \frac{\gamma p_0}{\rho_0}\frac{d\rho_0}{dz} \quad \text{(using (5))} \end{aligned} \tag{14}$$

This expression will prove useful later on. Now first we would like to find a way to remove the $\tilde{\rho}$ terms from our equations. Note that we can rearrange (7)

$$\begin{aligned} & \frac{\partial \tilde{\rho}}{\partial t} + \rho_0 \frac{\partial \tilde{u}}{\partial z} + \rho_0 \frac{\partial \tilde{w}}{\partial z} + \tilde{w}\frac{d\rho_0}{dz} = 0, \\ \Rightarrow \quad & \frac{\partial \tilde{u}}{\partial z} + \frac{\partial \tilde{w}}{\partial z} = -\frac{1}{\rho_0}\left(\frac{\partial \tilde{\rho}}{\partial t} + \tilde{w}\frac{d\rho_0}{dz}\right). \end{aligned}$$

Inserting this into (10) we see that

$$\begin{aligned} & \frac{\partial \tilde{p}}{\partial t} - \frac{\gamma p_0}{\rho_0}\left(\frac{\partial \tilde{\rho}}{\partial t} + \tilde{w}\frac{d\rho_0}{dz}\right) + g\rho_0\tilde{w} = 0. \\ \Rightarrow \quad & \frac{\partial \tilde{p}}{\partial t} - \frac{\gamma p_0}{\rho_0}\frac{\partial \tilde{\rho}}{\partial t} + \tilde{w}\left(g\rho_0 - \frac{\gamma p_0}{\rho_0}\frac{d\rho_0}{dz}\right) = 0. \end{aligned}$$

[1] In section 3.1 we show that under this assumption, $g = \frac{\gamma}{\gamma-1}\frac{\partial}{\partial z}(\rho_0^{\gamma-1})$, and since ρ_0 depends on z, we clearly see that g is not constant in general.

Then using (14), this simplifies to

$$\frac{\partial \tilde{p}}{\partial t} - \frac{\gamma p_0}{\rho_0}\frac{\partial \tilde{\rho}}{\partial t} = 0.$$

Using the initial conditions in (11), we can conclude by integrating with respect to t that

$$\tilde{p} = \frac{\gamma p_0}{\rho_0}\tilde{\rho}. \tag{15}$$

Thus we now have a direct relation between $\tilde{p}$ and $\tilde{\rho}$. This will allow us to easily remove $\tilde{\rho}$ from our equations.

Next in order to remove the $\tilde{u}$ and $\tilde{w}$ terms we want to make use of the second and third Euler equations (8) and (9). We can do this by differentiating (10) with respect to t to get

$$\frac{\partial^2 \tilde{p}}{\partial t^2} + \gamma p_0\left(\frac{\partial^2 \tilde{u}}{\partial x \partial t} + \frac{\partial^2 \tilde{w}}{\partial z \partial t}\right) + g\rho_0\frac{\partial \tilde{w}}{\partial t} = 0. \tag{16}$$

We rearrange (8) as follows,

$$\frac{\partial \tilde{u}}{\partial t} = -\frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial x},$$

and rearrange (9),

$$\frac{\partial \tilde{w}}{\partial t} = \frac{g\tilde{\rho}}{\rho_0} - \frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial z},$$

so equation (16) becomes

$$\begin{aligned}
&\frac{\partial^2 \tilde{p}}{\partial t^2} + \gamma p_0\frac{\partial}{\partial x}\left(-\frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial x}\right) + \gamma p_0\frac{\partial}{\partial z}\left(\frac{g\tilde{\rho}}{\rho_0} - \frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial z}\right) + g\rho_0\left(\frac{g\tilde{\rho}}{\rho_0} - \frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial z}\right) = 0,\\
\Rightarrow\ &\frac{\partial^2 \tilde{p}}{\partial t^2} - \frac{\gamma p_0}{\rho_0}\frac{\partial}{\partial x}\left(\frac{\partial \tilde{p}}{\partial x}\right) + p_0\frac{\partial}{\partial z}\left(\frac{g\tilde{p}}{p_0}\right) - \gamma p_0\frac{\partial}{\partial z}\left(\frac{1}{\rho_0}\frac{\partial \tilde{p}}{\partial z}\right) + g^2\tilde{\rho} - g\frac{\partial \tilde{p}}{\partial z} = 0,\\
&\quad \text{(used (15) for third term)}\\
\Rightarrow\ &\frac{\partial^2 \tilde{p}}{\partial t^2} - \frac{\gamma p_0}{\rho_0}\left(\frac{\partial^2 \tilde{p}}{\partial x^2}\right) + p_0\left(\frac{\tilde{p}}{p_0}\frac{\partial g}{\partial z} - \frac{g^2\rho_0\tilde{p}}{p_0^2} + \frac{g}{p_0}\frac{\partial \tilde{p}}{\partial z}\right) - \gamma p_0\left(\frac{1}{\rho_0}\frac{\partial^2 \tilde{p}}{\partial z^2} - \frac{1}{\rho_0^2}\frac{\partial \rho_0}{\partial z}\frac{\partial \tilde{p}}{\partial z}\right)\\
&+g^2\tilde{\rho} - g\frac{\partial \tilde{p}}{\partial z} = 0, \quad \text{(used (5) for third term)}\\
\Rightarrow\ &\frac{\partial^2 \tilde{p}}{\partial t^2} - \frac{\gamma p_0}{\rho_0}\frac{\partial^2 \tilde{p}}{\partial x^2} + \tilde{p}\frac{\partial g}{\partial z} - \frac{g^2\rho_0\tilde{p}}{p_0} + g\frac{\partial \tilde{p}}{\partial z} - \frac{\gamma p_0}{\rho_0}\frac{\partial^2 \tilde{p}}{\partial z^2} + g\frac{\partial \tilde{p}}{\partial z} + g^2\tilde{\rho} - g\frac{\partial \tilde{p}}{\partial z} = 0.\\
&\quad \text{(used (14) for seventh term)}
\end{aligned}$$

So finally after using (15) on the second to last term, we get a single equation for the pressure perturbation:

$$\Rightarrow \frac{\partial^2 \tilde{p}}{\partial t^2} - \frac{\gamma p_0}{\rho_0}\left(\frac{\partial^2 \tilde{p}}{\partial x^2} + \frac{\partial^2 \tilde{p}}{\partial z^2}\right) + g\frac{\partial \tilde{p}}{\partial z} + \left(\frac{\partial g}{\partial z} + \frac{1-\gamma}{\gamma}\frac{g^2\rho_0}{p_0}\right)\tilde{p} = 0. \tag{17}$$

3.1 Solutions to pressure equation

We now have a PDE (17) for the pressure perturbation that we can solve under certain assumptions. In particular there are a series of steps we will go through to do this

1. We will make an additional assumption of the background atmosphere which relates p_0 and ρ_0 that we will use to remove p_0, ρ_0 and g from the equation to get (19)
2. We will apply the Fourier transform to remove the $\partial_x \tilde{p}$ term to get (20)
3. We will apply the Laplace transform to remove the $\partial_{tt}\tilde{p}$ term to get (21)
4. We will solve the resulting ODE using a modified Bessel function of the second kind to get (23)
5. We transform this into a differential equation for $\hat{p}$ and make some further simplifications to get (26)
6. Finally we will apply the inverse Laplace and Fourier transforms to return to an equation for $\tilde{p}$ to get (32)

Step 1: First we will make (17) an ODE w.r.t. $\hat{p}$ by making an assumption for the values of the background states ρ_0 and p_0. That is to say we will assume they obey the following exponential law

Assumption 3.

$$q(z) := \frac{\gamma}{\gamma - 1}\frac{p_0}{\rho_0} = \frac{1}{a}e^{-2bz},$$

where $a, b > 0$ are constants. It should be noted that this assumption is a purely mathematical one that we choose because it causes waves to dissipate as z goes to infinity. If possible we would prefer to use an assumption that accurately represents the behaviour of the atmosphere (we will do so in section 4.5).

Using this equation and our first assumption, equation (13), we note that $g = dq/dz := q'$, and furthermore

$$q\frac{d}{dz}\left(\frac{q'}{q}\right) = \frac{dg}{dz} + \frac{1-\gamma}{\gamma}\frac{g^2\rho_0}{p_0}.$$

This allows us to remove g and dg/dz from the equation as well. Therefore we rewrite our PDE (17) in terms of $\tilde{p}$ and q

$$\frac{\partial^2\tilde{p}}{\partial t^2} + (1-\gamma)q\left(\frac{\partial^2\tilde{p}}{\partial x^2} + \frac{\partial^2\tilde{p}}{\partial z^2}\right) + q'\frac{\partial\tilde{p}}{\partial z} + q\frac{d}{dz}\left(\frac{q'}{q}\right)\tilde{p} = 0. \tag{18}$$

We then make use of assumption 3 by substituting in $q(z) = a^{-1}e^{-2bz}$ to the above, noting that $d/dz(q'q^{-1}) = d/dz(-2b) = 0$, to obtain

$$ae^{2bz}\frac{\partial^2\tilde{p}}{\partial t^2} + (1-\gamma)\left(\frac{\partial^2\tilde{p}}{\partial x^2} + \frac{\partial^2\tilde{p}}{\partial z^2}\right) - 2b\frac{\partial\tilde{p}}{\partial z} = 0. \tag{19}$$

Step 2: Next we will apply the **Fourier** transform to (19) above to remove the x derivative, i.e. we will use the following transform

$$p^\lambda(t,z) = \frac{1}{x_r}\int_0^{x_r} e^{i\lambda x}\tilde{p}(t,x,z)dx, \quad \lambda \in 2\pi\mathbb{Z},$$

where $[0, x_r]$ is the x domain. This gives us the following equation w.r.t. p^λ

$$ae^{2bz}\frac{\partial^2 p^\lambda}{\partial t^2} + (1-\gamma)\left(\frac{\partial^2 p^\lambda}{\partial z^2} - \lambda^2 p^\lambda\right) - 2b\frac{\partial p^\lambda}{\partial z} = 0. \tag{20}$$

Step 3: Let us now introduce the **Laplace** transform to (20) in order to remove the time derivative. i.e. we use the following transform

$$\hat{p}(s,z) = \int_0^\infty e^{-st}p^\lambda(t,z)dt.$$

Let also $\theta = (\gamma-1)^{-1}$. This gives us the following ODE w.r.t. $\hat{p}$

$$\frac{\partial^2\hat{p}}{\partial z^2} + 2b\theta\frac{\partial\hat{p}}{\partial z} - (a\theta s^2 e^{2bz} + \lambda^2)\hat{p} = 0. \tag{21}$$

Step 4: We note that the above ODE can be reduced to a modified Bessel equation, for which the solutions are defined in terms of the modified Bessel functions. To see this, we use the following substitution

$$\hat{p} = e^{\delta z}f(y),$$
$$\delta = b\theta, \quad y = \alpha e^{bz}, \quad \alpha = \frac{s}{b}\sqrt{a\theta}.$$

This transforms (21) to the following

$$\begin{aligned}
&\frac{\partial^2}{\partial z^2}(e^{\delta z}f(\alpha e^{bz})) + 2b\theta\frac{\partial}{\partial z}(e^{\delta z}f(\alpha e^{bz})) - (a\theta s^2 e^{2bz} + \lambda^2)(e^{\delta z}f(\alpha e^{bz})) = 0,\\
\Rightarrow\quad & b^2\alpha^2 e^{(2b+\delta)z}\frac{d^2 f}{dz^2} + b^2\alpha e^{(b+\delta)z}\frac{df}{dz} - (\delta^2 + b^2\alpha^2 e^{2bz} + \lambda^2)e^{\delta z}f = 0,\\
\Rightarrow\quad & b^2y^2\frac{d^2f}{dz^2} + b^2y\frac{df}{dz} - (\delta^2 + \lambda^2 + b^2y^2)f = 0,\\
\Rightarrow\quad & y^2\frac{d^2f}{dz^2} + y\frac{df}{dz} - (\nu^2 + y^2)f = 0, \qquad \text{where } \nu^2 = \left(\frac{\lambda}{b}\right)^2 + \theta^2.
\end{aligned} \tag{22}$$

Due to the boundary conditions we set for $\tilde{p}$ in (12), i.e. that $\tilde{p}$ goes to zero as z goes to infinity, we can say that the solution to this equation is a modified Bessel function of the second kind, i.e. $f = K_\nu(y)$. Recalling our substitution for $\hat{p}$, we thus have that our general solution to our pressure perturbation equation (19) takes the form

$$\hat{p} = Ce^{-\theta bz} K_\nu(\alpha e^{bz}) \tag{23}$$

where C is a constant.

Step 5: Ideally we want to get rid of the constant in the above expression, so to proceed we will differentiate with respect to z and rewrite our solution as a differential equation for $\tilde{p}$. This gives us

$$\begin{aligned} & \frac{d}{dz}\hat{p} = -C\theta b e^{-\theta bz} K_\nu(\alpha e^{bz}) + Cb\alpha e^{(b-\theta b)z} K'_\nu(\alpha e^{bz}) \\ \Rightarrow \quad & \frac{d}{dz}\hat{p} = \left(-\theta b + b\alpha e^{bz} \frac{K'_\nu(\alpha e^{bz})}{K_\nu(\alpha e^{bz})}\right)\hat{p}. \end{aligned} \tag{24}$$

We can use the properties of the modified Bessel functions to simplify this equation. Namely we note that through the asymptotic properties of $K_\nu(\sigma)$ for large σ, we have that

$$\frac{K'_\nu(\sigma)}{K_\nu(\sigma)} = -1 - \frac{1}{2\sigma} + O\left(\frac{1}{\sigma^2}\right).$$

i.e.

$$\frac{K'_\nu(\sigma)}{K_\nu(\sigma)} + 1 + \frac{1}{2\sigma} = O\left(\frac{1}{\sigma^2}\right). \tag{25}$$

Suppose we take $\sigma(s,z) = \alpha e^{bz} = sb^{-1}\sqrt{a\theta}e^{bz}$. We can make use of the above expression by introducing the following function

$$\begin{aligned} \hat{A}(s,z) := \hat{A}(\sigma) &= \sigma\left(\frac{K'_\nu(\sigma)}{K_\nu(\sigma)} + 1 + \frac{1}{2\sigma}\right) \\ &= \sigma\frac{K'_\nu(\sigma)}{K_\nu(\sigma)} + \sigma + \frac{1}{2}. \end{aligned}$$

Then note that due to equation (25), $\hat{A}(s,z)$ decays to 0 as $s \to \infty$ as it is of order σ^{-2} (and hence of order s^{-2}). Now we rewrite (24) in terms of $\hat{A}(s,z)$ as follows

$$\frac{d}{dz}\hat{p} = \left(-\theta b - b\alpha e^{bz} - \frac{b}{2} + b\hat{A}(s,z)\right)\hat{p}. \tag{26}$$

So we have now written our equation for the pressure in a form where the only term which requires significant computations is $\hat{A}(s,z)$.

Looking to simplify this term further, we can in fact use the following identity

$$K'_\nu(\sigma) = -K_{\nu+1}(\sigma) + \frac{\nu K_\nu(\sigma)}{\sigma},$$

to rewrite $\hat{A}(s,z)$ as

$$\hat{A}(s,z) = \hat{A}(\sigma) = -\sigma\frac{K_{\nu+1}(\sigma)}{K_\nu(\sigma)} + \nu + \sigma + \frac{1}{2}. \tag{27}$$

Step 6: We now want to apply the inverse Laplace and Fourier transforms to equation (26) to recover an equation for $\tilde{p}$. Firstly we want to check that we can apply the inverse Laplace transform. It is clear that we can do this for the first three terms. For $\hat{A}(s,z)$, we must satisfy the following conditions

1. $\lim_{s\to\infty} \hat{A}(s,z) = 0$,
2. $\lim_{s\to\infty} s \cdot \hat{A}(s,z)$ is finite.

However due to the way we defined $\hat{A}(s,z)$, it is of order $1/\sigma^2$, so we automatically get these two conditions. Thus we can apply the inverse Laplace transform $\mathcal{L}$ to $\hat{A}(s,z)$ to get

$$\mathcal{L}^{-1}[\hat{A}(\sigma)](t\sigma^{-1}) = A^\lambda(t,z). \tag{28}$$

Then to apply the inverse Laplace transform to $\hat{A}(s,z)\hat{p}(s,z)$, we use the convolution theorem for the Laplace transform, i.e.

$$\begin{aligned} \mathcal{L}^{-1}(\hat{A}(s)\hat{p}(s)) &= \sigma^{-1}\int_0^t A^\lambda(t-\tau)p^\lambda(\tau)d\tau, & (29)\\ &:= \sigma^{-1}A^\lambda * p^\lambda. & (30) \end{aligned}$$

Thus we can calculate the inverse Laplace transform of (26) to get

$$\frac{\partial}{\partial z}p^\lambda + \sqrt{a\theta}e^{bz}\frac{\partial}{\partial t}p^\lambda + \left(\theta b + \frac{b}{2}\right)p^\lambda - \frac{b^2e^{-bz}}{\sqrt{a\theta}}A^\lambda * p^\lambda = 0.$$

Let $c = e^{bz}/\sqrt{a\theta}$[2]. Then we can rewrite the above as follows

$$\frac{\partial}{\partial t}p^\lambda + c\frac{\partial}{\partial z}p^\lambda + \left(\theta b + \frac{b}{2}\right)cp^\lambda - b^2c^2A^\lambda * p^\lambda = 0. \tag{31}$$

Finally we can apply the inverse Fourier transform Q^{-1} to get the below equation

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right)c\tilde{p} - b^2c^2Q^{-1}(A^\lambda *)Q\tilde{p} = 0. \tag{32}$$

[2]Note that c corresponds to $\sqrt{\gamma p_0/\rho_0}$ which is the equation for the speed of sound (in an ideal gas).

3.2 Boundary conditions for the top boundary

In section 2.1 we derived the linearised Euler equations (7) - (10), and in sections 3 and 3.1 we derived (32) by solving an equation for the pressure perturbation. The equation is

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right) c\tilde{p} - b^2c^2Q^{-1}(A^\lambda *)Q\tilde{p} = 0. \tag{32}$$

Here we will show that we can use this equation as a boundary condition for $\tilde{p}$ at the top boundary of our domain. To this end we pose the following theorem

Theorem 1. *Consider the equation for the pressure perturbation (19):*

$$ae^{2bz}\frac{\partial^2\tilde{p}}{\partial t^2} + (1-\gamma)\left(\frac{\partial^2\tilde{p}}{\partial x^2} + \frac{\partial^2\tilde{p}}{\partial z^2}\right) - 2b\frac{\partial\tilde{p}}{\partial z} = 0. \tag{19}$$

Recall that we defined the domains $\Omega_{top} = [0, x_r] \times [z_\Gamma, \infty)$, $\Omega_{comp} = [0, x_r] \times [0, z_\Gamma]$ *and the top boundary* $\Gamma = \Omega_{top} \cap \Omega_{comp}$. *Suppose we impose the conditions in* $\Omega = \Omega_{comp} \cup \Omega_{top}$

$$\begin{aligned} &\bullet \textit{ initial conditions: } \tilde{p}(x,z,0) = \begin{cases} 0 & \textit{in } \Omega_{top} \\ \tilde{p}_0 & \textit{in } \Omega_{comp}. \end{cases} \\ &\bullet \textit{ boundary conditions: } \begin{cases} \tilde{p} = 0 & \textit{at } z = 0 \\ \tilde{p} \to 0 & \textit{for } z \to \infty \end{cases} \end{aligned} \tag{33}$$

Then if we restrict our domain to Ω_{comp} *and use the alternate conditions*

$$\begin{aligned} &\bullet \textit{ initial conditions: } \tilde{p}(x,z,0) = \tilde{p}_0 \\ &\bullet \textit{ boundary conditions: } \begin{cases} \tilde{p} = 0 & \textit{at } z = 0 \\ (32) & \textit{on } \Gamma \end{cases} \end{aligned} \tag{34}$$

we have the following

1. *A solution to (19) using conditions (33) is also a solution to (19) using (34).*
2. *A solution to (19) using conditions (34) which is continuously differentiable up to* Γ *can be extended into* Ω_{top} *to obtain a solution using (33).*

Proof: 1. This has already been proved by virtue of the fact that the TBCs (32) are derived from a solution to (19).

2. Suppose we have a solution in Ω_{comp} satisfying (34). This means the pressure perturbation (under the Fourier and Laplace transforms) takes the form we derived using the

Bessel functions in section 3.1 in (23), i.e.

$$\hat{p}(s,z) = Ce^{-\theta bz} K_\nu(ce^{bz}) \tag{23}$$

where $c = \frac{s}{b}\sqrt{\frac{a}{\gamma-1}}$. This however only gives us the solution up to a constant C. We can fix C for instance by prescribing values for the solution on Γ.

Let us denote the evaluation of $\tilde{p}(t,z)$ at z_Γ as p_Γ. In order to use the above formula, we consider p_Γ with the Fourier and Laplace transforms applied to it, i.e. $\hat{p}_\Gamma$. Let us choose C in (23) such that $\hat{p}(s,z_\Gamma) = \hat{p}_\Gamma$. This gives us the following extension of $\hat{p}$ into Ω_{top}

$$\hat{p} = \hat{p}_\Gamma e^{-\theta b(z-z_\Gamma)} \frac{K_\nu(ce^{bz})}{K_\nu(ce^{bz_\Gamma})}, \quad z > z_\Gamma$$

We can see that by setting $z = z_\Gamma$, this returns us $\hat{p}_\Gamma$ as required. Thus the extension is continuous on Γ. Since $\tilde{p}$ is continuous in Ω_{comp} and by extension Ω_{top}, it is therefore continuous on the whole domain Ω.

Now to get back the original $\tilde{p}$, we need to apply the inverse Laplace and Fourier transforms. To see that the inverse Laplace transform is valid here, we must show that

1. $\lim_{s\to\infty} \hat{p}(s) = 0$,
2. $\lim_{s\to\infty} s \cdot \hat{p}(s)$ is finite.

We can see both of these since $K_\nu(Mx)/K_\nu(x) \sim e^{(1-M)x}$, and since $z > z_\Gamma$, we have $M > 1$. Therefore $\hat{p}$ decays exponentially to zero with respect to s. Thus we apply the inverse Laplace and Fourier transforms, which gives us a solution for $\tilde{p}$ in Ω_{top}.

Now we know that since $\tilde{p}$ is continuous w.r.t.z, the x and t derivatives must be continuous as well, i.e. $\partial_{tt}\tilde{p}$ and $\partial_{xx}\tilde{p}$ are continuous. It remains to show that the $\partial_z\tilde{p}$ and $\partial_{zz}\tilde{p}$ terms in (19) are continuous at the boundary Γ.

To see that $\partial_z\tilde{p}$ is continuous, we consider (32) on both sides of the boundary. Since $\tilde{p}$ and its x and t derivatives are continuous at the boundary, we see that all the terms not involving $\partial_z\tilde{p}$ can be moved to the other side. Thus we see that $\partial_z\tilde{p}$ must be continuous.

To see that $\partial_{zz}\tilde{p}$ is continuous, we notice the same thing as above for (19). Therefore due to our extension, we have obtained a solution to (19) that is valid for the conditions (33). □

Now that we have proved that a solution to the pressure equation is valid on a smaller domain, we can generalise this to the system of linearised Euler equations quite easily. To do so we pose the following corollary.

Corollary 1. *Consider the linearised Euler equations defined on* $\Omega = [0, x_r] \times (0, \infty)$

$$\begin{aligned} \frac{\partial \tilde{\rho}}{\partial t} + \frac{\partial(\rho_0 \tilde{w})}{\partial z} &= 0, \\ \rho_0 \frac{\partial \tilde{w}}{\partial t} + \frac{\partial \tilde{p}}{\partial z} - g\tilde{\rho} &= 0, \\ \frac{\partial \tilde{p}}{\partial t} + \gamma p_0 \frac{\partial \tilde{w}}{\partial z} + g\rho_0 \tilde{w} &= 0. \end{aligned}$$

Let $\tilde{\boldsymbol{v}} = (\tilde{\rho}, \tilde{u}, \tilde{w}, \tilde{p})$. *Suppose we impose the conditions*

$$\begin{aligned} &\bullet \textit{ initial conditions: } \tilde{\boldsymbol{v}}(x, z, 0) = \begin{cases} 0 & \textit{in } \Omega_{top} \\ \tilde{\boldsymbol{v}}_0 & \textit{in } \Omega_{comp} \end{cases} \\ &\bullet \textit{ boundary conditions: } \begin{cases} \tilde{\boldsymbol{v}} = 0 & \textit{at } z = 0 \\ \tilde{\boldsymbol{v}} \to 0 & \textit{for } z \to \infty \end{cases} \end{aligned} \tag{35}$$

Then if we restrict our domain to Ω_{comp} *and use the alternate conditions*

$$\begin{aligned} &\bullet \textit{ initial conditions: } \tilde{\boldsymbol{v}}(x, z, 0) = \tilde{\boldsymbol{v}}_0 \\ &\bullet \textit{ boundary conditions: } \begin{cases} \tilde{\boldsymbol{v}} = 0 & \textit{at } z = 0 \\ (32) & \textit{on } \Gamma \\ \tilde{\rho} = c^{-2}\tilde{p} & \textit{on } \Gamma \\ \partial_t \tilde{u} = -\rho_0^{-1} \partial_x \tilde{p} & \textit{on } \Gamma \\ \partial_t \tilde{w} = g\rho_0^{-1}\tilde{\rho} - \rho_0^{-1}\partial_z \tilde{p} & \textit{on } \Gamma \end{cases} \end{aligned} \tag{36}$$

we have that

1. *A solution to the Euler equations using conditions (35) is also a solution to the Euler equations using (36).*
2. *A solution to the Euler equations using conditions (36), which is continuously differentiable up to* Γ, *can be extended into* Ω_{top} *to obtain a solution using (35).*

Remark: The boundary conditions used in (36) come directly from the Euler equations. In addition, the equation for $\tilde{\rho}$ comes from (15) after substituting assumption 3 and letting $c = e^{bz}/\sqrt{a\theta}$.

Proof: To prove both parts of the corollary, we will use theorem 1 which gives us a solution for $\tilde{p}$, from which we can recover the remaining variables. Thus we need only show that u, w and ρ are valid for the Euler equations (7) - (10). We use similar arguments

to the proof of theorem 1, by noting the continuity of the extended $\tilde{p}$ and its derivatives on the whole of the domain.

Firstly, to obtain the other variables from $\tilde{p}$, we use the same equations specified in the boundary conditions (36). i.e.

$$\begin{aligned} \tilde{\rho} &= c^{-2}\tilde{p} \\ \tilde{u} &= -\int_0^t \rho_0^{-1}\partial_x \tilde{p} \\ \tilde{w} &= \int_0^t g\rho_0^{-1}\tilde{\rho} - \rho_0^{-1}\partial_z \tilde{p} \end{aligned} \tag{37}$$

Since these equations are directly obtained from the Euler equations and our assumptions, we know that they are valid. This proves part 1 of the corollary. Now it remains to check that they are valid for equations (7) - (10) on the whole domain.

For (8), since $\partial_x \tilde{p}$ is continuous on Ω, we have that $\partial_t \tilde{u}$ is as well. Thus (8) is valid on Ω.

For (9), we have that $\partial_z \tilde{p}$ is continuous on Ω from theorem 1, ρ is continuous due to (37), and $\partial_t \tilde{w}$ is as well due to the continuity of $\tilde{w}$ with respect to z. Thus (9) is valid on Ω.

For (10), we note the continuity of $\partial_t \tilde{p}$, $\partial_x \tilde{u}$ and $\tilde{w}$ on Ω by the same reasons as above. Therefore $\partial_z \tilde{w}$ must be continuous, giving us that (10) is valid on Ω.

Finally we can easily see that (7) is valid on Ω since all of the variables have proved to be continuous on Ω. □

With this, we have shown that the TBCs that we have derived allow us to limit our domain to Ω_{comp}, as we originally sought to do.

4 Numerics - 1D

In previous sections we looked at the linearised version of the Euler equations (see 2.1), and how we might derive non-reflecting boundary conditions for them by looking at an equation for the pressure (see 3 and 3.1). Now in this section we want to test these boundary conditions by deriving a numerical method for solving them and seeing how effective they are.

To simplify things we will mostly only focus on the one-dimensional case. The effect of this is that λ is set to zero in the pressure equations, and in the Euler equations we will not have u or any x derivatives. In section 5 we will look more briefly at the two-dimensional case.

For our numerical implementation, we look to use **explicit** methods, since they are much easier to implement. The cost of using explicit methods over implicit methods, is

that generally the time step must be much smaller to ensure stability, but for our purposes we are generally able to take a small enough time step without many problems.

Specifically, we will divide the section into the following pieces.

1. In section 4.1 we will derive a discretised version of the boundary condition (32) we derived in section 3 for 1D.

2. In section 4.2 we will consider a simple 1D wave equation to satisfy ourselves that we have a working method.

3. In section 4.3 we will extend this method to one which solves the 1D version of the equation for the pressure (44), and we will look to implementing our desired boundary conditions.

4. In section 4.4 we will derive a method that solves the 1D system of linearised Euler equations (i.e. (7), (9) and (10)), and again we will look at the implementation of our boundary conditions.

4.1 Discretisation of transparent boundary condition

Here we will consider the boundary condition we derived previously

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right) c\tilde{p} - b^2c^2Q^{-1}(A^\lambda *)Q\tilde{p} = 0. \tag{32}$$

In 2D, as we will see in section 5.1, the $Q^{-1}(A^\lambda *)Q\tilde{p}$ term (which comes from (27)) is the most problematic term to calculate due to the Bessel functions, but as we will show in 1D (i.e. taking $\lambda = 0$), this term can be simplified a great deal. First recall the form we used for $\hat{A}$ from section 3.1.

$$\hat{A}(\sigma) = -\sigma\frac{K_{\nu+1}(\sigma)}{K_\nu(\sigma)} + \nu + \sigma + \frac{1}{2}, \tag{27}$$

where $\nu^2 = (\lambda/b)^2 + \theta^2$. In the specific case $\lambda = 0$, and $\theta = 3/2$ (which can be obtained by letting the adiabatic exponent $\gamma = 5/3$), we note that the above Bessel functions can be expressed using the following formulae (see [1])

$$K_{3/2}(\sigma) = \sqrt{\frac{\pi}{2}}\frac{e^{-\sigma}}{\sqrt{\sigma}}\left(1 + \frac{1}{\sigma}\right), \quad K_{5/2}(\sigma) = \sqrt{\frac{\pi}{2}}\frac{e^{-\sigma}}{\sqrt{\sigma}}\left(1 + \frac{3}{\sigma} + \frac{3}{\sigma^2}\right).$$

Putting these equations into (27) with $\nu = 3/2$, we get

$$\begin{aligned}\hat{A}(\sigma) &= -\sigma\frac{1+3/\sigma+3/\sigma^2}{1+1/\sigma}+\sigma+2\\ &= -\frac{\sigma^2+3\sigma+3}{1+\sigma}+\sigma+2\\ &= -\frac{1}{1+\sigma}\end{aligned}$$

Thus it is fairly simple to obtain the term in the above formula by applying the inverse Laplace transform. We see that

$$\mathcal{L}^{-1}\left(\frac{1}{1+\sigma}\right)(\tau) = e^{-\tau}$$

Therefore putting this into (32), using the convolution formula (29) and substituting $\tau = bct$, we obtain the following 1D form for the TBC

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right)c\tilde{p} + b^2c^2\int_0^t e^{-bc(t-t')}\tilde{p}(z,t')dt' = 0. \tag{38}$$

We now look to discretise the above equation so we can use it in a numerical scheme. Mirroring what we will be doing in the following sections, we use a finite difference scheme to discretise the equation. That is to say, let $p_{i,n}$ be the mesh function that approximates $\tilde{p}$ at the point (z_i, t_n), where $0 \leq i \leq I$ and $0 \leq n \leq N$, and let $\Delta z = z_{i+1} - z_i$, $\Delta t = t^{n+1} - t^n$ be the size of the space and time steps. For convenience, let z_1 denote the last spacial point in the domain, and z_0 the point before that. Also let t^1, t^0 and t^{-1} denote the current temporal points in the domain, i.e. t^{n+1}, t^n and t^{n-1}. Since this equation is intended to be used for points on the boundary, we will use **half-space** and **time** points. We define them as follows

$$z_{\frac{1}{2}} = \frac{1}{2}(z_0+z_1), \quad t^{\frac{1}{2}} = \frac{1}{2}(t^0+t^1), \quad t^{-\frac{1}{2}} = \frac{1}{2}(t^{-1}+t^0),$$

and the evaluations of $p_{i,j}$ at those points

$$p_{\frac{1}{2},\frac{1}{2}} = \frac{1}{2}(p_{1,0}+p_{0,1}), \quad p_{\frac{1}{2},0} = \frac{1}{2}(p_{0,0}+p_{1,0}) \quad p_{\frac{1}{2},1} = \frac{1}{2}(p_{0,1}+p_{1,1}),$$
$$p_{0,\frac{1}{2}} = \frac{1}{2}(p_{0,0}+p_{0,1}), \quad p_{1,\frac{1}{2}} = \frac{1}{2}(p_{1,0}+p_{1,1}).$$

The significance of using half-space and time points is that we can use central differences at these points to approximate the derivatives of $\tilde{p}$ (which would not be possible if we evaluated them at the final points in the domain). Consequently we get the following

approximations at the half points

$$\partial_z p_{\frac{1}{2},\frac{1}{2}} \approx \frac{1}{\Delta z}(p_{1,\frac{1}{2}} - p_{0,\frac{1}{2}}) = \frac{1}{2\Delta z}(p_{1,0} + p_{1,1} - p_{0,0} - p_{0,1}),$$
$$\partial_t p_{\frac{1}{2},\frac{1}{2}} \approx \frac{1}{\Delta t}(p_{\frac{1}{2},1} - p_{\frac{1}{2},0}) = \frac{1}{2\Delta t}(p_{0,1} + p_{1,1} - p_{0,0} - p_{1,0}),$$

where $\partial_z = \partial/\partial z$ and $\partial_t = \partial/\partial t$.

This now gives us an approximation for everything in (38) except for the integral. Note that this integral must be computed for a different t^n for each time step in our domain, which would ordinarily require a large amount of resources to do. Here however, we will simplify things by using the evaluation of the integral at the previous time step t^{n-1} as a base approximation for the integral at the current time step t^n.

We do this by splitting up the integral as follows (as before we evaluate it at the half-time and space points $z_{\frac{1}{2}}$ and $t_{\frac{1}{2}}$)

$$\begin{aligned}
\int_0^{t^{1/2}} e^{-bc(t^{1/2}-t')}\tilde{p}(z^{\frac{1}{2}},t')dt' &= \int_0^{t^{-1/2}} e^{-bc(t^{1/2}-t')}\tilde{p}(z^{\frac{1}{2}},t')dt' \\
&\quad + \int_{t^{-1/2}}^{t^{1/2}} e^{-bc(t^{1/2}-t')}\tilde{p}(z^{\frac{1}{2}},t')dt' \\
&= e^{-bc\Delta t}\int_0^{t^{-1/2}} e^{-bc(t^{-1/2}-t')}\tilde{p}(z^{\frac{1}{2}},t')dt' \\
&\quad + \int_0^{\Delta t} e^{-bc(\Delta t-t')}\tilde{p}(z^{\frac{1}{2}},t'+t^{-1/2})dt'
\end{aligned}$$

Let $A^{\frac{1}{2}}$ be the approximation of the LHS integral at a given time $t^{\frac{1}{2}}$. Then we can set the first integral on the RHS to be equal to the previous approximation in time, i.e.

$$A^{-\frac{1}{2}} = \int_0^{t^{-1/2}} e^{-bc(t^{-1/2}-t')}\tilde{p}(z^{\frac{1}{2}},t')dt'$$

For the second RHS integral, since Δt is small, we can directly approximate it using Simpson's rule, i.e.

$$\int_0^{\Delta t} e^{-bc(\Delta t-t')}\tilde{p}(z^{\frac{1}{2}},t'+t^{-1/2})dt' \approx \frac{\Delta t}{6}(e^{-bc\Delta t}p_{\frac{1}{2},-\frac{1}{2}} + 4e^{-bc\Delta t/2}p_{\frac{1}{2},0} + p_{\frac{1}{2},\frac{1}{2}})$$

Putting these together, we get a formula for the new approximation in terms of the old one, i.e.

$$A^{\frac{1}{2}} = e^{-bc\Delta t}A^{-\frac{1}{2}} + \frac{\Delta t}{6}(e^{-bc\Delta t}p_{\frac{1}{2},-\frac{1}{2}} + 4e^{-bc\Delta t/2}p_{\frac{1}{2},0} + p_{\frac{1}{2},\frac{1}{2}}) \tag{39}$$

Thus (39), along with the approximations we stated earlier, give us the following finite

difference approximation for the point on the boundary

$$p_{1,1} = \left(\frac{1}{2\Delta t}(p_{0,0}+p_{1,0}-p_{0,1})+\frac{c}{2\Delta z}(p_{0,0}+p_{0,1}-p_{1,0})-\frac{c}{2}\left(b\theta+\frac{b}{2}\right)(p_{1,0}+p_{0,1})\right.$$
$$\left.-b^2c^2A^{\frac{1}{2}}\right)/\left(\frac{1}{2\Delta t}+\frac{c}{2\Delta z}\right) \tag{40}$$

We will later use this approximation as a boundary condition for our finite element method.

4.2 Discretisation of the 1D wave equation

In this section we will start with deriving a numerical method for solving the one dimensional wave equation (in the z direction). We do this in part because the equation we eventually want to solve (i.e. (44)) takes a similar form to the wave equation, and will serve as a base method, though we also would like to convince ourselves we have a working method. The 1D **wave equation** takes the form

$$\partial_{tt}u - c^2\partial_{zz}u = 0 \tag{41}$$

Here $\partial_{tt} = \partial^2/\partial t^2, \partial_{zz} = \partial^2/\partial z^2$ and c is a constant. We also specify the **initial** and **boundary conditions** to $u(z,t)$

$$u(z,0) = f(z), \quad \partial_t u(z,0) = 0,$$
$$u(0,t) = u(z_\Gamma,t) = 0,$$

where f is a function. We pose two methods for solving this equation: one a second order finite difference method, which is the standard method for solving the wave equation. The other method, an upwind scheme, is detailed in appendix B.

As we did in the previous section, we will discretise the domain in space and time. Let u_i^n be the mesh function that approximates u at the point (z_i, t_n), where $0 \le i \le I$ and $0 \le n \le N$. Then we replace the second derivatives by central differences as follows

$$\frac{\partial^2}{\partial t^2}u_i^n \approx \frac{u_i^{n+1}-2u_i^n+u_i^{n-1}}{\Delta t^2}, \quad \frac{\partial^2}{\partial z^2}u_i^n \approx \frac{u_{i+1}^n-2u_i^n+u_{i-1}^n}{\Delta z^2}$$

Substituting these into our original wave equation we get the following **finite difference scheme**

$$u_i^{n+1} = -u_i^{n-1}+2u_i^n+k^2(u_{i+1}^n-2u_i^n+u_{i-1}^n) \tag{42}$$

where $k = c\Delta t/\Delta z$. The scheme is **explicit** since it only uses previously calculated terms to calculate the next term in the sequence. In order to provide the initial step for this scheme, it is also necessary to discretise the initial conditions (namely $\partial_t u(z,0) = 0$) since

the above method at $t = 0$ requires u_i^{-1}, which is outside of the domain. We see that

$$\frac{\partial}{\partial t} u_i^0 \approx \frac{u_i^1 + u_i^{-1}}{2\Delta t} = 0,$$
$$\Rightarrow u_i^1 = u_i^{-1}.$$

Thus by substituting this into (42) we have the following **initial step** at $t = 0$

$$u_i^1 = u_i^0 + k^2/2(u_{i+1}^0 - 2u_i^0 + u_{i-1}^0).$$

In addition we discretise the other initial condition $u(z, 0) = f(z)$, and the boundary conditions $u(0, t) = u(z_\Gamma, t) = 0$ as follows

$$u_i^0 = f(z), \quad \forall\ 0 \leq i \leq I$$
$$u_0^n = u_I^n = 0, \quad \forall\ 0 \leq n \leq N$$

The stability of this scheme is dependent on the following **Courant–Friedrichs–Lewy** (CFL) condition being satisfied (however this is not necessarily sufficient)

$$|k| = \left| \frac{c\Delta t}{\Delta z} \right| < 1.$$

We now want to write a simple code to test our numerical scheme, which we will do in C++. We will set $0 < i < 200$, $0 < n < 2000$, $0 < z < 10$ and $0 < t < 20$, which consequently means $\Delta z = 0.05$ and $\Delta t = 0.01$. Thus it is necessary to take $|c| < 5$ due to the CFL condition, so we will choose in this instance $c = 0.5$. We also set the following initial condition

$$u_i^0 = f(z) = e^{(z-5)^2}$$

This initial condition represents a perturbation in the middle of the domain. Running the program, this gives us figures 1-3 of a wave over time.

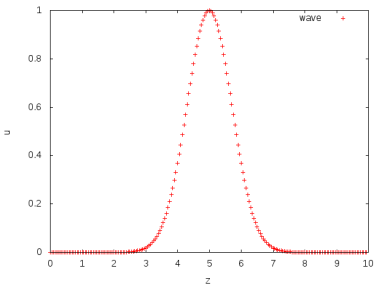


Figure 1: $t = 0$

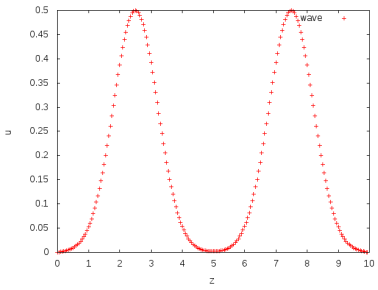


Figure 2: $t = 5$

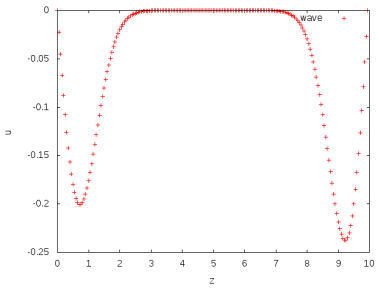


Figure 3: $t = 10.5$

We see that the wave splits in half, and travels outwards to the boundaries. Once it reaches the boundary, it reflects off and begins to travel backwards with a negative

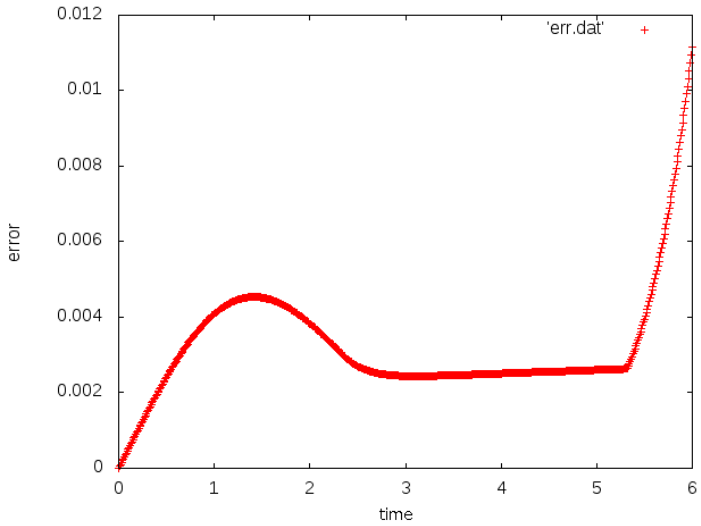


Figure 4: $|\epsilon^n|_\infty$ for $0 \leq t \leq 6$

amplitude.

For the 1D wave equation, it is relatively easy to compute the exact solution using the method of characteristics in terms of the initial condition f (see e.g. [[8], p. 65-68]). In fact d'Alembert's formula gives us

$$u(z,t) = \frac{1}{2}(e^{-(z+ct-5)^2} + e^{-(z-ct-5)^2}) \tag{43}$$

We can use this to test the accuracy of the solution on the inside of our domain (once the wave hits the boundary, it is no longer valid). We do this by calculating the **error** ϵ at each point (z_i, t_n) as

$$\epsilon_i^n = \left| u_i^n - \left(\frac{1}{2}(e^{-(z[i]+ct[n]-5)^2} + e^{-(z[i]-ct[n]-5)^2}) \right) \right|$$

To see the error over time, we can calculate for each i, $|\epsilon^n|_\infty = \sup_{0<i<I} \epsilon_i^n$. Plotting these until $t = 6$, we get figure 4. We note that the error is relatively small, up until $t \approx 5$ when the wave first reaches the boundary. Thus we can conclude that our program is simulating the solution to the wave equation in a satisfactory manner.

4.3 Discretisation of the pressure equation

After having successfully created a working code for the 1D wave equation, we want to move on to a 1D equation for the pressure, which takes a similar form. Specifically, we will consider the discretisation of the one-dimensional version of equation (20) that we derived after having implemented our assumptions and applied the Fourier transform. To convert this equation to 1D we simply look at the case of $\lambda = 0$, as this removes the

Fourier terms which we obtained from the x derivatives. For convenience we will also use the constants c and θ we introduced in section 3.1. This gives us the following **pressure equation**

$$\partial_{tt}\tilde{p} - c^2\partial_{zz}\tilde{p} - 2b\theta c^2\partial_z\tilde{p} = 0. \tag{44}$$

Recall that here, $c = e^{bz}/\sqrt{a\theta}$, and $\theta = (\gamma - 1)^{-1}$. We also plan to use the following **initial** and **boundary conditions**

$$\tilde{p}(z,0) = f(z), \quad \partial_t\tilde{p}(z,0) = 0,$$
$$\tilde{p}(0,t) = 0, \quad (38) \text{ at } z = z_\Gamma,$$

where (38) is the equation for our TBC. Comparing (44) to the wave equation we looked at in section 4.2, i.e. (41), (and letting $u \to \tilde{p}$), we see that we gain an extra $\partial_z\tilde{p}$ term, and the coefficients (namely c) now depend on z. However there is still no problem for us to extend our existing method to this equation. For the first derivative term we can use the following finite difference approximation

$$\frac{\partial}{\partial z}p_i^n \approx \frac{p_{i+1}^n - p_i^n}{\Delta z}.$$

By discretising the rest of the terms in the same way as before and rearranging, we get the following **explicit finite difference scheme**

$$p_i^{n+1} = -p_i^{n-1} + k^2 p_{i-1}^n + 2\left(1 - k^2 - \frac{\theta b c^2 \Delta t^2}{\Delta z}\right)p_i^n + \left(k^2 + \frac{2b\theta c^2\Delta t^2}{\Delta z}\right)p_{i+1}^n, \tag{45}$$

where $k = cdt/dz$. In order to calculate the **initial step** we once again make use of our initial condition $\partial_t p(z,0) = 0$ to get

$$p_i^{n+1} = \frac{k^2}{2}p_{i-1}^n + \left(1 - k^2 - \frac{\theta b c^2\Delta t^2}{\Delta z}\right)p_i^n + \left(\frac{k^2}{2} + \frac{b\theta c^2\Delta t^2}{\Delta z}\right)p_{i+1}^n.$$

To implement the TBC (38), we make use of what we calculated in (4.1), i.e. equations (39) and (40). The other conditions are discretised in the same way as before, i.e.

$$p_i^0 = f(z), \quad \forall\ 0 \le i \le I$$
$$p_0^n = 0, \quad \forall\ 0 \le n \le N$$

As before we want to test our numerical scheme by creating a C++ program. This time, we do not have an exact solution to compare the computed solution to, however we can run tests to see whether the TBC is working effectively. Namely we can run our program on a smaller domain in which the wave reaches the top boundary, and a larger domain in which the wave will take some time to reach the top boundary, and compare how similar

they are. Ideally, we want to observe little difference between the two computations, as this would imply that the boundary conditions are indeed transparent.

For the smaller domain we take $0 \leq z < 6$, and for the larger domain we take $0 \leq z < 10$. We choose the same mesh size $\Delta z = 0.02$ for both domains, giving values of I of 300 and 500 respectively, and $\Delta t = 0.001$ giving $N = 10000$. For our atmospheric values, we take $\gamma = 5/3, a = 2$ and $b = 0.4$. Given c takes values between $0.01 < c < 0.57$ for these values, our CFL condition is satisfied. For our initial conditions, we choose a perturbation that lies directly next to the upper boundary for the smaller domain, so we can easily observe the interaction

$$p_i^0 = f(z) = \begin{cases} \exp\left(\frac{-4(z-5)^2}{1-(z-5)^2}\right), & \text{for } 4 < z < 6 \\ 0. & \text{elsewhere} \end{cases}$$

Running the program gives us figures 5 and 6 (where for comparison, Dirichlet conditions are used in figure 6). As we can see from 6, the TBC case (red) quite closely mirrors the

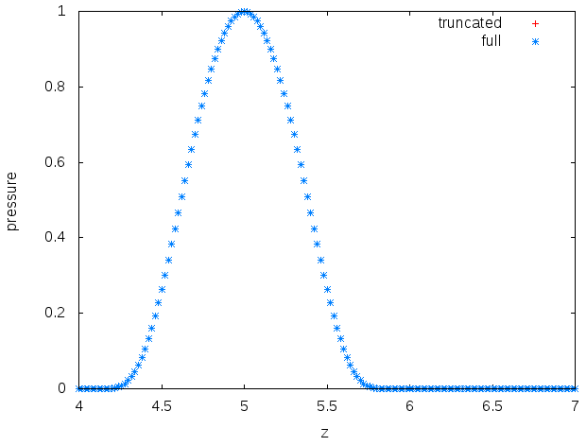


Figure 5: $t = 0$

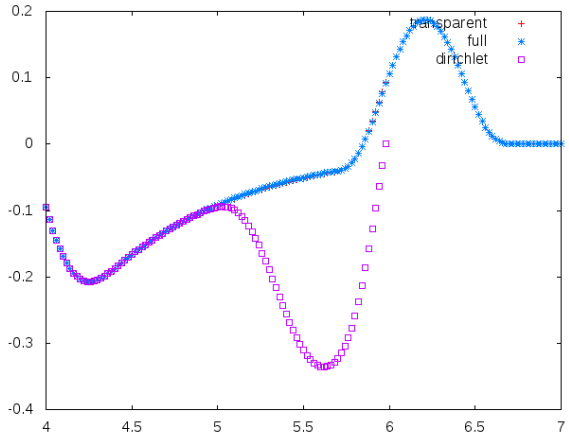


Figure 6: $t = 19$

solution from the larger domain (blue). On the other hand, setting $p = 0$ on the boundary (purple) naturally does not work so well. For a closer look at the difference between the two domains, we can look once more at the L_∞ error over time. This time we define the error ϵ_i^n as being the absolute value of the difference between the two solutions at each point (x_i, t^n), and we take $|\epsilon^n|_\infty = \sup_{0<i<300} \epsilon_i^n$. Plotting this we get figure 7.

As shown in the figure, whilst the error does increase over time, it is still relatively small compared to the size of the solution. Thus it stands to reason that the TBC is doing a good job of not reflecting waves at the boundary.

4.4 Discretisation of the Euler equations

In this section we will look at a method which directly solves the one-dimensional linearised Euler equations. The advantage of doing so over solving an equation for just the pressure,

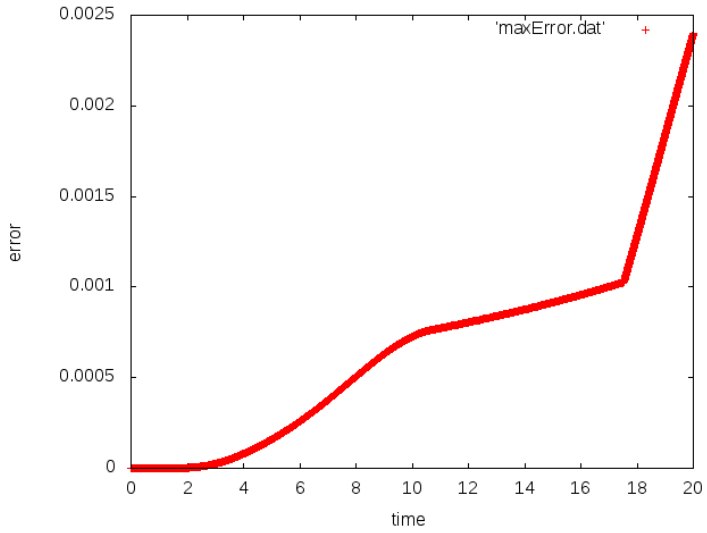


Figure 7: $|\epsilon^n|_\infty$ for $0 \leq t \leq 20$

is that we can consider the equations before applying the assumptions to our background atmosphere 2 and 3. This means that we can potentially choose different values for the background pressure and density, p_0 and ρ_0. In particular we will do this in section 4.5 where we will try to fit a more realistic atmosphere than the one using these assumptions.

Ultimately the goal of this section is to derive a solution to the Euler equations that gives us a pressure more or less identical to the one derived in section 4.3, since we are effectively solving the same equations in either case.

First of all, let us consider a 1D version of the **Euler equations** (7) - (10)

$$\frac{\partial \tilde{\rho}}{\partial t} + \rho_0 \frac{\partial \tilde{w}}{\partial z} + \tilde{w} \frac{\partial \rho_0}{\partial z} = 0, \tag{46}$$

$$\rho_0 \frac{\partial \tilde{w}}{\partial t} + \frac{\partial \tilde{p}}{\partial z} - g\tilde{\rho} = 0, \tag{47}$$

$$\frac{\partial \tilde{p}}{\partial t} + \gamma p_0 \frac{\partial \tilde{w}}{\partial z} + g\rho_0 \tilde{w} = 0. \tag{48}$$

For the initial and boundary conditions, we have to assign them differently to the case of the pressure equation. Instead of looking at $\tilde{p}$ and $\partial_t \tilde{p}$ for the initial conditions, we assign $\tilde{p}$ and $\tilde{w}$ values at $t = 0$ and obtain $\tilde{\rho}$ using (15).

For the bottom boundary, looking to make the most realistic assumption for an atmospheric model possible, we take $\tilde{w} = 0$ (signifying no vertical velocity at ground level), and recover $\tilde{p}$ and $\tilde{\rho}$ using the Euler equations. For the top boundary we again apply our TBC to $\tilde{p}$, and afterwards recover $\tilde{w}$ and $\tilde{\rho}$ using the Euler equations.

Together this gives us the following **initial** and **boundary conditions**

$$\tilde{p}(z,0) = f(z), \quad \tilde{w}(z,0) = 0,$$
$$\tilde{w}(0,t) = 0, \quad (38) \text{ at } z = z_\Gamma.$$

In addition to the previous methods, we must also assign the background values for p_0 and ρ_0, as well as the gravitational force g which is non-constant. Using assumptions 2 and 3, we calculate that they take the following forms

$$\rho_0(z) = \left(\frac{\gamma-1}{\gamma a}\right)^\theta e^{-2b\theta z}, \quad p_0(z) = \left(\frac{\gamma-1}{\gamma a}\right)^{\gamma\theta} e^{-2b\gamma\theta z}, \quad g(z) = \frac{-2b}{a} e^{-2bz} \tag{49}$$

Now we can look to discretise the above equations, using a finite difference scheme as before. We note that all the terms are at most first derivatives with respect to z and t, therefore it is a simple task to approximate them the same way as before. Substituting the finite difference approximations into (46) - (48) and rearranging them, we get the following **explicit scheme**

$$\rho_i^{n+1} = \rho_i^n - \Delta t \rho_{0,i} \frac{w_{i+1}^n - w_{i-1}^n}{2\Delta z} - \Delta t w_i^n \frac{\rho_{0,i+1} - \rho_{0,i-1}}{2\Delta z}, \tag{50}$$
$$w_i^{n+1} = w_i^n - \frac{\Delta t}{\rho_{0,i}} \frac{p_{i+1}^n - p_{i-1}^n}{2\Delta z} + \frac{\Delta t g}{\rho_{0,i}} \rho_i^n, \tag{51}$$
$$p_i^{n+1} = p_i^n - \gamma \Delta t p_{0,i} \frac{w_{i+1}^n - w_{i-1}^n}{2\Delta z} - \Delta t g \rho_{0,i} w_i^n. \tag{52}$$

For the top and bottom boundary conditions, as stated previously, we set one of the variables at a time, and recover the others using the Euler equations and our assumptions. These equations are the following

$$\rho_0 \partial_t \tilde{w} + \partial_z \tilde{p} - \tilde{\rho} g = 0 \tag{47}$$
$$\tilde{\rho} = \frac{\rho_0}{\gamma p_0} \tilde{p} \tag{15}$$

Because we are using derivatives on the outer boundary, we once again make use of the half-space and time values that we defined in section 4.1. After substituting these approximations and rearranging, this gives us the following conditions on the **bottom boundary**

$$w_0^{n+1} = 0,$$
$$p_0^{n+1} = p_1^{n+1} + p_1^n - p_0^n - \frac{\Delta z (g_0 + g_1)}{2(\rho_0^n + \rho_1^{n+1})},$$
$$\rho_0^{n+1} = \frac{\rho_{0,0}}{\gamma p_{0,0}} p_0^{n+1},$$

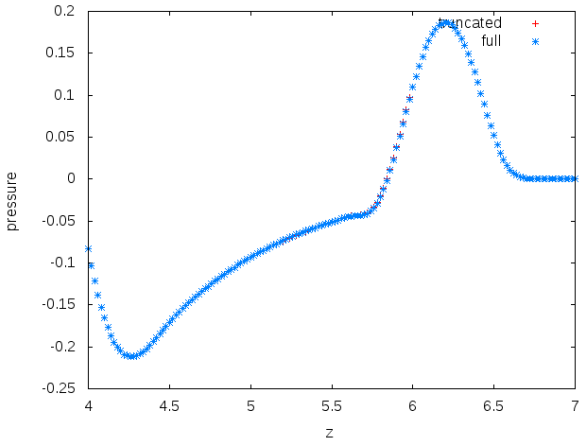


Figure 8: $t = 19$

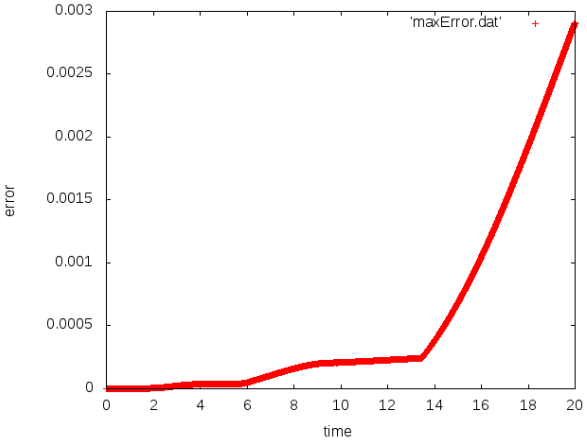


Figure 9: $|\epsilon^n|_\infty$ for $0 \leq t \leq 20$

and the following conditions on the **top boundary**

$$\begin{aligned} p_I^{n+1} &= (40) \\ \rho_I^{n+1} &= \frac{\rho_{0,I}}{\gamma p_{0,I}} p_I^{n+1} \\ w_I^{n+1} &= w_I^n + w_{I-1}^n - w_{I-1}^{n+1} - 2\Delta t \frac{p_I^{n+1} + p_I^n - p_{I-1}^{n+1} - p_{I-1}^n}{\Delta z(\rho_{0,I-1} + \rho_{0,I})} \\ &\quad + \frac{\Delta t(\rho_I^n + \rho_{I-1}^{n+1})(g_{I-1} + g_I)}{(\rho_{0,I-1} + \rho_{0,I})} \end{aligned}$$

We now consider as before the numerical implementation of this method. Seeking to reproduce the same results for the pressure as earlier (at least near the top boundary, since the bottom boundary conditions are slightly different), we use the same values for the range of z and t, for the time and space steps Δz and Δt, for the atmosphere and for the initial conditions. We will not look at the other variables w and ρ here since they are not our primary concern.

Running the program as before gives us figure 8 for the solution at the boundary, and figure 9 for the errors.

Comparing these figures to 6 and 7, we can see that they are pretty much identical, even though the errors are slightly different (this might be due to the bottom boundary conditions creating a small interference). Thus we have managed to successfully create a finite difference program for the Euler equations.

4.5 Fitting a realistic atmosphere

In this section, we intend to go a bit beyond the theory that we have definitively proved, and see whether it is possible to verify some results from a purely numerical standpoint. In particular, we want to see if it is possible to replace (partially or fully) the artificial assumptions we made for the background values p_0 and ρ_0 with values used in actual

weather forecast models. The assumptions we have in mind come from chapter 8 of Brdar's thesis [5], whereby values from the COSMO model (used by the German weather service and others) are referenced.

Here we will name the assumptions and how they differ from the ones we have used in this paper. The first is identical to assumption 1 that we have already stated

Assumption 4.

$$\frac{d}{dz}p_0 = g\rho_0, \tag{5}$$

$$\boldsymbol{u}_0 = 0. \tag{6}$$

The latter two assumptions replace assumptions 2 and 3. The second is the **ideal gas law**

Assumption 5.

$$p = \rho R_d T_0. \tag{53}$$

Here $R_d = 287.05 J/kg/K$ is the individual gas constant, and T_0 is the **background temperature**. It should be noted that this law applies to both the background values and the perturbations. The last assumption gives us this background temperature

Assumption 6.

$$T_0(z) = T_\infty + T_{ref} e^{-z/h_{ref}}. \tag{54}$$

Here $T_\infty = 225K$, $T_{ref} = 75K$ and $h_{ref} = 10km$.

To compare these to our previous assumptions, we see that as we had before, assumptions 5 and 6 similarly give us a direct relation between p and ρ. One notable difference however, is that equation (13) (from assumption 2) is not an accurate comparison to the relation between p_0 and ρ_0 in (53), since $R_d T_0 \approx 8 \times 10^5$ for the values of z we are looking at, and so if we are interested in using our artificial atmosphere as an approximation, we would need to modify (15) to more closely match (53). In fact the modified assumption we would like to use takes the following form

$$p_0 = A\rho_0^\gamma, \tag{55}$$

where $A \approx 10^5$. To use this assumption in place of the old one, it becomes necessary to check whether the rest of the analysis in section 3 remains the same. However crucially, the derivation of equation (15) remains the same, and provided we make assumption (3) as before, p_0 and ρ_0 soon drop out of the equation, leaving us with the same equation for the pressure as before. It should also be noted that under these assumptions, the gravitational force g becomes a constant $\approx -9.81 ms^{-2}$ (whereas it obeyed an exponential law previously).

Now, by using assumption 4, we can directly obtain expressions for p_0 and ρ_0 as we have in (49) by solving a differential equation. (53) gives us

$$\rho_0 = \frac{p_0}{R_d T_0} \tag{56}$$

Putting this into (5), we get the following differential equation

$$\frac{dp_0}{dz} = \frac{g}{R_d T_0} p_0$$

Solving this for p_0 gives us the following

$$p_0 = C \exp\left(\frac{g h_{ref}}{R_d T_\infty}\left(\frac{z}{h_{ref}} + \log(T_0(z)/T_0(0))\right)\right);$$

where C is a constant. In order for this equation to be accurate to the realistic atmosphere, we choose $C = p_{ref} = 10^5$. We then recover ρ_0 using (56).

To implement these new background values into our existing scheme, we ideally want to match them up with our existing values in some way such that our TBC (which depends on a and b) still works. Thus we will attempt to choose A, a and b such that p_0 and ρ_0 from our existing scheme take similar values to the ones we just calculated above. We do this by matching the coefficients at the boundary. For clarity, we denote p_0 and ρ_0 under the new assumptions as p_{real} and ρ_{real}

To show this, let us denote $\epsilon_1 = p_{real}(z_\Gamma)$, $\epsilon_2 = \rho_{real}(z_\Gamma)$ and $\epsilon_3 = \partial_z p_{real}(z_\Gamma)$. We know that our original background pressure and density obey an exponential law, so we let $\rho_0 = c_1 e^{c_2(z-z_\Gamma)}$, which means $p_0 = A\rho_0^\gamma = A c_1^\gamma e^{c_2\gamma(z-z_\Gamma)}$ (from (55)). Setting $p_{real}(z_\Gamma) = p_0(z_\Gamma)$, $\rho_{real}(z_\Gamma) = \rho_0(z_\Gamma)$ and $\partial_z p_{real}(z_\Gamma) = \partial_z p_0(z_\Gamma)$ gives us respectively

$$\epsilon_1 = A c_1^\gamma, \quad \epsilon_2 = c_1, \quad \epsilon_3 = -\gamma A c_1^\gamma c_2$$

Rearranging these in terms of c_1, c_2 and A, we find

$$c_1 = \epsilon_2, \quad A = \frac{\epsilon_1}{\epsilon_2^\gamma}, \quad c_2 = -\frac{1}{\gamma}\frac{\epsilon_3}{\epsilon_1}.$$

This gives us expressions for p_0 and ρ_0. We then use assumption 3 to recover a and b, to get

$$a = \frac{\gamma - 1}{\gamma A c_1^{\gamma-1}} e^{-c_2(\gamma-1)z_\Gamma}, \quad b = \frac{c_2(\gamma-1)}{2} \tag{57}$$

Thus we have a way to match the two different atmospheres at the boundary. With this in mind we once again look to write a program to simulate our atmosphere numerically. Since we already have the code from 4.4, this becomes simply a case of substituting in

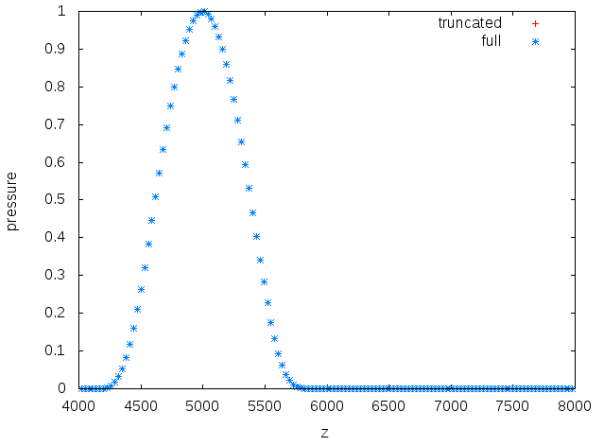


Figure 10: $t = 0$

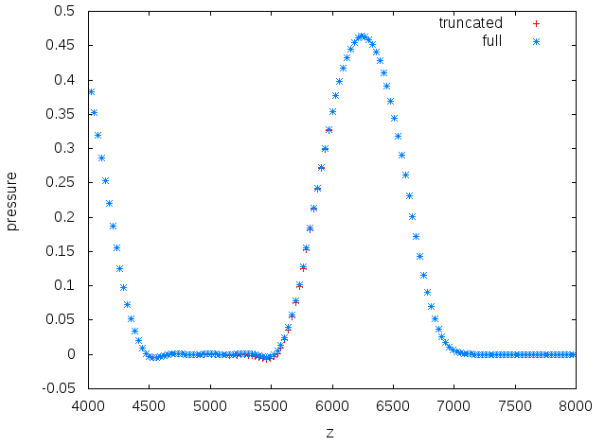


Figure 11: $t = 3.5$

the background values of p_{real} and ρ_{real}, and implementing (57) to recover a and b. In addition, for this atmosphere we must take much larger domains for z of $0 \leq z < 6000$ for the smaller domain, and $0 \leq z < 10500$ for the larger one, with $\Delta z = 30$. Consequently we must be sure to take Δt small enough to still fulfil the CFL condition, since c is much larger now. In this instance we take $0 < t < 20$ and $\Delta t = 10^{-4}$. We must also use a new **initial condition**

$$p_i^0 = f(z) = \begin{cases} \exp\left(\frac{-4(z-5000)^2}{10^6-(z-5000)^2}\right), & \text{for } 4000 < z < 6000 \\ 0. & \text{elsewhere} \end{cases}$$

For our simulations we once again run the program with the smaller and larger atmospheres, and compare the solutions near the top boundary (i.e. $z = 6000$). Doing this gives us figures 10 and 11.

Once again, we observe that our TBC is very effective at causing no reflections at the boundary, even for this untested case of a more realistic atmosphere. To observe the errors more closely, we perform the same analysis as before to get figure 12. We note that for the most part the errors are small, and once the wave has gone past a certain point, the error accumulation becomes a lot smaller.

5 Numerics - 2D

In this section we will take the ideas we used in section 4, and extend them to a two-dimensional setting. Firstly in section 5.1 we re-derive our transparent boundary conditions for the two-dimensional case. Then in section 5.2, we will look at the pressure equation (44) from section 4.3 with the λ term added back in, i.e.

$$\partial_{tt}\tilde{p} - c^2\partial_{zz}\tilde{p} - 2b\theta c^2\partial_z\tilde{p} + \lambda^2 c^2\tilde{p} = 0. \tag{64}$$

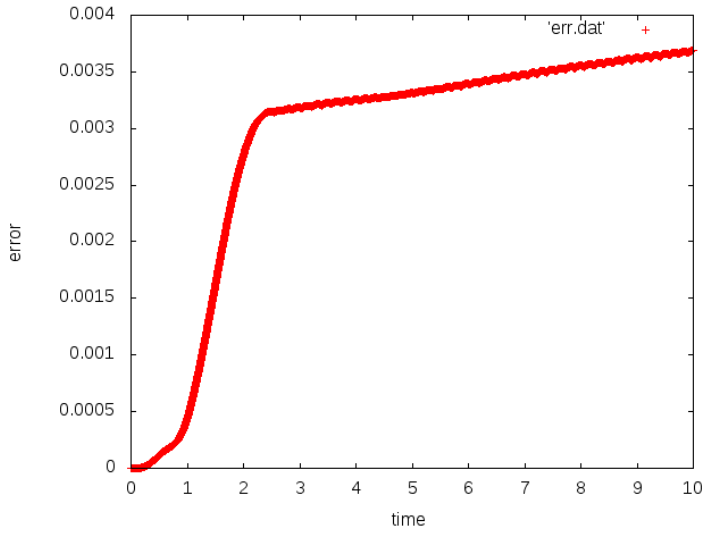


Figure 12: $|\epsilon^n|_\infty$ for $0 \leq t \leq 10$

5.1 Discretisation of transparent boundary condition for 2D

In this section we will reconsider our TBC for two dimensions (i.e. we will take $\lambda > 0$). This section will follow a similar structure to that used for the 1D case, in section 4.1, however as we will see, taking λ to be nonzero complicates the approximation somewhat. Recall the equation for the boundary condition

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right) c\tilde{p} - b^2c^2Q^{-1}(A^\lambda *)Q\tilde{p} = 0. \tag{32}$$

Once again we look at the $Q^{-1}(A^\lambda *)Q\tilde{p}$ term, though in this case, we do not have an easy way to simplify the Bessel functions. The general method we plan to use here is to first approximate $\hat{A}(\sigma)$ with a rational function which we can simplify, and then apply the inverse Laplace transform to it.

Firstly, we suppose we can approximate $\hat{A}(\sigma)$ from (27) as a rational function, i.e.

$$\hat{A}(\sigma) = \frac{P_n(\sigma)}{Q_{n+1}(\sigma)},$$

where P_n and Q_{n+1} are polynomials of degree n and $n+1$ respectively. This approximation is done using a MAPLE program, which is detailed in the appendix A. To show how we can write this in a simpler way, we pose the following theorem.

Theorem 2. *Let the roots of Q_{n+1} be $\beta_1, ..., \beta_{n+1}$ be distinct, and we also define*

$$\eta_j = \frac{P_n(\beta_j)}{Q'_{n+1}(\beta_j)}, \quad \forall\ j \in \{1, ..., n+1\}$$

Then we can rewrite $\hat{A}(\sigma)$ in terms of these values as follows

$$\hat{A}(\sigma) = \sum_{j=1}^{n+1} \frac{\eta_j}{\sigma - \beta_j}. \tag{58}$$

Proof: Since $\beta_1, ..., \beta_{n+1}$ are the roots of Q_{n+1}, we have by definition

$$Q_{n+1}(\sigma) = \prod_{k+1}^{n+1} (\sigma - \beta_k). \tag{59}$$

Using the product rule we see that the derivative of Q_{n+1} takes the form

$$Q'_{n+1}(\sigma) = \sum_{l=1}^{n+1} \prod_{k \neq l} (\sigma - \beta_k).$$

Substituting the roots into the above, we see that

$$Q'_{n+1}(\beta_j) = \prod_{k=j} (\beta_j - \beta_k), \quad \forall\ j \in \{1, ..., n+1\}. \tag{60}$$

Now let us look at the expression we are trying to get to. We see that

$$\sum_{j=1}^{n+1} \frac{\eta_j}{\sigma - \beta_j} = \frac{1}{Q_{n+1}(\sigma)} \sum_{j=1}^{n+1} \left(\frac{P_n(\beta_j)}{Q'_{n+1}(\beta_j)} \prod_{k \neq j} (\sigma - \beta_k) \right),$$

where we have used our definition of η_j, and that

$$\begin{aligned} \frac{1}{\sigma - \beta_j} &= \prod_{k+1}^{n+1} \frac{1}{\sigma - \beta_k} \cdot \prod_{k \neq j} (\sigma - \beta_k), \\ &= \frac{1}{Q_{n+1}(\sigma)} \prod_{k \neq j} (\sigma - \beta_k). \quad \text{(using (59))} \end{aligned}$$

From here we substitute equation (60) to get

$$\begin{aligned} \sum_{j=1}^{n+1} \frac{\eta_j}{\sigma - \beta_j} &= \frac{1}{Q_{n+1}(\sigma)} \sum_{j=1}^{n+1} \left(P_n(\beta_j) \prod_{k \neq j} \frac{\sigma - \beta_k}{\beta_j - \beta_k} \right), \\ &= \frac{R_n(\sigma)}{Q_{n+1}(\sigma)}. \end{aligned}$$

where we define $R_n(\sigma) = \sum_{j=1}^{n+1} \left(P_n(\beta_j) \prod_{k \neq j} (\sigma - \beta_k)(\beta_j - \beta_k)^{-1} \right)$. We note that

$$R_n(\beta_j) = P_n(\beta_j), \quad \forall\ j \in \{1, ..., n+1\},$$

thus since the degrees of the polynomials R_n and P_n are both n, and they are equal at $n+1$ distinct points, they are therefore equivalent. This gives us the desired expression as required. □

Now the next step is for us to apply the inverse Laplace transform $\mathcal{L}^{-1}$ to $\hat{A}(\sigma)$. We note that it is linear and works in the following way

$$\mathcal{L}^{-1}\left(\frac{1}{\sigma-\beta}\right)(\tau) = e^{\beta\tau}$$

Thus applying $\mathcal{L}^{-1}$ to (58) and substituting $\tau = bct$ (as in (28)) we get

$$A^{\lambda} = \sum_{j=1}^{n+1} \eta_j e^{bct}$$

Thus we substitute this into (32) to get

$$\frac{\partial}{\partial t}\tilde{p} - c\frac{\partial}{\partial z}\tilde{p} + \left(\theta b + \frac{b}{2}\right)c\tilde{p} - b^2c^2\int_0^t \sum_{j=1}^{n+1} \eta_j e^{bc\beta_j(t-t')}\tilde{p}(z,t')dt' = 0. \tag{61}$$

Comparing this equation to (38), we actually see that this is a generalisation, and that we can recover the 1D equation by taking the degree of the polynomial $n = 0$, and $\beta_0 = \eta_0 = -1$. With this in mind, we can follow the steps from section 4.1 in a similar manner to obtain the following discretised boundary condition

$$\begin{aligned} p_{1,1} \;=\; & \left(\frac{1}{2\Delta t}(p_{0,0}+p_{1,0}-p_{0,1}) + \frac{c}{2\Delta z}(p_{0,0}+p_{0,1}-p_{1,0}) - \frac{c}{2}\left(b\theta+\frac{b}{2}\right)(p_{1,0}+p_{0,1})\right. \\ & \left. + b^2c^2\sum_{j=1}^{n+1}\eta_j A_j^{\frac{1}{2}}\right) \Big/ \left(\frac{1}{2\Delta t}+\frac{c}{2\Delta z}\right) \end{aligned} \tag{62}$$

It should be noted that we now have a sum of $n+1$ number of $A^{\frac{1}{2}}$ where n is our polynomial size. They are each defined in the same way as before

$$A_j^{\frac{1}{2}} = e^{-bc\beta_j\Delta t}A_j^{-\frac{1}{2}} + \frac{\Delta t}{6}(e^{-bc\beta_j\Delta t}p_{\frac{1}{2},-\frac{1}{2}} + 4e^{-bc\beta_j\Delta t/2}p_{\frac{1}{2},0} + p_{\frac{1}{2},\frac{1}{2}}). \tag{63}$$

5.2 Discretisation of pressure equation in 2D

As discussed previously, here we consider the 2D version of (44) from section 4.3

$$\partial_{tt}\tilde{p} - c^2\partial_{zz}\tilde{p} - 2b\theta c^2\partial_z\tilde{p} + \lambda^2c^2\tilde{p} = 0. \tag{64}$$

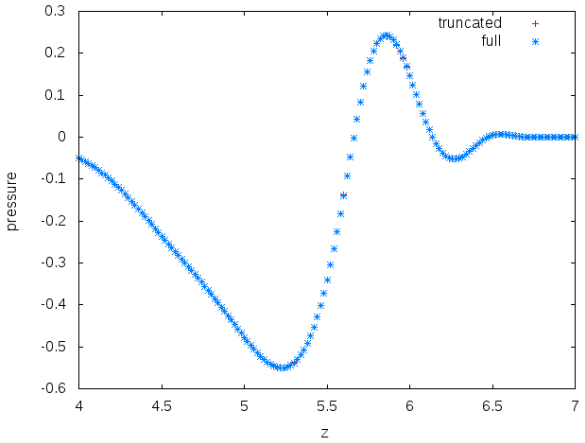


Figure 13: $\lambda = 2\pi$, $t = 19$

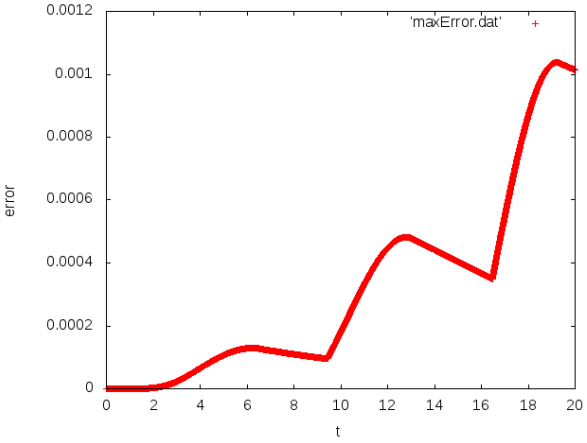


Figure 14: $|\epsilon^n|_\infty$ for $0 \leq t \leq 20$

The boundary and initial conditions are the same as the ones we used previously, save that we are now using the 2D TBC

$$\tilde{p}(z,0) = f(z), \quad \partial_t \tilde{p}(z,0) = 0,$$
$$\tilde{p}(0,t) = 0, \quad (61) \text{ at } z = z_\Gamma,$$

The scheme that we derive also takes virtually then same form as in (45), with the main difference being an additional term involving λ

$$p_i^{n+1} = -p_i^{n-1} + k^2 p_{i-1}^n + 2\left(1 - k^2 - \frac{\theta b c^2 \Delta t^2}{\Delta z} - \frac{\lambda^2 c^2 \Delta t^2}{2}\right) p_i^n + \left(k^2 + \frac{2b\theta c^2 \Delta t^2}{\Delta z}\right) p_{i+1}^n, \tag{65}$$

For the implementation, we do mostly the same thing we did in section 4.3, i.e. we compare the solution on two domains $0 \leq z < 6$ and $0 \leq z < 10$, and see how effective the TBC is. We look at two cases for λ here. In figures 13 and 14 we take $\lambda = 2\pi$, with $\Delta z = 0.02$ and $\Delta t = 0.002$. For figures 15 and 16 we take $\lambda = 18\pi$, with $\Delta z = 0.004$ and $\Delta t = 0.001$.

For both cases we see that the results on the smaller and larger domains are very similar, so we can conclude that the TBC is performing well in the 2D case.

6 Further Work

There remain a variety of avenues that can be explored with regards to this project. Most pertinently, the work covered in this paper mostly covers the 1D case and briefly looks at the 2D one. First of all, we would like to if possible extend the results for the *realistic* atmosphere to the 2D case as well. In addition, for the purposes of applications it is necessary to consider the 3D case, in particular spherical domains in the case of the global model. This would be a significant extension to make, since it would require a

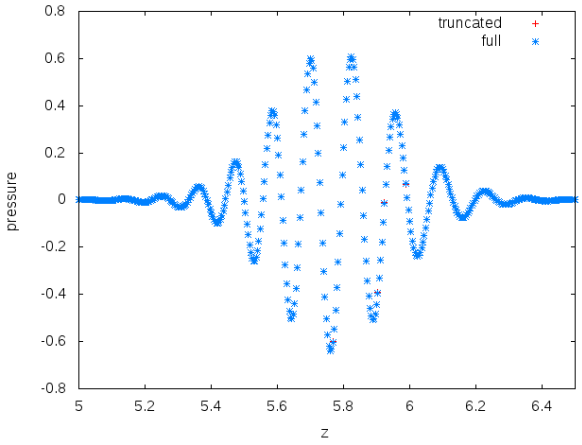


Figure 15: $\lambda = 18\pi$, $t = 29$

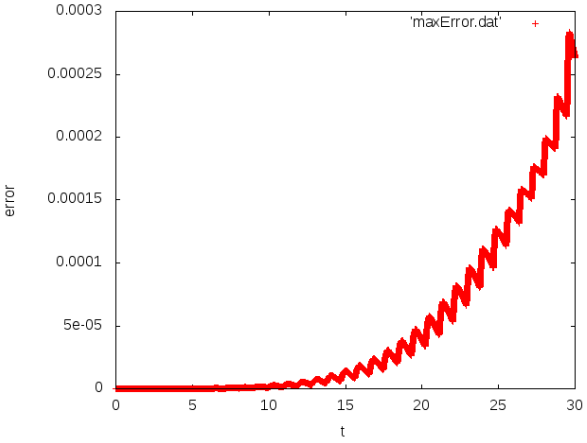


Figure 16: $|\epsilon^n|_\infty$ for $0 \leq t \leq 30$

re-derivation of the TBC.

We would also like to properly compare the TBC to other boundary conditions actually used for hyperbolic equations, such as the PML previously mentioned, and if possible the BCs used actively for weather models. This would make for a much better comparison than for instance the Dirichlet conditions that never realistically see use.

Another interesting thing to consider would be whether the TBCs take the same form if a different variation of the Euler equations are used, such as those used in [9].

Appendices

A Calculation of rational approximation

Here we will explain the Maple code used in [7] to calculate the rational function approximation of $A(\sigma)$ found in equation (27).

To give an outline of what it does, the code first expands $A(\sigma)$ in a Chebyshev series, and then converts the result to a rational function (i.e. a quotient $P(z)/Q(z)$ of two polynomials). It then calculates the coefficients needed for the inverse Laplace transform by first finding the roots β_j of $Q(z)$, and then substituting these roots into $P(z)/Q'(z)$ to find the η_j.

To start off with we define the necessary constants used.

```
### base precision ###

Digits:=25;

### constants ###
```

```
koef_a        := 2;
koef_alpha    := 0.4;
koef_gamma    := 5.0/3.0;
koef_L        := 10;
koef_theta    := 1/(koef_gamma-1);
koef_degreep  := 8;

### filename for output ###

filename:=cat(`apptop`,koef_L,`_`,koef_degreep,`.maple`);

### additional functions from libraries ###

with(numapprox);
with(orthopoly,T);
```

Here koef_a and koef_alpha correspond to a and b from assumption 3, koef_gamma is γ, the physical constant introduced with the Euler equations, koef_L is the number of λ we want to repeat the calculation for, and koef_degreep is the degree of the polynomial we are using to approximate $A(\sigma)$.

```
  f_nu:=proc(p_lambda)

  evalf(sqrt((p_lambda/koef_alpha)^2+koef_theta^2));

end;
```

Next, the function f_nu calculates ν given λ.

```
  f_Alambda:=proc(p_lambda,p_s)

  local nu;

  nu:=f_nu(p_lambda);
  -p_s*BesselK(nu+1,p_s)/BesselK(nu,p_s) + nu + p_s + 1/2

end;
```

Here f_Alambda calculates $A(\sigma)$ as given in (27) using the *Besselk* function.

```
  f_ratapp:=proc(p_f,p_an,p_degp,p_nod,p_smax)

  local f,oldD,c2,c3,fa,nfa;

  f:=proc(s)
```

```
    p_an * p_f(s)
  end;

  print(cat('f_ratapp: using ',p_nod,' digits'));
  oldD   := Digits;
  Digits := p_nod;

  c2:=chebyshev(f,s=10^(-5)..p_smax, 10^(-2*p_degp-1))/p_an;
  Digits := Digits-1;

  c3:=convert(c2,ratpoly,p_degp,p_degp+1);
  fa:=confracform(c3);

  nfa:=normal(fa);
  Digits:=oldD;
  simplify(nfa)

end;
```

f_ratapp takes a function (namely $A(\sigma)$) and finds its Chebyshev series expansion by using the function *chebyshev*. That is to say it writes it as a linear combination of the Chebyshev polynomials $T_i(z)$. Next it uses *convert* to convert this polynomial into a rational function (i.e. a quotient $P(z)/Q(z)$ of two polynomials). It does both of these operations with a higher amount of precision ($10 * koef\_degreep$ digits) than the rest of the code (25 digits by default).

The last procedures are for the sake of cleaning up the equation. *confracform* converts the rational function to continued-fraction form, which minimises the number of arithmetic operations required to evaluate it. Lastly, *normal* converts the function to factored normal form, and *simplify* performs simplification rules on the expression.

```
  f_kerapp:=proc(p_lambda,p_degp)

  local nu,nrofdigits,an,smax,f,nfa,pk,gradpk;

  print(cat('f_kerapp: calculating rational approximation of
  order ',p_degp));

  nu         := f_nu(p_lambda);
  nrofdigits := 10*p_degp;
  an         := 10^p_degp;
  smax       := 4*nu;
```

```
  f:=proc(s)
    f_Alambda(p_lambda,s)
  end;

  nfa := f_ratapp(f,an,p_degp,nrofdigits,smax);
  pk  := sort(numer(nfa));

  gradpk := degree(pk);
  print(cat('f_kerapp: reached degree ',gradpk));

  while (gradpk<p_degp) do
    nrofdigits := nrofdigits+50;
    an         := an*10;
    nfa        := f_ratapp(f,an,p_degp,nrofdigits,smax);
    pk         := sort(numer(nfa));
    gradpk     := degree(pk);
    print(cat('f_kerapp: reached degree ',gradpk));
  od;

  nfa

end;
```

The main purpose of f_kerapp is to take $A(\sigma)$ (which we obtain from f_Alambda above) and apply f_ratapp to it. The while loop at the end will keep applying f_ratapp to higher degrees of precision until the specified degree of the numerator of our rational polynomial has been reached.

```
f_invlap:=proc(p_nfa)

  local pk,qk,roots,nrofroots,r,i,coef,pcoef;

  print('f_invlap: calculating coefficients for Laplace
  transform');

  pk := sort(numer(subs(s=x,p_nfa)));
  qk := sort(denom(subs(s=x,p_nfa)));

  roots     := fsolve(qk,x,complex);
  nrofroots := 0;
  for r in roots do
    nrofroots:=nrofroots+1;
  od;
```

```
  appendto(filename);
  lprint(`M           :`,nrofroots);
  appendto(terminal);

  for i from 1 to nrofroots do
    appendto(filename);
    lprint(Re(roots[i]),Im(roots[i]));
    appendto(terminal);
  od;

  for i from 1 to nrofroots do
    appendto(filename);
    lprint(Re(subs(x=roots[i],pk)/subs(x=roots[i],diff(qk,x))),
           Im(subs(x=roots[i],pk)/subs(x=roots[i],diff(qk,x))));
    appendto(terminal);
  od;

end;
```

Once we have calculated the rational function, we are able to calculate the coefficients we need for the inverse Laplace transform with f_invlap. To do this we first find the roots of the denominator polynomial using *fsolve*. These are the β_j coefficients. We then output these roots in the form (*real part, imaginary part*). Next we output $\eta_j = P(\beta_j)/Q'(\beta_j)$ in the same form.

```
  writeto(filename);
lprint(`a           :`,evalf(koef_a));
lprint(`alpha       :`,evalf(koef_alpha));
lprint(`gamma       :`,evalf(koef_gamma));
lprint(`L           :`,koef_L);
appendto(terminal);

for n from 0 to koef_L-1 do
  print(`-----------------------------------`);
  lambda:=evalf(2*Pi*n);
  nu:=evalf(f_nu(lambda));
  print(cat(`(coefficient `,n+1,` of `,koef_L,`)`));
  print(`-----------------------------------`);
  appendto(filename);
  lprint(`lambda      :`,evalf(lambda));
  appendto(terminal);
```

```
  if (n=0) then
    print(`using exact values`);
    appendto(filename);
    lprint(`M           :`,1);
    lprint(evalf(-1.0),evalf(0.0));
    lprint(evalf(-1.0),evalf(0.0));
    appendto(terminal);
  else
    f_invlap( f_kerapp(lambda,koef_degreep) );
  fi;
  print(`-------------------------------------`);
od;
```

Lastly we look at the main part of the function. The code loops over the values of λ from 0 to $2\pi(L-1)$ and outputs the results to the file.

B Upwind scheme for the 1D wave equation

As stated in section 4.2, the standard method for numerically solving the wave equation would be a second order finite difference method. Here we will look at an approach which keeps in mind the origin of our pressure perturbation equation: the Euler equations. Specifically we will first split our wave equation into a system of first order PDEs, and solve this system instead. If we do this we get the following system in terms of u, v

$$\partial_t u - c^2 \partial_z v = 0 \tag{66}$$

$$\partial_t v - \partial_z u = 0 \tag{67}$$

We can see this system gives us back the wave equation for u by applying ∂_t to (66) and ∂_z to (67) and combining the results. Alternatively we can write this in matrix-vector form

$$\partial_t \mathbf{u} + A\partial_z \mathbf{u} = \mathbf{0}, \tag{68}$$

where

$$\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix}, \quad A = \begin{pmatrix} 0 & -c^2 \\ -1 & 0 \end{pmatrix}$$

We will use a first-order upwind scheme to solve this system. To derive this we start by diagonalising A

$$\begin{aligned} A &= \begin{pmatrix} 0 & -c^2 \\ -1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} c & -c \\ 1 & 1 \end{pmatrix} \begin{pmatrix} -c & 0 \\ 0 & c \end{pmatrix} \begin{pmatrix} 1/2c & 1/2 \\ -1/2c & 1/2 \end{pmatrix} = PDP^{-1} \end{aligned} \tag{69}$$

To apply the scheme we *separate* the diagonal matrix D into upwind and downwind components as follows

$$D^+ = \begin{pmatrix} 0 & 0 \\ 0 & c \end{pmatrix}, \quad D^- = \begin{pmatrix} -c & 0 \\ 0 & 0 \end{pmatrix}.$$

Consequently this gives us a separation for A by putting D^+ and D^- into (69)

$$A^+ = \frac{1}{2}\begin{pmatrix} c & -c^2 \\ -1 & c \end{pmatrix}, \quad A^- = \frac{1}{2}\begin{pmatrix} -c & -c^2 \\ -1 & -c \end{pmatrix}.$$

Thus for our finite difference discretisation of (68), we have the following upwind scheme

$$\frac{\mathbf{u}_i^{n+1} - \mathbf{u}_i^n}{\Delta t} + A^+\left(\frac{\mathbf{u}_i^n - \mathbf{u}_{i-1}^n}{\Delta z}\right) + A^-\left(\frac{\mathbf{u}_{i+1}^n - \mathbf{u}_i^n}{\Delta z}\right) = 0$$

We can rearrange this to get two iterative equations for u_i^{n+1} and v_i^{n+1}. Let $k = \Delta t/2\Delta z$, then we have

$$u_i^{n+1} = kcu_{i-1}^n - kc^2 v_{i-1}^n + (1-2kc)u_i^n + kcu_{i+1}^n + kc^2 v_{i+1}^n \tag{70}$$

$$v_i^{n+1} = -ku_{i-1}^n + kcv_{i-1}^n + (1-2kc)v_i^n + ku_{i+1}^n + kcv_{i+1}^n \tag{71}$$

As with the first scheme, the stability of this scheme is dependent on the following Courant–Friedrichs–Lewy (CFL) condition being satisfied

$$\left|\frac{c\Delta t}{\Delta z}\right| < 1.$$

References

[1] Modified Bessel function of the second kind: Introduction to the Bessel functions. http://functions.wolfram.com/Bessel-TypeFunctions/BesselK/introductions/Bessels/05/. Accessed: 2015-08-18.

[2] S. Abarbanel, D. Gottlieb, and J. S. Hesthaven. Well-posed Perfectly Matched Layers for Advective Acoustics. *J. Comput. Phys.*, 154:266–283, 1999.

[3] X. Antoine, A. Arnold, C. Besse, M. Ehrhardt, and A. Schädle. A Review of Transparent and Artificial Boundary Conditions Techniques for Linear and Nonlinear Schrödinger Equations. *Commun. Comput. Phys*, 4(4):729–796, 2008.

[4] T. Hagstrom B. Alpert, L. Greengard. Nonreflecting Boundary Conditions for the Time-Dependent Wave Equation. *J. Comput. Phys.*, 180:270–296, 2002.

[5] S. Brdar. *A Higher Order Locally Adaptive Discontinuous Galerkin Approach for Atmospheric Simulations*. PhD thesis, University of Freiburg, 2012.

[6] J.-P. Bérenger. A perfectly matched layer for the absorbtion of electromagnetic waves. *J. Comput. Phys.*, 114(1):185–200, 1994.

[7] A. Dedner, D. Kröner, I. L. Sofronov, and M. Wesenberg. Transparent boundary conditions for MHD simulations in statified atmospheres. *J. Comput. Phys.*, 171(2):448–478, 2001.

[8] L. C. Evans. *Partial differential equations, Second Edition*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2010.

[9] F. X. Giraldo, M. Restello, and M. Läuter. Semi-implicit formulations of the Navier-Stokes equations: application to nonhydrostatic atmospheric modeling. *J. Sci. Comput.*, 32(6):3394–3425, 2010.

[10] D. Givoli and B. Neta. High-order non-reflecting boundary conditions for dispersive waves. *Wave Motion*, 37:257–271, 2003.

[11] T. H. Pulliam. The Euler Equations, November 1994.

[12] I.L. Sofronov, L. Dovgilovich, and N. Krasnov. Application of transparent boundary conditions to high-order finite-difference schemes for the wave equation in waveguides. *Applied Numerical Mathematics*, 93:195–205, 2015.