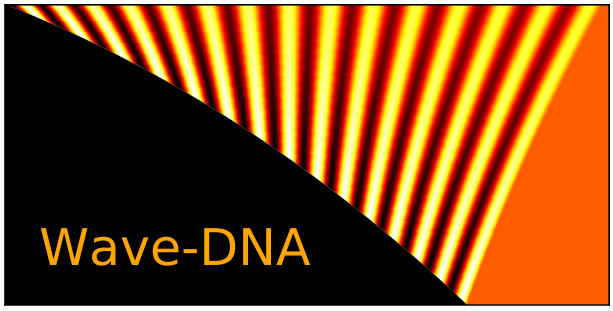
Wave-DNA

(Wave Doppler effects in Nonlinear Acoustics)



Sören Schenke
Fabian Sewerin
Berend van Wachem
Fabian Denner

9 August 2023

Contents

Chapter 1

About Wave-DNA

`Wave-DNA` is a simulation tool for one-dimensional and spherically-symmetric nonlinear acoustic waves in transient and spatially variable background flow fields. The motion of the background medium is accounted for by considering a convective form of the lossless Kuznetsov wave equation [9], derived from first principles based on perturbations of the continuity equation and the transient Bernoulli equation. The background flow field in which the acoustic waves propagate is treated as an input to the wave solver. In principle, the background flow field my be obtained from analytical considerations, numerical simulations, or experimental measurements. The tool recommends itself for the fundamental study of nonlinear Doppler phenomena in acoustic wave propagation.

The moving boundary may represent a moving wave emitter or scatterer, and it may move relative to the fluid or displace the surrounding fluid. This allows the user to study nonlinear Doppler effects associated with moving wave emitters/scatterers and/or non-uniform background flow fields. The motion of the domain boundary is conveniently taken into account by a generic coordinate transformation of the governing wave equation, where the transformed wave equation is solved in a fixed computational domain. This technique enables accurate numerical solutions of the combined wave-flow problem without the necessity to interpolate data between the moving grid points in the physical domain. The numerical technique is based on explicit finite differences, and it is equipped with a predictor-corrector method to counteract the onset and growth of dispersive numerical noise in the cases of broad frequency bands or shock formation [8]. Furthermore, absorbing boundary conditions can be imposed in order to let the acoustic waves pass the domain boundaries without reflection.

A particularly well-suited example to demonstrate the capability of the simulation framework is the so-called acoustic black/white hole analogue as presented in [7, 9]. The present documentation provides detailed instructions on how to conduct acoustic black and white hole simulations using `Wave-DNA`, as well as other examples to demonstrate the capabilities and functionalities of the tool.

The `Wave-DNA` repository is located at https://github.com/polycfd/Wave-DNA, is under the copyright of its developers and made available as open-source software under the terms of the MIT License. Details about the theory and numerical methods of `Wave-DNA`, as well as examples of its scientific applications, can be found in these papers: [7, 8, 9]. The development of `Wave-DNA` has directly benefitted from research funding provided by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), grant number 441063377.

Chapter 2

Using Wave-DNA

2.1 Installation

It is convenient to compile the source code in a build directory separate from the source directory by executing the `cleanbuild_release.sh` script to compile in optimized release mode, or by excecuting the `cleanbuild_debug.sh` script to compile in debug mode. The program runs significantly faster in release mode.

By running the build scripts (`cleanbuild_release.sh` or `cleanbuild_debug.sh`), all files in the build directory that were previously generated by `cmake` are removed. Subsequently, a `cmake` run is performed on the `CMakeLists.txt` file. In the `CMakeLists.txt` file, the `gcc` compiler must be set as `CMAKE_C_COMPILER`, providing the path to the `gcc` executable. For instance:

```
set(CMAKE_C_COMPILER /usr/bin/gcc)
```

Furthermore, the environmental variable `$mylibdirs` must be set in `CMakeLists.txt`. For instance:

```
set(mylibdirs $mylibdirs /usr/lib64/)
```

Finally, the `Wave-DNA` source directory must be specified. This is done by either directly setting the path to the source directory as

```
FILE (GLOB_RECURSE MYFILES ABSOLUTE /path/to/wavedna/src/*.c)
```

or by setting:

```
FILE (GLOB_RECURSE MYFILES ABSOLUTE $ENV{WaveDNA_DIR}/src/*.c)
```

The latter option requires to set the environmental variable `$WaveDNA_DIR` to `/path/to/wavedna` in the configuration file of the `Unix` shell, as for instance `/path/to/home/.bashrc`. With the default settings in the provided `CMakeLists.txt` file, the executable `WaveDNA` is created in the build directory upon successful compilation. See the following lines in `CMakeLists.txt`:

```
add_executable(WaveDNA ${MYFILES})
target_link_libraries(WaveDNA ${mylibs})
```

2.2 Running a simulation

The simulation case directory needs to have an options file, called `run.DNA` by default, which contains the user options[1], and a `results` directory is generated upon executing `Wave-DNA`, in which the the output files are written. The simulation is run in the case directory by executing the following command:

```
/path/to/some/arbitrary/build/directory/WaveDNA
```

The options file may also take another name different from `run.DNA`. In that case, the simulation must be executed with the following command line option:

```
/path/to/some/arbitrary/build/directory/WaveDNA -options optionsFileName
```

2.3 Run options and fluid properties

The basic run options and fluid properties are set in the sections `RUNOPTIONS` and `FLUID`, respectively. Next to the background density and speed of sound, ρ_0 and c_0, respectively, one can specify the nonlinearity coefficient β to admit cumulative nonlinearities. Independent of this option, one can further indicate, whether local nonlinearties (a non-zero Lagrangian density $\mathcal{L}$) are taken into account. By this means, one can effectively solve

- the linear wave equation ($\beta = 0$, $\mathcal{L} = 0$),
- the lossless Westervelt equation ($\beta \neq 0$, $\mathcal{L} = 0$) [8],
- or the lossless Kuznetsov equation ($\beta \neq 0$, $\mathcal{L} \neq 0$) [9].

The following table provides a detailed overview of the run and fluid property options.

Command	**Description**
Section `RUNOPTIONS`	
`TimeStart <float>`	Physical start time of the simulation.
`TimeEnd <float>`	Physical end time of the simulation.
`WriteFrequency <float>`	Number of time steps between successive write to disc occurrences. This determines the time instances at which the field data is written to disc (fields.dat). The data recorded at sample points is stored in buffer and written to disc (probes.dat) at this occasion as well.
`SamplePoints <int> <float> ...`	Number of sample points (`<int>`) followed by the coordinates (`<float>`) of the sample points. The nearest neighbour grid points to the target locations and their IDs is identified and the sample is taken at those grid points throughout the entire simulation. This means that the probe is moving with the grid. If the indicated number of sample points is larger than zero and does not match the number of given points, the simulation aborts with the error message "`An unknown option ...`".
Section `FLUID`	
`SoundSpeed <float>`	Background speed of sound c_0 of the fluid.

[1] If the file is empty, a default run is performed based on the default options as indicated in Section 2.7 (also see `src/io/iodefaultoptions.c`)

`Density <float>`	Background density ρ_0 of the fluid.
`FluidNonLinearity <float>`	Nonlinearity coefficient β of the fluid.
`LocalNonlinearity <string>`	If set to `True`, the second-order acoustic terms giving rise to local nonlinear wave propagation are active. The option `False` (default) in combination with a non-zero value for β effectively means that the Westervelt equation is solved.

2.4 Wave excitation

Several options are available to excite an acoustic pressure wave at a specific emission node. The present modeling framework is particularly designed to solve boundary value problems, where the domain boundaries act as wave-emitting boundaries. Before explaining the the different types of excitation functions, there some important remarks:

- The moving boundary is always associated with the grid node 0, regardless of whether its coordinate $R(t)$ takes a larger or smaller value than the coordinate R_{stat} of the fixed boundary. This means that the grid node $N_{\text{points}} - 1$ is always associated with the fixed boundary R_{stat}.
- Only one single excitation node can be specified in the present version.
- The user is free to specify any node between 0 and $N_{\text{points}} - 1$. However, as the present modeling framework is specifically designed to solve boundary value problems, the source pressure is not a volumetric source.
- The emission of the acoustic pressure as specified by the excitation function is always strictly enforced as per the applied boundary condition, regardless of whether the wave-emitting boundary is in relative motion to the background medium or not. Consequently, the effect of convective amplification/attenuation, which relies on the Doppler-factor of the relative motion between emitter and background medium [6], must be taken into account in the specification of the emitted acoustic pressure amplitude.

The sinusoidal excitation function for the acoustic pressure p_1 at the emission node is given by

$$p_1(t)|_{\text{emission node}} = G(t)\,\Delta p_{\text{a}}\left[\sin\left(2\pi f_{\text{a}} t - \Delta\varphi\right)\right]^n, \tag{2.1}$$

where Δp_{a}, f_{a}, and $\Delta\phi$ are the emitted acoustic pressure amplitude, the excitation frequency and a fixed phase shift, respectively. The sine function can be raised to a power n, to admit nonlinearities in the excitation signal. The factor $G(t)$ is a Gauss function given by

$$G(t)|_{\text{emission node}} = q^{4(f_{\text{a}} t - N + 1/2)^2}, \tag{2.2}$$

which has a peak value of unity and where q is a shape coefficient that specifies the width of the Gaussian. Eq. (2.2) can be used in conjunction with Eq. (2.1) to define a Gauss-enveloped sinusoidal excitation signal, where Gaussian peaks at the center of the N^{th} period. Alternatively, it can be used to emit a Gaussian pulse $\Delta p_{\text{a}} G(t)$, where f_{a} and N as well Δ_{a} must be specified to indicate the center time and the amplitude of the pulse, respectively. The parameters of Eqs. (2.1) and (2.2) are set in the section `EXCITATION` as follows.

Command	**Description**
Section `EXCITATION`	
`ExcitationStart <float>`	Physical start time of the wave excitation, equal to the simulation start time `TimeStart` per default (see `RUNOPTIONS`).

`ExcitationEnd <float>`	Physical end time of the wave excitation, equal to the simulation end time `TimeEnd` per default (see `RUNOPTIONS`).
`ExcitationNode <int>`	Node (ID of the grid point) at which the wave is excited (0 is default).
`ExcitationFunctionType <string>`	Type of the excitation function. Options are `sine` (Eq. (2.1)) and `GaussPulse`. The latter is given by Eq. (2.2) times the pressure amplitude specified below.
`PressureAmplitude <float>`	Excitation amplitude Δp_a of the acoustic pressure signal p_1. This applies to both the options `sine` and `GaussPulse`.
`ExcitationFrequency <float>`	Unmodulated frequency f_a of the excitation signal if `ExcitationFunctionType` is set to `sine`.
`PhaseShiftAngle <float>`	Phase shift angle $\Delta\varphi$ in the argument of the excitation signal if `ExcitationFunctionType` is set to `sine`.
`PowerCoeff <float>`	The sinusoidal excitation function (`sine` must be set for `ExcitationFunctionType`) can be raised to a power n (1.0 is default).
`GaussEnvelope <string>`	The excitation function can be enveloped by the Gauss function $G(t)$ with peak value 1.0 by setting this option to `True` (`False` is default). If `ExcitationFunctionType` is set to `GaussPulse`, the corresponding pointer is set to a function that returns the amplitude of the excitation signal (`PressureAmplitude`), which is then convoluted by the Gauss function, irrespective of the option for `GaussEnvelope <string>`.
`EnvelopedPeriod <float>`	Period number N of the sinusoidal excitation function that is Gauss-enveloped by Eq. (2.2) (not necessarily an integer). This setting can also be used to specify the center of the peak in the time domain if `ExcitationFunctionType` is set to `GaussPulse`.
`GaussShapeCoeff <float>`	Coefficient q in Eq. (2.2) that determines the width of the Gauss pulse in the time domain.

2.5 Boundary, grid, and background flow motion

The present simulation framework for moving domain boundaries and background flow fields admits the following configurations, where one boundary is always fixed, whereas the other can be either fixed or moving:

- **Moving emitter/scatterer in a quiescent medium:** One of the domain boundaries is in motion whereas the background medium is at rest. This configurations gives rise to the classical Doppler effect. If the moving boundary is the wave emitting boundary, it acts like a moving wave emitter. If the non-moving boundary is the wave emitting boundary, the moving boundary acts like a moving wave scatterer, given that the corresponding boundary condition is not an absorbing one.
- **Moving flow-inducing boundary:** The moving boundary induces a background flow field by mimicking a solid boundary that displaces the surrounding fluid. The wave may be emitted by the moving boundary itself, so that the emitter does not move relative to the background flow field. Alternatively, the wave may be excited at the fixed boundary (or any node between the left and the right boundary), so that emitter and background medium are in relative motion.

- **Stationary acoustic black/white hole:** A special case of the previous configuration is the acoustic black or white hole. In principle, any of the options to specify the boundary and fluid motion can mimic a background flow field with sonic transition. However, `Wave-DNA` specifically allows to simulate stationary spherical acoustic black holes and white holes with a pre-defined radius of the sonic horizon.

All of the above options can be combined with the assumption of linear acoustics ($\beta = 0$, $\mathcal{L} = 0$), or with nonlinear acoustics as specified in Section 2.3.

The distribution of the background flow velocity is either spatially uniform in the case of a one-dimensional Cartesian geometry (it may still be time-dependent if one of the boundaries is moving), or it follows from the relation

$$u_0(r,t) = \dot{R}(t)\left(\frac{R(t)}{r}\right)^n, \tag{2.3}$$

if a variable crosse-sectional area or a spherically-symmetric configuration is considered. In the present implementation, the parameter n is a constant equal to 2 per default, such that Eq. (2.3) represents the spherically-symmetric velocity field obeying the continuity equation for an incompressible fluid. The value of n may be changed in the options file, and with some minor modifications of the source, even time- and space-dependent distributions of n can be realized. However, if considerable Mach numbers are involved, it should be noted that according to the theory underpinning the present modeling framework, the background density and speed of sound are assumed to be constants as well. Therefore, the background fluid state obtained for the acoustic black hole scenario represents a so called slab geometries rather than a fully physical configuration.

It follows from the linear coordinate transformation

$$\xi(r,t) = \mathcal{X}_\infty + (r - R_{\text{stat}})\frac{\mathcal{X}_\infty - \mathcal{X}_R}{R_{\text{stat}} - R(t)} \tag{2.4}$$

that

$$\frac{\partial \xi}{\partial r} = \frac{\mathcal{X}_\infty - \mathcal{X}_R}{R_{\text{stat}} - R(t)}, \tag{2.5}$$

$$\frac{\partial \xi}{\partial t} = -\frac{\mathcal{X}_\infty - \xi}{R_{\text{stat}} - R(t)}\dot{R}(t), < \tag{2.6}$$

$$\frac{\partial^2 \xi}{\partial t \partial \xi} = \frac{\dot{R}(t)}{R_{\text{stat}} - R(t)}, \tag{2.7}$$

$$\frac{\partial^2 \xi}{\partial r \partial \xi} = 0, \tag{2.8}$$

$$\frac{\partial^2 \xi}{\partial t^2} = 2\frac{\partial \xi}{\partial t}\frac{\partial^2 \xi}{\partial t \partial \xi} - \ddot{R}\frac{\partial \xi}{\partial r}\frac{\mathcal{X}_\infty - \xi}{\mathcal{X}_\infty - \mathcal{X}_R}. \tag{2.9}$$

The motion of the grid-points is then fully described by the instantaneous position R of the moving domain boundary and the velocity $\dot{R}$ and acceleration $\ddot{R}$ thereof. In the case of linear boundary motion, $R = \text{const}$ so that $\dot{R} = 0$ and $\ddot{R} = 0$.

An oscillatory boundary motion is described by the function

$$R(t) = R_0 + \frac{\Delta v_{\text{b,a}}}{2\pi f_{\text{b}}}\sin(2\pi f_{\text{b}} t), \tag{2.10}$$

$$\dot{R}(t) = \Delta v_{\text{b,a}}\cos(2\pi f_{\text{b}} t), \tag{2.11}$$

$$\ddot{R}(t) = -2\pi f_{\text{b}}\Delta v_{\text{b,a}}\sin(2\pi f_{\text{b}} t), \tag{2.12}$$

where f_{b} and $\Delta v_{\text{b,a}}$ are the frequency and the velocity amplitude of the oscillating boundary, respectively.

For the acoustic black and white hole analogues, with r_h being the steady-state sonic horizon radius in a flow field obeying Eq. (2.3), the functions describing the boundary motion are given by

$$R\left(t\right) = \left[R_0^{n+1} - \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{\frac{1}{n+1}}, \tag{2.13}$$

$$\dot{R}\left(t\right) = -c_0 r_\mathrm{h}^n \left[R_0^{n+1} - \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{-\frac{n}{n+1}}, \tag{2.14}$$

$$\ddot{R}\left(t\right) = -n c_0^2 r_\mathrm{h}^{2n} \left[R_0^{n+1} - \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{-\frac{2n+1}{n+1}} \tag{2.15}$$

for the acoustic black hole scenario, and by

$$R\left(t\right) = \left[R_0^{n+1} + \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{\frac{1}{n+1}}, \tag{2.16}$$

$$\dot{R}\left(t\right) = c_0 r_\mathrm{h}^n \left[R_0^{n+1} + \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{-\frac{n}{n+1}}, \tag{2.17}$$

$$\ddot{R}\left(t\right) = -n c_0^2 r_\mathrm{h}^{2n} \left[R_0^{n+1} + \left(n+1\right) c_0 r_\mathrm{h}^n t\right]^{-\frac{2n+1}{n+1}} \tag{2.18}$$

for the acoustic white hole scenario.

Command	**Description**
Section `BOUNDARYMOTION`	
`BoundaryMotionType <string>`	Function type describing the motion of the moving domain boundary. Options are `<linear>` for a constant velocity, `<oscillating>` for an oscillatory motion around the initial position, described by a sine function (see Eq. (2.10)), and `<stationaryBlackHole>` as well as `<stationaryWhiteHole>` for the simulation of a stationary sonic horizon in acoustic black black hole or white hole configurations (see Eq. (2.13)).
`BoundaryMotionStartTime <float>`	Physical time at which the moving boundary starts to move from its intial position. The function describing the boundary motion depends on the time interval that has passed relative to this start time.
`BoundaryMotionEndTime <float>`	Physical time at which the boundary motion ends. The boundary position will remain constant for the remaining simulation time.
`InitialMovingBoundaryPosition <float>`	Initial position of the moving domain boundary, where 0.0 is default.
`FixedBoundaryPosition <float>`	Position of the fixed domain boundary. The default setting is a domain that comprises ten emitted wavelengths $\lambda_0 = c_0/f_\mathrm{a}$ based on the given or the default `InitialMovingBoundaryPosition`.
`MovingBoundaryVelocityAmplitude <float>`	Velocity amplitude $\Delta v_\mathrm{b,a}$ of the moving domain boundary if `BoundaryMotionType` is set to `<oscillating>`. If `BoundaryMotionType` is set to `<linear>`, $\lambda_0 = c_0/f_\mathrm{a}$ represents the constant boundary velocity.
`MovingBoundaryFrequency <float>`	Frequency f_b of the moving domain boundary if `BoundaryMotionType` is set to `<oscillating>`, where $f_\mathrm{b} = 0.1 f_\mathrm{a}$ is default.
Section `BACKGROUNDFLOW`	

`BackgroundMotionMode <string>`	Specifies whether the background flow field is unconditionally quiescent (option `<quiescent>`), regardless of the motion of the moving domain boundary, or whether the background flow field is coupled to the moving domain boundary (option `<coupledToMovingBoundary>`).
`Geometry <string>`	Specifies whether the flow field is spatially homogeneous (option `<1dCartesian>`) or satisfying the continuity equation of a spherically symmetric flow (option `<3dSphericallySymmetric>`).
`HorizonRadius <float>`	Point of sonic transition if `BoundaryMotionType` is set to `<stationaryBlackHole>` or `<stationaryWhiteHole>`.
`BackgroundVelocityScalingFactor <float>`	Exponent n in Eq. (2.3), defining the parametrization of the spatially non-uniform background flow.
`GravitationalPotential <string>`	If set to `True` and if `BackgroundMotionMode` is set to `<stationaryBlackHole>` or `<stationaryWhiteHole>`, a gravitational potential is included to mimic a simplified Bondi-Parker-type flow.

2.6 Finite-difference discretization

To be compatible with the predictor-corrector method and the wave-absorbing boundary conditions as explained further below, we use first-order finite differences in time. For the spatial discretization, second-order central differences are used. The Laplacian and the mixed spatial-temporal derivatives only have to be evaluated in the inner field, whereas the gradient also has to be evaluated at the boundary, to impose the wave-absorbing boundary condition. At the boundaries, a first-order finite difference stencil is used to compute the gradient.

Next to the time step size Δt and the total number of grid points, one can specify the parameter γ of a predictor-corrector method as developed by Dey and Dey [2] and Nascimento and Pestana [5], where the interim solution $\widetilde{\Phi}_1$, predicted by the explicit finite-difference integration, is weighted by $(1-\gamma)$, whereas the corrected solution is weighted by γ. With i and j indicating the grid point and the time step, respectively, the spatial and temporal finite-difference approximations are given by

$$\left(\frac{\partial^k \Phi_1}{\partial t^k}\right)^j_i = \frac{a_0^{(k)}}{\Delta t^k}\Phi^{j+1}_{1,i} + \frac{1}{\Delta t^k}\underbrace{\sum_{n=1}^{N-1} a_n^{(k)}\Phi^{j+1-n}_{1,i}}_{b_i^{(k,j)}}, \tag{2.19}$$

$$\left(\frac{\partial^k \Phi_1}{\partial \xi^k}\right)^j_i = \frac{a_0^{(k)}\Phi^j_{1,i} + \sum_{n=1}^{(N-1)/2}\left(a_n^{(k)}\Phi^j_{1,i+n} + a_n^{(k)}\Phi^j_{1,i-n}\right)}{\Delta\xi^k}. \tag{2.20}$$

The predicted solution $\widetilde{\Phi}_{1,i}$ is obtained as

$$\widetilde{\Phi}_{1,i}\underbrace{\sum_{k=1}^{2}\mathcal{A}^j_{k,i}\frac{a_0^{(k)}}{\Delta t^k}}_{\mathcal{H}^j_i} + \underbrace{\mathcal{R}^j_i + \sum_{k=1}^{2}\mathcal{A}^j_{k,i}\frac{b_i^{(k,j)}}{\Delta t^k}}_{\mathcal{S}^j_i} = -\mathcal{A}^j_{L,i}\left(\frac{\partial^2\Phi_1}{\partial\xi^2}\right)^j_i + \left(\frac{c_0^2}{A}\frac{\partial A}{\partial r}\frac{\partial\xi}{\partial r}\right)^j_i\left(\frac{\partial\Phi_1}{\partial\xi}\right), \tag{2.21}$$

where the term $\mathcal{R}^j_i$ involves the finite difference approximations of the gradient and mixed derivative

terms. The new solution $\Phi_{1,i}^{j+1}$ is then obtained as

$$\Phi_{1,i}^{j+1} = (1-\gamma)\,\widetilde{\Phi}_{1,i} + \frac{\gamma}{\mathcal{H}_i^j\left(\Phi_{1,i}^j\right)}\left[\underbrace{\mathcal{A}_{L,i}^j\left(\Phi_{1,i}^j\right)\left(\frac{\partial^2\widetilde{\Phi}_1}{\partial\xi^2}\right)_i - \left(\frac{c_0^2}{A}\frac{\partial A}{\partial r}\frac{\partial\xi}{\partial r}\right)_i^j\left(\frac{\partial\widetilde{\Phi}_1}{\partial\xi}\right)_i}_{\text{spherical Laplacian}} + \mathcal{S}_i^j\left(\Phi_{1,i}^j\right)\right], \quad (2.22)$$

where the spherical Laplcian is updated based on the predicted solution $\widetilde{\Phi}_1$, whereas all remaining terms, summarized in the terms $\mathcal{H}$ and $\mathcal{S}$, and the time- and space-dependent coefficient $\mathcal{A}_L$, remain unchanged in the corrector step. The corrector step may be performed multiple times in a corrector loop. However, a single corrector step should serve the purpose of counteracting the dispersive noise. In fact, the value of γ is more important. For $\gamma = 0$, the standard explicit finite-difference time integration scheme is obtained. With increasing γ, the predictor-corrector method becomes increasingly diffusive. The resolution of the simulation must be increased at constant Courant-Friedrichs-Lewy (CFL) number to avoid excessive diffusion while preserving the stabilizing effect of the scheme. To this end, the CFL number is given by

$$\mathrm{CFL} = \frac{c_0\Delta t}{\Delta x} = \frac{c_0\Delta t\,(N_{\mathrm{points}-1})}{L_{\mathrm{domain}}}, \quad (2.23)$$

where Δt and Δx are the time-step size and the spatial step size, respectively, and N_{points} and L_{domain} are the total number of grid points in the domain, including the boundaries, and the length of the physical domain, respectively. In other words, the increase in numerical stability comes at the expense of an increased computational effort to preserve the accuracy of the numerical solution. Rather than representing a blended solution, the predictor-corrector method effectively generates a Δx- and Δt-dependent attenuation term [5] predominantly acting on the higher frequencies. Therefore, values of $\gamma \leq 1$ are possible and may even be required, in particular if shocks are involved.

The boundary conditions can either be scattering, where the incoming wave is fully reflected at the domain boundary, or absorbing, where the incoming wave passes the domain boundary without reflection. The absorbing boundary condition is based on Mur's first-order absorbing boundary condition, given by [4]

$$\Phi_{1,0}^{j+1} = \Phi_{1,1}^j + \frac{\mathrm{CFL}-1}{\mathrm{CFL}+1}\left(\Phi_{1,1}^{j+1} - \Phi_{1,0}^j\right) + 2u_{1,1}^j\frac{\Phi_{1,0}^j - \Phi_{1,N}^j}{\Delta x}, \quad (2.24)$$

$$\Phi_{1,N}^{j+1} = \Phi_{1,N-1}^j + \frac{\mathrm{CFL}-1}{\mathrm{CFL}+1}\left(\Phi_{1,N-1}^{j+1} - \Phi_{1,N}^j\right) + 2u_{1,N}^j\frac{\Phi_{1,N-1}^j - \Phi_{1,N}^j}{\Delta x} \quad (2.25)$$

for the West and the East boundary, respectively, and where the advective terms are added in order to allow the waves to pass the domain boundaries in the presence of a non-zero background flow field.

The following table provides an overview of the options for the finite difference discretization.

Command	**Description**
Section `FINITEDIFFERENCE`	
`TimeStepSize <float>`	Constant time step size Δt, where the acoustic CFL number takes the value 0.1 per default.
`NPoints <int>`	Total umber of grid points in the computational domain, including the domain boundaries, where the default setting is such that 100 points per emitted wavelength $\lambda_0 = c_0/f_\mathrm{a}$ are obtained based on the given or default domain size.

`nCorrectors <int>`	Number of corrector steps applied in the predictor-corrector method to counteract the formation of dispersive numerical noise (1 per default).
`correctorWeight <float>`	Corrector weight γ to specify the weight of the corrected solution in the predictor-corrector method, where $\gamma \geq 0$ and $\gamma = 0.1$ per default.
`BoundaryConditionEast <string>`	Boundary condition at the Eastern (right) domain boundary. Options are `<scattering>` for wave-reflecting boundaries and `<absorbing>` (default) for non-reflecting boundaries.
`BoundaryConditionWest <string>`	Boundary condition at the Western (right) domain boundary. Options are `<scattering>` for wave-reflecting boundaries and `<absorbing>` (default) for non-reflecting boundaries.

2.7 Default settings

These are the default settings for the `Wave-DNA` options file `run.DNA`, indicating the dependencies between certain dafault settings (see the source file `src/io/iodefaultoptions.c`). The default settings are used if the corresponding option is not included in `run.DNA`.

```
FLUID
SoundSpeed 1500.0
Density 1000.0
FluidNonLinearity 0.0
LocalNonlinearity False
END


RUNOPTIONS
TimeStart 0.0
TimeEnd 10.0/<ExcitationFrequency>
WriteFrequency (<TimeEnd> - <TimeStart>)/<TimeStepSize>
SamplePoints 0
END

BOUNDARYMOTION
BoundaryMotionType linear
BoundaryMotionStartTime <TimeStart>
BoundaryMotionEndTime <TimeEnd>
InitialMovingBoundaryPosition 0.0
FixedBoundaryPosition 10.0*<SoundSpeed>/<ExcitationFrequency>
MovingBoundaryVelocityAmplitude 0.0
MovingBoundaryFrequency 0.1*<ExcitationFrequency>
END

BACKGROUNDFLOW
Geometry 1dCartesian
BackgroundMotionMode quiescent
HorizonRadius 0.9*<InitialMovingBoundaryPosition>
BackgroundVelocityScalingFactor 2.0
GravitationalPotential False
```

```
END

EXCITATION
ExcitationStart <TimeStart>
ExcitationEnd <TimeEnd>
ExcitationNode
ExcitationFunctionType sine
PressureAmplitude 1.0e3
ExcitationFrequency 1.0e5
PhaseShiftAngle 0
PowerCoeff 1
GaussEnvelope False
EnvelopedPeriod 1
GaussShapeCoeff 1.0
END

FINITEDIFFERENCE
TimeStepSize (<FixedBoundaryPosition> - <InitialMovingBoundaryPosition>)/...
                ...(<NPoints> - 1)*0.1/<SoundSpeed>
NPoints 100*(<FixedBoundaryPosition> - <InitialMovingBoundaryPosition>)/...
                ...(<SoundSpeed>/<ExcitationFrequency>)) + 1
nCorrectors 1
correctorWeight 0.1 BoundaryConditionEast absorbing
BoundaryConditionWest absorbing
END
```

2.8 Results

Three different result files may be written to the `results` sub-directory:

- `fields.dat`,
- `probes.dat`,
- `horizon.dat`.

The output file `fields.dat` is always written to disk and contains the field data (data at the grid points). The data is written at time instances specified by the `WriteFrequency`. The output columns represent the data for

$$\text{Grid point ID } [-], \quad x\,[\text{m}], \quad p_1\,[\text{Pa}], \quad \phi_1\,\left[\text{m}^2\text{s}^{-1}\right], \quad u_0\,\left[\text{ms}^{-1}\right].$$

Each output section starts with indicating the physical time, the corresponding time step number, the instantaneous position R of the (possibly) moving boundary as well as its instantaneous velocity $\dot{R}$. The output section concludes with the key word `EOS`.

The output file `probes.dat` is only written if a non-zero number $N_{\text{sample nodes}}$ of `SamplePoints` is given. The sample points represent grid nodes for which time signals of the acoustic pressure are written to disk. The output columns represent

$$\text{Time step } [-], \quad t\,[\text{s}], \quad p_1\,(\text{sample node } 1)\,[\text{Pa}], \quad ..., \quad p_1\,(\text{sample node } N_{\text{sample nodes}})\,[\text{Pa}].$$

The grid nodes may be moving if one of the domain boundaries is in motion.

The output file `horizon.dat` is only written if the parameter `BoundaryMotionType` is set to `stationaryBlackHole` or `stationaryWhiteHole` as described in Section 2.5. The outfile contains time signals recorded at the sonic horizon of the acoustic black hole, representing

$$\text{Time step } [-], \quad t\,[\text{s}], \quad R\,[\text{m}], \quad \dot{R}\,\left[\text{ms}^{-1}\right], \quad r_{\text{h}}\,[\text{m}], \quad p_1\,[\text{Pa}], \quad \phi_1\,\left[\text{m}^2\text{s}^{-1}\right].$$

The sonic horizon radius r_{h} should be constant over time in the case of a stationary acoustic black or white hole.

Chapter 3

Source code guide

3.1 Header files

The header files (`*.h`) are located in `src/include`. In `DNA-functions.h`, the function prototypes are defined. The file `DNA-constants.h` contains macros of constants (such as π) and mathematical operators (such as trigonometric functions or min/max operators). In the `DNA.h` file, all structures used throughout the program are declared. There are four main structures, the instances of which are created in `main.c`:

- `struct DNA_RunOptions`: Structure containing variables and function pointers associated with the run options. Nested structures:
  - `struct DNA_NumericsFD`: Structure containing variables used for the temporal and spatial discretization (e.g. Δt, Δx, etc.) and function pointers associated with the boundary conditions. Nested structure:
  - `struct DNA_FDCoeffs`: Finite-difference coefficients.
  - `struct DNA_WaveExcitation`: Variables of the wave excitation functions (see Eqs. (2.1) and (2.2)).
  - `struct DNA_Probes`: Variables specifying the acoustic pressure probes (see Section 2.3).
- `struct DNA_FluidProperties`: Fluid properties such as the background density and speed of sound, as well as the fluid nonlinearity.
- `struct DNA_Fields`: Structure comprising all scalar fields. Nested structures:
  - `struct DNA_Grid`: Scalars and scalar fields, represented by pointers, representing the dependent coordinates $x(\xi, t)$ of the time-dependent physical domain $\Omega(t)$, the fixed computational domain Θ, and the metrics of the coordinate transformation according to Eqs. (2.5) to (2.9).
  - `struct DNA_BackgroundFlowField`: Scalar fields, represented by pointers, involving the background flow velocity field as well as spatial and temporal derivatives thereof.
  - `struct DNA_PhiField`: Scalar fields, represented by pointers, involving the acoustic (velocity) potential field (the primary solution) as well as spatial and temporal derivatives thereof. The acoustic pressure is included in this structure as well.
  - `struct DNA_OldPhiField`: Similar to the previous, but containing the data from the two preceding time steps. This information is required for the time integration.
- `struct DNA_MovingBoundary`: Variables associated with the moving boundary of the physical domain $\Omega(t)$.

3.2 Source files

The core of the computational method evolves around solving the lossless convective Kuznetsov equation [9] in the coordinates of the fixed computational domain. The following table provides an overview and brief description of the source files.

Source file	Description
Directory `backgroundflow`	
`backgroundflowgravitation.c`	Function to compute the gravitational potential if applicable.
`backgroundflowmotion.c`	Functions to compute the background flow velocity field u_0 and its spatial and temporal derivatives.
Directory `boundary`	
`boundaryconditions.c`	Calling the functions to set the boundary conditions.
`boundarymotion.c`	Functions to describe the motion of the moving domain boundary.
Directory `fd`	
`fdboundaryconditions.c`	Functions to apply the boundary conditions. The functions for the West and East boundaries are called by reference. The corresponding pointers are `int (*FDBC_West)(...)` and `int (*FDBC_East)(...)`, respectively.
`fdcoeffs.c`	Finite difference coefficients for the spatial and temporal discretization.
`fddt.c`	Functions to construct the explicit finite difference approximations of the time derivatives.
`fddx.c`	Functions to construct the explicit finite difference approximations of the spatial derivatives.
`fdsumaphi.c`	Functions to compute the sums over previous timesteps needed for the finite difference discretiztion, see Eq. (2.19).
Directory `grid`	
`gridmakegrid.c`	Functions to create the time-dependent grid for the physical domain $\Omega(t)$ and the fixed grid for the computational domain Θ.
`gridmotion.c`	Updating the derivatives of the dependent coordinate $\xi(r,t)$ as given by Eqs. (2.5) to (2.9).
Directory `include`	
`DNA.h`	See Section 3.1.
`DNA-constants.h`	See Section 3.1.
`DNA-functions.h`	See Section 3.1.
Directory `initialize`	
`initializefield.c`	Routines to initialize single fields of the type `struct DNA_Fields`.
`initializeprocessoptions.c`	Setting all function pointers based on the specifications in the `Wave-DNA` options file or the default options.

`initializesimulation.c`	Routine that allocates memory for all scalar fields and results vectors by calling functions in `memoryallocfields.c` (see below). Initial values are set and the grid point IDs corresponding to the coordinates of the probe sample locations (IDs of the nearest neighbours are taken) specified in the `Wave-DNA` options file are identified.
Directory `io`	
`iodefaultoptions.c`	Defaults for the options that can be set in the `Wave-DNA` options file. A simulation is performed even with an empty options file (see Section 4.1).
`ioonscreen.c`	Functions for output on screen during the run time of the simulation.
`ioreadcommandlineoptions.c`	Function to read command line options.
`ioreadoneoption.c`	Called in `ioreadoptionsfile.c` to read a single option the `Wave-DNA` options file.
`ioreadoptionsfile.c`	Function that reads the `Wave-DNA` options file.
`ioresults.c`	Functions to write ouput data to disk.
Directory `memory`	
`memoryalloc.c`	Function to allocate memory for fields of the type `struct DNA_Field` and the probe sample points and vectors.
`memoryallocfields.c`	Routine calling `memoryalloc.c` to allocate all fields.
`memoryfreefields.c`	Routine called in `main.c` to free all fields upon termination of the programme.
Directory `solve`	
`solveexplicitderivatives.c`	Routine calling the functions in `fddx.c` and `fddt.c` to construct the finite difference approximations explicit temporal, spatial, and mixed derivatives.
`solveloop.c`	This file contains the function `SolveTimeLoop`, representing the time loop of the simulation, and the function `Solve`, which is called in `SolveTimeLoop` at each time step and which calls the functions relevant to the numerical algorithm.
`solvepredictorcorrector.c`	Routines to apply the predictor (see Eq. (2.21)) and the corrector steps (see Eq. (2.22)). Additional routines are to update the Laplacian prior to the corrector step and to store the initial guess $\widetilde{\Phi_{1,i}}$ computed based on Eq. (2.21) and required in Eq. (2.22).
`solvesumaphi.c`	Routine calling the functions in `fdsumaphi.c` to compute the sums over the previous time steps required for the explicit finite difference approximations of the time-derivatives.
`solveupdatefields.c`	Contains the function that updates the old fields.
Directory `transform`	

`transformeqn.c`	Functions to compute the space- and time-dependent coefficients of the transformed wave equation, and to assemble the terms of the discretized equation. The file contains the function `TransformEqn_Predictor` to assemble the terms for the predictor step of the predictor-corrector method (see Eq. (2.21)), and the function `TransformEqn_Corrector`, which only updates the coefficients and terms that are relevant for the corrector step, see Eq. (2.22).
`transformgeometricaldecay.c`	Function to account for the change of the cross-sectional area A in the Laplacian in the case of spherical symmetry.
`transformpotential.c`	Routines to convert the acoustic pressure p_1 into the perturbation potential Φ_1 (required at the excitation node) and vice versa (required to reconstruct the acoustic pressure field from the solution for Φ_1).
Directory `waveexcitation`	
`waveexcitation.c`	Functions to excite the acoustic pressure wave at the specified excitation node.

Chapter 4

Examples

The following examples are found in the `Wave-DNA` source directory `</path/to/wavedna/examples/>`. The case folder must contain the following files and directorie

4.1 Default run

This example is found in `examples/DefaultRun`.

If the `run.DNA` options file is empty, the simulation is exclusively based on the default options. The result is a sinusoidal wave comprising ten emitted wave periods in a domain that comprises ten wavelengths. The is excited at the left (West) boundary and travels from left to right. The CFL number is equal to 0.1 and the spatial resolution is $N_{\text{ppw}} = 100$ points per wavelength. `fields.dat` is the only output file as no probes are specified. The instantaneous obtained at a physical time of ten wave periods is shown in Fig. 4.1. See Sec. 2.7 and/or the source file `src/io/iodefaultoptions.c` for the default settings.

4.2 Absorbing vs reflecting boundary conditions

This example is found in `examples/ScatteringAbsorbing`.

This test case illustrates the options for the boundary conditions. A Gaussian pulse is emitted as indicated by the red lines in Fig. 4.2 (see the `run.DNA` file). The pulse is emitted at $x = 0$ and travels from left to right. In Fig. 4.2 (a), the settings are:

```
BoundaryConditionEast scattering
```

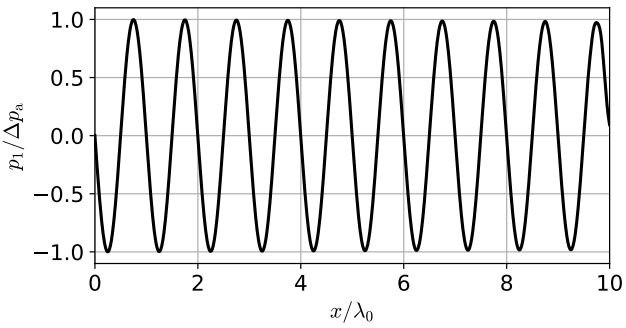
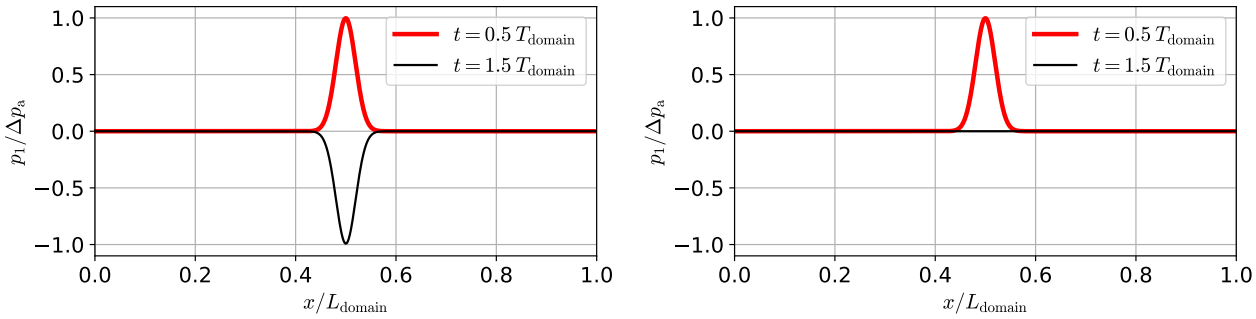


Figure 4.1: Instantaneous wave profile obtained for the default settings (empty `run.DNA` file).



(a) Scattering boundary

(b) Absorbing boundary

Figure 4.2: Illustration of the `scattering` (b) and the `absorbing` (b) boundary condition. A pulse travels from left to right (red lines). In (a), the pulse is reflected at the East domain boundary and travels back to the left (black line). In (b), the pulse passes the domain so that no reflection is visible (black line).

```
BoundaryConditionWest scattering
```

With the `scattering` condition at the East boundary, the pulse is reflected at the boundary as indicated by the black line in Fig. 4.2 (a). In Fig. 4.2 (b), the settings are:

```
BoundaryConditionEast absorbing
BoundaryConditionWest scattering
```

With the `absorbing` condition at the East boundary, the pulse leaves the domain without reflection as shown by the black line in Fig. 4.2 (b). The settings for the West boundary are arbitrary as this boundary is the emission node in this case.

4.3 Shock-driven attenuation

This example is found in `examples/ShockDrivenAttenuation`.

This test case is concerned with the progressive deformation, shock formation, and shock-driven attenuation of an initially sinusoidal wave. The background medium is quiescent and the computational domain is at rest throughout the entire simulation. The CFL number is 0.05 with a spatial resolution of $N_{\text{ppw}} = 400$ points per wavelength λ_0 and a corrector weight of $\gamma = 1.5$, set as follows:

```
FluidNonLinearity 3.5
```

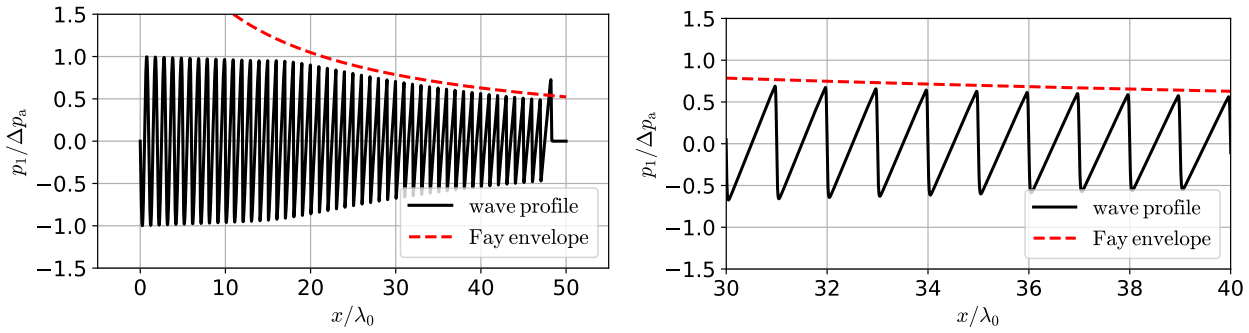
This rather high resolution is needed in order avoid excessive numerical dissipation due to the presence of the steep wavefront. The wave is emitted at the left boundary and travels to the right. The excitation amplitude and the nonlinearity coefficient β are set in such a way that the nominal shock formation distance, given by [1]

$$x_{\text{sh}} = \frac{\rho_0 c_0^3}{2\pi\beta f_{\text{a}} \Delta p_{\text{a}}}, \tag{4.1}$$

takes the value $x_{\text{sh}} = 10$. At the shock formation distance, the progressively steepening wavefront first starts to develop a discontinuity. The red dashed line in Fig. 4.3 represents the envelope $\widehat{p}_1$ owing to the Fay solution [3], given by [1]

$$\frac{\widehat{p}_1}{\Delta p_{\text{a}}} = \frac{\pi}{1 + x/x_{\text{sh}}}. \tag{4.2}$$

The Fay solution is not applicable to the close vicinity of the emitter, but asymptotically approaches the wave profile decay at distances larger than approximately $3x_{\text{sh}}$, where the initially sinusoidal has



(a) Decaying shock amplitude

(b) Detail of the saw-tooth profile

Figure 4.3: Initially sinusoidal and progressively steepening and shock-forming wave. As the wave has traveled past the shock formation distance $x_{\text{sh}} = 10\lambda_0$ (see Eq. (4.1)) in subfigure (a), its amplitude starts to decay rapidly, asymptotically approaching the envelope of Fay solution [3] as given by Eq. (4.2) [1]. Subfigure (b) depicts a detail of the fully developed saw-tooth profile.

fully evolved into a saw-tooth shape. Fig. 4.3 (a) shows that the wave profile exhibits only a minor decay prior to the shock formation distance, after which starts to approach the Fay envelope. Fig. 4.3 (b) shows a detail of the fully developed saw-tooth pattern. It is noted that the predictor-corrector method drives the attenuation associated with the dissipation across the shock front. See Schenke *et al.* [8] for further discussions of this effect.

4.4 Moving emitter, boundary and flow

The following examples are found in `examples/MovingBoundary`.

4.4.1 Moving emitter

This example is found in `examples/MovingBoundary/RelativeEmitterMotion`.

This example illustrates how the Doppler shift due to the motion of an emitter relative to a quiescent medium can be reproduced with `Wave-DNA`. A sine wave is excitated at the left (West) domain boundary and travels from left to right. The background flow field is quiescent:

```
BackgroundMotionMode quiescent
```

The speed of sound is:

```
SoundSpeed 1500.0
```

The moving boundary moves at a constant velocity magnitude of 500 m/s $= c_0/3$, where the setting

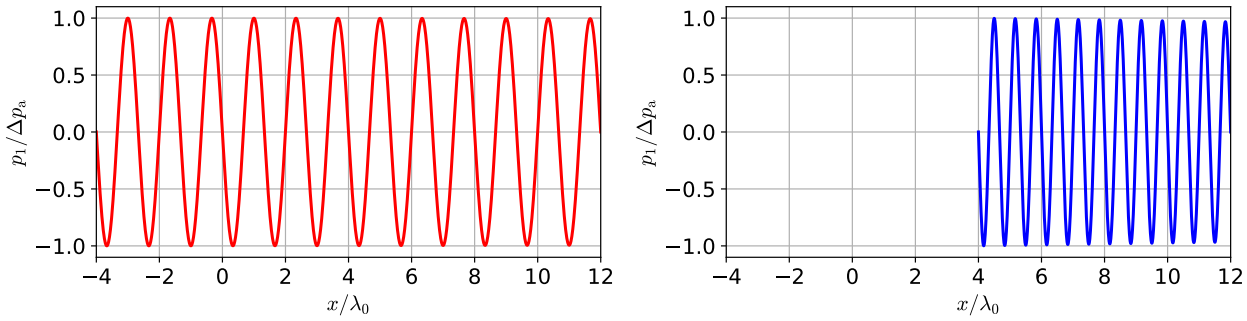
```
BoundaryMotionType linear
```

is used to indicate a constant velocity of the boundary. The excitation location at the left boundary (default) is specified by:

```
ExcitationNode 0
```

The initial position of the moving boundary is $x = 0$ (default) and the fixed boundary is located at $x = 0.18$ m (12 times the unmodulted waveength λ_0), specified as follows:

```
InitialMovingBoundaryPosition 0.0
FixedBoundaryPosition 0.18
```



(a) Red-shifted wave

(b) Blue-shifted wave

Figure 4.4: Doppler-shift induced by an emitter moving relative to a quiescent medium: the wave is excited at the moving boundary. The wave emitting boundary starts at $x = 0$ and moves in negative x-direction in (a), thereby red-shifting the emitted wave, and in positive x-direction in (b), thereby blue-shifting the emitted wave. The magnitude of the boundary velocity is $c_0/3$.

Hence, the moving boundary acts as a wave emitting boundary so that the acoustic wave is Doppler-shifted upon emission. In Fig. 4.4 (a), the wave emitting boundary moves from right to left, hence against the direction of wave propagation, by setting:

```
MovingBoundaryVelocityAmplitude -500
```

This causes a red-shift (frequency decrease) of the emitted wave. In Fig. 4.4 (b), the wave emitting boundary moves from left to right, hence in the direction of wave propagation, by setting:

```
MovingBoundaryVelocityAmplitude 500
```

This causes a blue-shift (frequency increase) of the emitted wave.

4.4.2 Induced flow field

This example is found in `examples/MovingBoundary/InducedFlow`.

Fig. 4.5 shows how an acoustic wave is Doppler-shifted by a temporaily constant and spatially uniform background flow field in one-dimensional Cartesian coordinates. In both Figs. 4.5 (a) and (b), the left domain boundary moves to the right with a velocity of $c_0/3$, and in both cases it induces a background flow field of the same velocity and direction by setting:

```
BackgroundMotionMode coupledToMovingBoundary
```

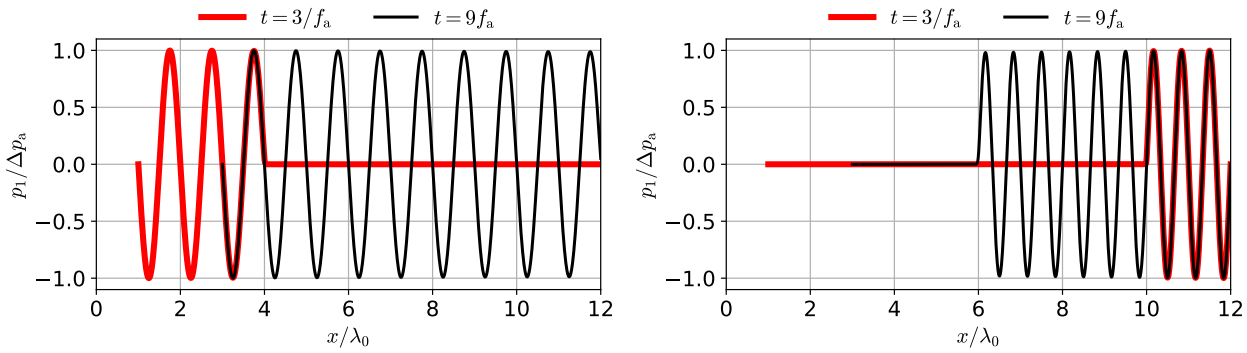
In Fig. 4.5 (a), the wave is excited at the moving boundary, which is initially at $x = 0$ by setting:

```
ExcitationNode 0
```

As a result, the wave travels from left to right and there is no relative motion between the wave emitter and the background flow field and the emitted wavelength λ_0 is not modulated. The instantaneous wave profile in Fig. 4.5 is shown at two time instances in order to illustrate the variable position of the wave emitting boundary. In Fig. 4.5 (b), the wave is excited at the fixed right boundary by setting:

```
ExcitationNode 4800
```

Hence, the wave travels from right to left and the fixed wave emitting boundary is in relative motion to the encountered background flow field. This causes a blue-shift of the emitted wavelength.



(a) Flow inducing and wave emitting boundary

(b) Flow moving relative to fixed emitter

Figure 4.5: Illustration of a flow-induced Doppler shift. The left domain boundary is initially at $x = 0$ and moves from left to right with a velocity of $c_0/3$. At the same time, the moving boundary induces a temporarily and spatially constant background flow field. In (a), the wave is excited at the moving boundary and travels from left to right. As there is no relative motion between background medium in (a), the emitted wavelength is not modulated. In (b), the wave is excited at the right boundary and travels from right to left, so that the wave emitting boundary is in relative motion to the encountered background flow. This causes a blue-shift of the emitted wave.

4.5 Possible combinations of moving boundaries, flow fields and emitters

In `examples/MovingBoundary/PossibleCombinations`, the possible combinations of setting up moving boundaries wave-emitting or non-emitting boundaries and quiescent or induced moving flow fields are demonstrated. Table 4.1 illustrates the hierarchy of possible configurations.

The initial positions of the moving and the fixed boundary determines, which of them is identified as the West and the East boundary, where the boundary further to the left is the West boundary. The boundary condition (either `scattering` or `absorbing`) is assigned to the West and East boundaries, respectively (`BoundaryConditionWest` or `BoundaryConditionEast`). Irrespective of which of the two domain boundaries is the moving one, any of the two boundaries can be the wave-emitting boundary. It is recalled from Sec. 2.4 that the wave-emitting boundary with instantaneous position $R(t)$ is alawys associated with grid node 0, whereas the fixed boundary with position R_{stat} is always associated with grid node $N_{\text{points}} - 1$. Finally, for each of the four configurations of moving vs fixed and emitting vs non-emitting boundaries, the moving boundary can be a flow inducing one or move relative to a quiescent background flow field. This gives the eight different configurations as listed in Table 4.1.

Each of the test cases in this example directory is set up in such a way that waves move past the domain domain boundary. Hence, the test cases also illustrate how the wave-absorbing boundary condition is applied in these configurations.

4.6 Slowly oscillating emitter and flow field

The example of a slowly oscillating emitter is concerned with a slowly oscillating domain boundary, where "slow" means that the frequency f_b of the sinusoidal motion of the domain boundary is smaller than the emitted wave frequency f_a. In this example, a spherically symmetric wave is considered, where the settings for the background flow field are:

```
Geometry 3dSphericallySymmetric
BackgroundMotionMode quiescent
```

Moving West boundary	Emitting West boundary (`ExcitationNode` 0)	Quiescent flow
		Induced flow
	Emitting East boundary (`ExcitationNode` $N_{\text{points}} - 1$)	Quiescent flow
		Induced flow
Moving East boundary	Emitting West boundary (`ExcitationNode` $N_{\text{points}} - 1$)	Quiescent flow
		Induced flow
	Emitting East boundary (`ExcitationNode` 0)	Quiescent flow
		Induced flow

Table 4.1: Eight possible configurations of moving vs fixed, emitting vs non-emitting, and flow inducing vs relatively moving (quiescent flow) boundaries.

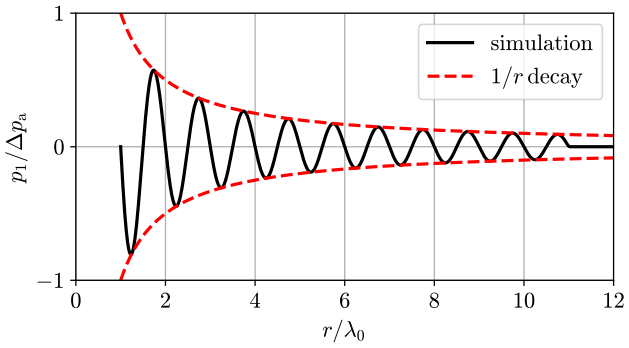


Figure 4.6: spherically symmetric wave decaying like $1/r$.

```
BackgroundVelocityScalingFactor 2.0
```

for a quiescent background medium to which the wave-emitting boundary is in relative motion or

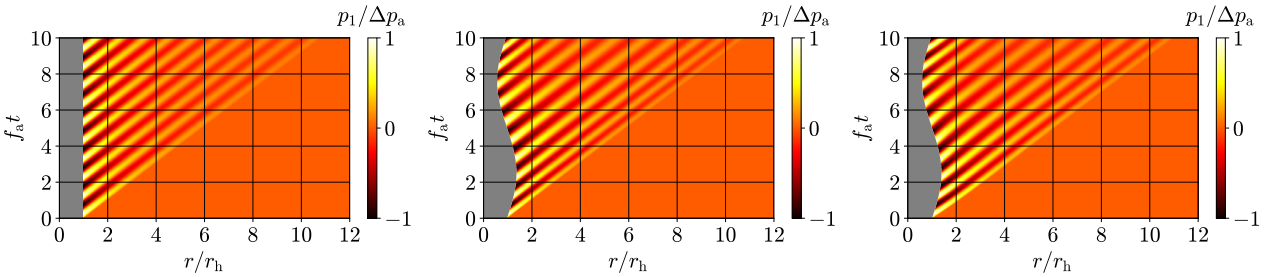
```
BackgroundMotionMode coupledToMovingBoundary
```

for the case where the moving boundary displaces the fluid. The settings for the oscillatory boundary motion are:

```
BoundaryMotionType oscillating
BoundaryMotionStartTime 0.0e-0
BoundaryMotionEndTime 2.3e10
InitialMovingBoundaryPosition 0.015
FixedBoundaryPosition 0.195
MovingBoundaryVelocityAmplitude 375
MovingBoundaryFrequency 1e4
```

Hence, the boundary oscillates around an initial radius of $R_0 = 0.015\,\text{m}$, which corresponds to one wavelength with an emitted wave frequency of $f_\text{a} = 100\,\text{kHz}$ (ten times larger than the frequency the boundary motion) and a speed of sound of $c_0 = 1500\,\text{m/s}$. With a velocity amplitude of 375 m/s of the boundary motion, the maximum Mach number is 0.25.

For reference, Fig. 4.6 shows the instantaneous wave profile for a resting wave-emitting boundary (`MovingBoundaryVelocityAmplitude 0`), where the wave decays like $1/r$ as indicated by the red dashed line. Figs. 4.7 (a), (b), and (c) show the space-time diagrams for the resting, the relatively moving, and the flow-inducing moving boundaries, respectively. In the latter case, the characteristics posses a slight curvature due the variable background flow field in which the acoustic wave propagates.



(a) Fixed wave-emitting boundary (b) Oscillating wave-emitting boundary in a quiescent medium (c) Oscillating flow-inducing wave-emitting boundary

Figure 4.7: Space-time diagrams of an oscillating wave-emitting boundary, where ten acoustic wave periods are emitted during one oscillation cycle of the moving boundary.

4.7 Acoustic black and white holes

The following examples show how `Wave-DNA` can be used to simulate the acoustic wave propgation in acoustic black and white hole configurations [7]. The individual test case folders, containing illustrative acoustic black and white hole examples, include python postprocessing scripts in order to produce space-time diagrams, the plots of the instantaneous wave profiles, and the acoustic pressure evolution at the sonic horizon.

It is important to note that rather than representing compressible background flow fields with variable ρ_0 and c_0, the acoustic black/white hole configurations are model systems for acoustic waves propagating in accelerating flow fields of any (also subsonic) velocity magnitude. The reason for the acoustic black/white hole being such a convenient system for the computational study of acoustic waves in accelerating flow fields lies in the fact that they allow to trap a wave characteristic at the point of sonic transition, the sonic horizon, where it is subject to both a constant background flow velocity and a constant background flow acceleration. Due to the non-zero acceleration, the neighbouring characteristics are either converging towards or diverging away from the sonic horizon.

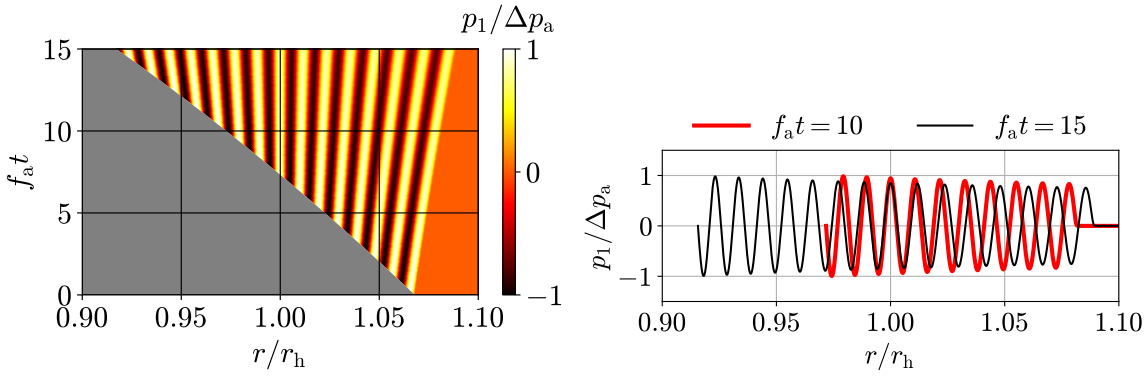
In the acoustic black hole example shown here, nonlinearities are neglected ($\beta = 0$ and $\mathcal{L} = 0$). Fig. 4.8 exemplarily shows (a) the space-time diagram and (b) the instantaneous acoustic pressure profiles for an acoustic black hole. The entire physical domain in Fig. 4.8 is initially located outside the acoustic black hole. Subsequently, the moving wave emitting boundary collapses and eventually passes the sonic horizon while emitting outgoing characteristics. The example in Fig. 4.8 is realized by the following boundary settings:

```
BoundaryMotionType stationaryBlackHole
InitialMovingBoundaryPosition 1.60175
FixedBoundaryPosition 1.75175
```

The stationary sonic horizon radius and the geometry settings are:

```
Geometry 3dSphericallySymmetric
BackgroundMotionMode coupledToMovingBoundary
HorizonRadius 1.5
```

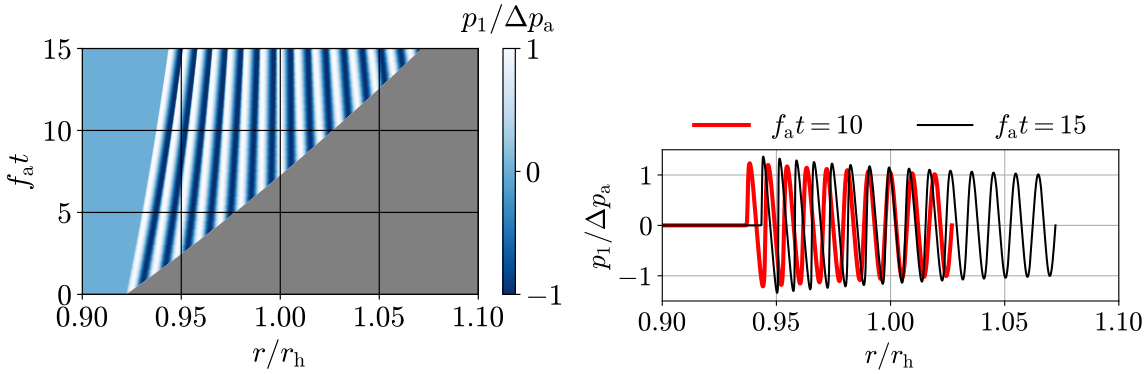
With these settings, the wave emitting boundary radiates a wavetrain that propagates in positive r-direction. At the same time, the spherical moving boundary collapses and induces an inward-directed spherically symmetric background fluid flow that satisfies Eq. (2.3). The boundary satisfies Eq. (2.13) so that a stationary flow field is obtained. The data at the stationary sonic horizon is written to the output file `horizon.dat`. This file can be consulted to examine the evolution of the acoustic pressure and the acoustic (velocity) potential at the sonic horizon.



(a) Space-time diagram

(b) Instantaneous acoustic pressure profile

Figure 4.8: Example of an acoustic black hole.



(a) Space-time diagram

(b) Instantaneous acoustic pressure profile

Figure 4.9: Example of an acoustic white hole.

Fig. 4.9 exemplarily shows (a) the space-time diagram and (b) the instantaneous acoustic pressure profiles for an acoustic white hole, which can be seen as the inverse problem of the acoustic black hole. This time, the entire physical domain is initially located inside the acoustic white hole. Subsequently, the wave emitting boundary expands and eventually passes the sonic horizon while emitting ingoing characteristics. The sonic horizon radius, the speed of sound, and the emitted wave frequency are the same as in Fig. 4.8. The settings to trap the same emitted wave period at the sonic horizon as in the acoustic black hole configuration are as follows:

```
BOUNDARYMOTION
BoundaryMotionType stationaryWhiteHole
InitialMovingBoundaryPosition 1.3824
FixedBoundaryPosition 1.2359
```

Furthermore, Fig. 4.9 shows the results for the nonlinear wave, which exhibits the well-known nonlinear wave-steepening. The settings to achieve the nonlinear wave propagation behavior, in this particular case dominated by the constitutive nonlinearity represented by the nonlinearity coefficient β (`FluidNonLinearity`), are:

```
FluidNonLinearity 3.5
LocalNonlinearity True
```

In the acoustic white hole in Fig. 4.9, the characteristics converge towards the sonic horizon so that the acoustic wave is progressively blue-shifted while the acoustic wave amplitude increases over time.

Bibliography

[1] Blackstock, D. T. (1966). Connection between the Fay and Fubini solutions for plane sound waves of finite amplitude. *The Journal of the Acoustical Society of America*, **39**(6), 1019–1026.

[2] Dey, S. and Dey, C. (1983). An explicit predictor-corrector solver with applications to Burgers' equation: NASA technical memorandum 84402. Technical report, National Aeronautics and Space Administration.

[3] Fay, R. D. (1931). Plane sound waves of finite amplitude. *The Journal of the Acoustical Society of America*, **3**(2A), 222–241.

[4] Mur, G. (1981). Absorbing boundary conditions for the finite-difference approximation of the time-domain electromagnetic-field equations. *IEEE Transactions on Electromagnetic Compatibility*, **EMC-23**, 377–382.

[5] Nascimento, W. C. R. and Pestana, R. C. (2010). An anti-dispersion wave equation based on the predictor-corrector method for seismic modeling and reverse time migration. In *SEG Technical Program Expanded Abstracts*, pages 3226–3230.

[6] Ostashev, V. E., Wilson, D. K., Liu, L., Aldridge, D. F., Symons, N. P., and Marlin, D. (2005). Equations for finite-difference, time-domain simulation of sound propagation in moving inhomogeneous media and numerical implementation. *The Journal of the Acoustical Society of America*, **117**(2), 503–517.

[7] Schenke, S., Sewerin, F., van Wachem, B., and Denner, F. (2022a). Acoustic black hole analogy to analyze nonlinear acoustic wave dynamics in accelerating flow fields. *Physics of Fluids*, **34**(9), 097103.

[8] Schenke, S., Sewerin, F., van Wachem, B., and Denner, F. (2022b). Explicit predictor–corrector method for nonlinear acoustic waves excited by a moving wave emitting boundary. *Journal of Sound and Vibration*, **527**, 116814.

[9] Schenke, S., Sewerin, F., van Wachem, B., and Denner, F. (2023). Amplitude modulation of acoustic waves in accelerating flows quantified using acoustic black and white hole analogues. *Journal of the Acoustical Society of America*, **154**(2), 781–791.