DYNAMIC GRIDS FOR FINITE-DIFFERENCE SCHEMES IN MUSICAL INSTRUMENT SIMULATIONS

Silvin Willemsen
Multisensory Experience Lab
Aalborg University Copenhagen
Copenhagen, Denmark
sil@create.aau.dk

Stefan Bilbao, Michele Ducceschi
Acoustics and Audio Group
University of Edinburgh
Edinburgh, UK

Stefania Serafin
Multisensory Experience Lab
Aalborg University Copenhagen
Copenhagen, Denmark

ABSTRACT

For physical modelling sound synthesis, many techniques are available; time-stepping methods (e.g., finite-difference time-domain (FDTD) methods) have an advantage of flexibility and generality in terms of the type of systems they can model. These methods do, however, lack the capability of easily handling smooth parameter changes while retaining optimal simulation quality and stability, something other techniques are better suited for. In this paper, we propose an efficient method to smoothly add and remove grid points from a FDTD simulation under sub-audio rate parameter variations. This allows for dynamic parameter changes in physical models of musical instruments. An instrument such as the trombone can now be modelled using FDTD methods, as well as physically impossible instruments where parameters such as e.g. material density or its geometry can be made time-varying. Results show that the method does not produce audible artefacts and stability analysis is ongoing.

1. INTRODUCTION

The operation of most musical instruments can be subdivided into excitation and resonator components [1]. Examples of excitation-resonator combinations are the bow and violin and the lips and trumpet. In some instruments, the parameters describing the excitation global search/replace exciter –> excitation are continuously varied by the performer to play the instrument. As an example, the bow velocity, position and bow pressure for stringed instruments, and lip pressure and frequency for brass instruments. Naturally, the resonator is also altered by fingering the strings of the violin or pressing valves on the trumpet to change the instrument pitch. But, even under such variable playing conditions, physical dimensions of the resonators do not change: the string length stays the same and the total tube length remains unchanged; it is only the portions that resonate that are shortened or lengthened.

While the physical dimensions of the instrument do not change, there are several examples where the parameters of the resonator are also modified. A prime example of this is the trombone, where the tube length is dynamically changed in order to generate different pitches. The slide whistle is another example in this category. Guitar strings are another category where the tension can be smoothly modulated during performance using the fretting finger or a whammy bar to create smooth pitch glides. This is odd because you've covered this case in the previous paragraph. I thought this paragraph was about cases where the resonator actually changes size... The same kind of tension modulation is used for the membranes of timpani or "hourglass drums" to change the pitch. It is these direct parameter modifications of the resonators that we are interested in simulating.

In addition to simulating existing instruments, one could potentially simulate instruments that can be manipulated in physically impossible ways. Examples of this could be to dynamically change material properties such as density or stiffness, or even the geometry and size of the instrument where this is physically impossible.

Finite-difference time-domain (FDTD) methods are flexible and generalisable techniques which have recently seen increased use in physical modelling sound synthesis applications [2]though not so recent anymore... The normal approach, for a given system such as a musical instrument, described by a set of partial differential equations (PDEs), is to first represent the instrument over a spatial grid, and then develop a time-stepping method, yielding a fully discrete approximation to the target PDE system.

In many cases, the system itself is static, so that the defining parameters do not change over time. In others, such as the trombone and others mentioned above, this is not the case, and various technical challenges arise when trying to design a simulation using FDTD methods; all relate to the choice of the spatial grid. For example, the grid density is usually closely tied to the parameters themselves through a stability condition. Also, adding and removing points from the grid is nontrivial and can cause audible artefacts and new stability concerns. The default approach of defining a grid globally, according to a very conservative stability condition, as done in [3], is possible, but introduces numerical dispersion and bandlimiting effects. Full-grid interpolation [2, Ch. 5] could be used to change between grid configurations, but extremely high sample rates are necessary to avoid audible artefacts and low-passing effects, rendering any implementation offline.

In this paper, a new method is proposed, allowing the efficient and smooth insertion and deletion of grid points from 1D finite-difference grids to allow for dynamic parameter changes. We are interested in varying parameters 'slowly' (i.e., at sub-audio rate corresponding to human gestural control). In a companion paper we present a physical model of the trombone using the method proposed in this paper [4]. Notice that other techniques do allow for dynamic parameter changes but come with their own drawbacks [2]. Examples of dynamic parameters using modal synthesis [5] are shown in [6, 7] and digital waveguides [8] in [9]. maybe another (better) one here

This paper is structured as follows: Section 2 presents the 1D wave equation, to be used as an illustrative example for the proposed method. Section 3 gives an introduction to numerical meth-

ods, stability and simulation quality. The proposed method for dynamic grids is then presented in Section 4 and applied to the 1D wave equation. Section 5 shows the results of an analysis performed on the method, which are discussed in Section 6. Finally, concluding remarks and future perspectives are given in 7.

2. CONTINUOUS SYSTEMS

The wave equation is a useful starting point for investigations of time-varying behaviour in musical instruments. In 1D, the wave equation may be written as

$$\frac{\partial^2 q}{\partial t^2} = c^2 \frac{\partial^2 q}{\partial x^2}\,, \tag{1}$$

and is defined over spatial domain $x \in [0, L]$, for length L (in m) and time $t \geq 0$ (in s). c (in m/s) is the wave speed. The dependent variable $q = q(x, t)$ in Eq. (1) may be interpreted as the transverse displacement of an ideal string, or the acoustic pressure in the case of a cylindrical tube. Two possible choices of boundary conditions are

$$q(0, t) = q(L, t) = 0 \quad \text{(Dirichlet)}, \tag{2a}$$

$$\frac{\partial}{\partial x} q(0, t) = \frac{\partial}{\partial x} q(L, t) = 0 \quad \text{(Neumann)}, \tag{2b}$$

and describe 'fixed' or 'free' boundary respectively in the case of an ideal string, and 'open' or 'closed' conditions respectively in the case of a cylindrical acoustic tube.

2.1. Dynamic parameters

In the case of the 1D wave equation, only the wave speed c and length L can be altered (in the case of an acoustic tube, only L is variable, and for a string, c could exhibit variations through changes in tension). If Dirichlet-type boundary conditions Or Neumann! Just not a mixture of Dirichlet and Neumann and the two ends– as in Eq. (2a) – are used, and under static conditions, the fundamental frequency f_0 of vibration can be calculated according to

$$f_0 = \frac{c}{2L}. \tag{3}$$

In the dynamic case, and under slow (sub-audio rate) variations of c or L, Eq. (3) still holds approximately. From Eq. (3), one can easily conclude that in terms of fundamental frequency, halving the length of Eq. (1) is identical to doubling the wave speed and vice versa. Looking at Eq. (1) in isolation, f_0 is the only behaviour that can be changed. One can thus leave L fixed and allow time variation in c, so that $c = c(t)$, which will prove easier to work with in the following sections. This fact can more easily be seen if Eq. (1) is scaled or non-dimensionalised as in [2], where scaled domain $x' = x/L \;\Rightarrow\; x' \in [0, 1]$ and $\gamma = c/L$ such that $f_0 = \gamma/2$. For clarity, however, we will employ a fully dimensional representation here. I changed it to this what do you think?

3. NUMERICAL METHODS

This section will provide a brief introduction to physical modelling using FDTD methods, including details on stability and quality of the simulations based on these methods. In the following, c is assumed constant.

3.1. Discretisation

In FDTD methods, the first step is the definition of a grid. The spatial variable can be discretised using $x_l = lh$ with integer $l \in \{0, \ldots, N\}$. The grid spacing h (in m) is the distance between adjacent grid points, and the total number of points covering the domain, including endpoints, is $N + 1$. Here, integer N describes the total number of intervals between the grid points, and thus the total domain length is $L = Nh$. The temporal variable can be discretised using $t_n = nk$ with positive integer n, time step $k = 1/f_s$ (in s) for sample rate f_s (in Hz). The state variable q can then be approximated using $q_l^n \approxeq q(x = lh, t = nk)$.

The following operators can then be applied to q_l^n to get the following approximations to the derivatives in Eq. (1)

$$\delta_{tt} q_l^n = \frac{1}{k^2}\left(q_l^{n+1} - 2q_l^n + q_l^{n-1}\right) \;\approx\; \frac{\partial^2 q}{\partial t^2}\,, \tag{4a}$$

$$\delta_{xx} q_l^n = \frac{1}{h^2}\left(q_{l+1}^n - 2q_l^n + q_{l-1}^n\right) \;\approx\; \frac{\partial^2 q}{\partial x^2}\,. \tag{4b}$$

Substituting these definitions into Eq. (1) yields the following finite-difference scheme (FDS)

$$\delta_{tt} q_l^n = c^2 \delta_{xx} q_l^n. \tag{5}$$

Expanding the operators as in (4) and solving for q_l^{n+1} yields the following update equation

$$q_l^{n+1} = 2q_l^n - q_l^{n-1} + \lambda^2 \left(q_{l+1}^n - 2q_l^n + q_{l-1}^n\right). \tag{6}$$

which is suitable for direct software implementation. Here,

$$\lambda = \frac{ck}{h} \tag{7}$$

is referred to as the Courant number, constrained by numerical stability conditions, and also has an impact on the quality and behaviour of the simulation. This will be described in detail in Sections 3.2 and 3.3.

In the FDS Suggest chaging FDS globally to FDTD scheme or FD scheme, for consistency.described in Eq. (5), the boundary locations are at $l = 0$ and $l = N$. Substituting these locations into Eq. (6) seemingly introduces the need of grid points outside of the defined domain, namely q_{-1}^n and q_{N+1}^n. These can be referred to as *virtual grid points* and can be accounted for using the boundary conditions in Eq. (2). Discretising these yields

$$q_0^n = q_N^n = 0 \quad \text{(Dirichlet)} \tag{8a}$$

$$\delta_{x\cdot} q_0^n = \delta_{x\cdot} q_N^n = 0 \quad \text{(Neumann)} \tag{8b}$$

where

$$\delta_{x\cdot} q_l^n = \frac{1}{2h}\left(q_{l+1}^n - q_{l-1}^n\right) \;\approx\; \frac{\partial q}{\partial x} \tag{9}$$

is a second-order accurate approximation of the first-order spatial derivative. The Dirichlet condition in (8a) says that the displacements of q at the boundary locations are always 0. In practice, this means that these grid points do not need to be updated and the spatial range of calculation for Eq. (6) then becomes $l \in \{1, \ldots, N-1\}$. If the Neumann condition is used, the boundary points do need to be updated as these are not necessarily 0; rather, their 'slope' is 0. Eq. (8b) can then be expanded to yield defnitions for these virtual grid points

$$q_{-1}^n = q_1^n \quad \text{and} \quad q_{N+1}^n = q_{N-1}^n\,. \tag{10}$$

Now that the full system is described, audio output at sample rate f_s can be drawn from the state q_l^n in Eq. (6) at $0 < l < N$ (when using Dirichlet boundary conditions).

3.2. Stability

Explicit FDTD methods for hyperbolic systems such as the 1D wave equation must necessarily satisfy a stability condition. In the case of the update in Eq. (6) it can be shown – using von Neumann analysis [10] – that the system is stable if

$$\lambda \leq 1, \tag{11}$$

which is referred to as the Courant-Friedrichs-Lewy (CFL) condition. The more closely λ approaches this condition with equality, the higher the quality of the simulation (see Section 3.3) and if $\lambda = 1$, Eq. (6) yields an exact solution to Eq. (1). If $\lambda > 1$ the system will become unstable. Recalling (7), Eq. (11) can be rewritten in terms of grid spacing h to get

$$h \geq ck. \tag{12}$$

This shows that the CFL condition in (11) puts a lower bound on the grid spacing, determined by the sample rate and wave speed. Usually, the following steps are taken to calculate λ:

$$h := ck, \quad N := \left\lfloor \frac{L}{h} \right\rfloor, \quad h := \frac{L}{N}, \quad \lambda := \frac{ck}{h}. \tag{13}$$

In other words, condition (12) is first satisfied with equality and used to calculate number of intervals N. Thereafter, h is recalculated based on integer N and used to calculate λ. The calculation of λ in Eq. (13) can be compactly rewritten as

$$\lambda = \frac{ck}{L} \cdot \left\lfloor \frac{L}{ck} \right\rfloor. \tag{14}$$

The flooring operation causes the CFL condition in (11) to not always be satisfied with equality and results in a reduced simulation quality described in the following section.

3.3. Simulation Quality

Choosing $\lambda < 1$ in Eq. (6) will decrease the simulation quality in two ways. Firstly, it will decrease the maximum frequency that the simulation is able to produce, i.e., it will decrease the bandwidth of the output sound of the system.

By analysing the scheme in Eq. (6), it can be shown that the maximum frequency produced by the system can be calculated using $f_{\text{max}} = f_s \sin^{-1}(\lambda)/\pi$ [2, Chap. 6]. Note that only a small deviation of λ from condition (11) leads to a large reduction in output bandwidth. Secondly, choosing $\lambda < 1$ causes numerical dispersion. Harmonic partials become unnaturally closely spaced at higher frequencies (i.e. spurious inharmonicity increases) as λ decreases, which is generally undesirable.

4. THE DYNAMIC GRID

The time variation of the wave speed c leads to various complications in the simulation framework presented above. First of all, a change in c causes a change in λ according to Eq. (14), affecting the simulation quality and bandwidth. Secondly, and more importantly, a change in c could result in a change in N through Eq. (13). As N directly relates to the number of grid points, this raises questions as to *where* and especially *how* one would add and remove points to the grid according to the now-dynamic wave speed.

We propose a method that allows for a non-integer number of intervals to smoothly change between grid configurations, i.e, the number of grid points. This removes the necessity of the flooring operation in Eqs. (13) and (14), and consequently satisfies the CFL condition in (11) with equality at all times. Introducing fractional number of intervals $\mathcal{N}$, where $N = \lfloor \mathcal{N} \rfloor$, Eq. (3) can be rewritten in terms of $\mathcal{N}$ by substituting the calculation of N from (13) into Eq. (3) (using $h = ck$) yielding

$$f_0 = \frac{1}{2\mathcal{N}k} \quad \text{with} \quad \mathcal{N} = L/h. \tag{15}$$

This shows that if $\lambda = 1$, $\mathcal{N}$ solely determines the fundamental frequency of the simulation.

Ideally, a method that dynamically changes the grid size of a FDS should

- r1. generate an output with a fundamental frequency f_0 which is proportional to the wave speed c ($f_0 \propto c$),
- r2. allow for a fractional number of intervals $\mathcal{N}$ to smoothly (without audible artefacts) transition between different grid configurations,
- r3. generate an output containing $N - 1$ modes which are integer multiples of the fundamental ($f_p = f_0 p$ with integer p),
- r4. work in real time to have a playable simulation.

These requirements will be used in Section 6 to evaluate the proposed method.

4.1. Proposed Method

In the following, the location of a grid point (in m from the left boundary) q_l at time index n is denoted by $x_{q_l}^n$. Furthermore, some variables are now time dependent as indicated by superscript n. These are c^n, h^n, $\mathcal{N}^n$, N^n and f_0^n.

4.1.1. System Setup

Consider two grid functions, $u_{l_u}^n$ and $w_{l_w}^n$ defined over discrete domains $l_u \in \{0, \ldots, M^n\}$ and $l_w \in \{0, \ldots, M_w^n\}$ respectively with integers $M^n = \lceil 0.5N^n \rceil$ with $\lceil \cdot \rceil$ denoting the ceiling operation and $M_w^n = \lfloor 0.5N^n \rfloor$, i.e., half the number of points allowed by the stability condition, plus one for overlap. The two grid functions are assumed to lie adjacent to each other on the same domain x. For now, the grid locations $l_u = M^n$ Actually l_u^n, right? hmm.. no I don't think so, as l_u^n is only an element of a set (that changes size). (Also see the email I sent you.) I elaborated right below Eq. (22) that the n in M^n is unaffected by δ_{tt} and $l_w = 0$ are assumed to overlap so that $x_{u_{M^n}}^n = x_{w_0}^n = M^n h^n$, and are referred to as the inner boundaries. The grid locations $l_u = 0$ and $l_w = M_w^n$ are placed at $x_{u_0}^n = 0$ and $x_{w_{M_w^n}}^n = L$ and will be referred to as the outer boundaries. See Figure 1a. The following boundary conditions are then imposed:

$$u_0^n = w_{M_w^n}^n = 0, \quad \text{(Dirichlet)} \tag{16a}$$

$$\delta_{x\cdot} u_{M^n}^n = \delta_{x\cdot} w_0^n = 0. \quad \text{(Neumann)}, \tag{16b}$$

Also need to use M^n in the subscript..? If so, I'd like to mention somewhere here that this is suppressed for brevity..I think you

need to maintain the n everywhere here...otherwise it starts to get very confusing! Like in (16) above. In other words, grid points at the outer boundaries are fixed, according to the usual Dirichlet condition, and those at the inner boundaries are free. ***→It is important to note that the Neumann condition is just used as a starting point for the method here, but will be modified in Section 4.1.2. ←***OK, but then why indicate it at all? Can you not just say that the boundary conditions at the inner boundaries will be set shortly? The problem here is that if you impose these conditions, you have something kind of unphysical—string slope is always zero at the string midpoint! And then you need to undo this later... The systems can then be connected at the inner boundaries using a rigid connection

$$u^n_{M^n} = w^n_0, \quad \text{if } x^n_{u_{M^n}} = x^n_{w_0}. \tag{17}$$

Notice that this condition only needs to be satisfied when the inner boundaries perfectly overlap, which is not always the case when c^n is varied (see Section 4.1.2).

To sum up, a grid function with N intervals as per Eq. (13) is divided into two separate subsystems connected at their respective inner boundaries.

I added some ***→<text> ←*** as additions to clarify the fact that I'm replacing the Neumann condition (and rigid connection) with the method. There is one right below (16b) and two below (26b).

With the above boundary conditions imposed, the following state vectors can be defined:

$$\mathbf{u}^n = [u^n_1, \ldots, u^n_{M^n}]^T, \text{ and } \mathbf{w}^n = [w^n_0, \ldots, w^n_{M^n_w - 1}]^T, \tag{18}$$

with T denoting the transpose operation, and have M^n and M^n_w points respectively. Note that the grid function values at the outer boundaries are excluded as they are 0 at all times due to the Dirichlet boundary condition in (8a). A vector concatenating (18) is then defined as

$$\boldsymbol{\mathcal{U}}^n = \begin{bmatrix} \mathbf{u}^n \\ \mathbf{w}^n \end{bmatrix}. \tag{19}$$

OK, so this big concatenated vector is also changing size at each time step? It changes size only when a point is added and removed, but yes, the maximum speed that this can happen for is 1 element added/removed per time step (but this too fast and will cause artefacts on its own).

Even though the new system has an extra (overlapping) grid point, the behaviour of the new system should be identical to that of the original system. That this holds will be shown below.

Using $u^n_{l_u}$ and $w^n_{l_w}$ in the context of the 1D wave equation, a system of FDSs can be defined as

$$\begin{cases} \delta_{tt} u^n_{l_u} = (c^n)^2 \delta_{xx} u^n_{l_u} + J_u(x^n_{u_{M^n}}) F^n \\ \delta_{tt} w^n_{l_w} = (c^n)^2 \delta_{xx} w^n_{l_w} - J_w(x^n_{w_0}) F^n \end{cases} \tag{20}$$

OK, here is where things start to get nasty. You have $\delta_{tt} u^n_{l_u}$ appearing. But really this should be $\delta_{tt} u^n_{l^n_u}$, shouldn't it? And then how does the operator δ_{tt} work in this case? Like, if l^n_u is changing from one time step to the next? with spreading operators

$$J_u(x^n_i) = \begin{cases} \frac{1}{h^n}, & l_u = \lfloor x^n_i / h^n \rfloor \\ 0, & \text{otherwise} \end{cases} \quad \text{and}$$
$$J_w(x^n_i) = \begin{cases} \frac{1}{h^n}, & l_w = \lfloor x^n_i / h^n \rfloor - M^n \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

Also n superscripts for $l_{u,i}$ and $l_{w,i}$ here? applying the effect of the connection F^n (in $\mathrm{m}^2/\mathrm{s}^2$) to grid points $u^n_{M^n}$ and w^n_0 respectively. Expanding the spatial operators in system (20) at the inner boundaries, recalling the Neumann condition in (16b) and the definition for the virtual grid points needed for this condition in Eq. (10) yields

$$\begin{cases} \delta_{tt} u^n_{M^n} = \frac{\lambda^2}{k^2}(2u^n_{M^n-1} - 2u^n_{M^n}) + \frac{1}{h^n} F^n \\ \delta_{tt} w^n_0 = \frac{\lambda^2}{k^2}(2w^n_1 - 2w^n_0) - \frac{1}{h^n} F^n. \end{cases} \tag{22}$$

It is important to note that the time index n in M^n will not be affected by the δ_{tt} operator and all obtained terms after expansion (Eq. (4a)) will use the same value for M^n. Because of the rigid connection in (17), it is also true that $\delta_{tt} u^n_{M^n} = \delta_{tt} w^n_0$ if $x^n_{u_{M^n}} = x^n_{w_0}$, and F^n can be calculated by setting the right side of the equations in (22) equal to each other:

$$\frac{\lambda^2}{k^2}(2u^n_{M^n-1} - 2u^n_{M^n}) + \frac{1}{h^n} F^n = \frac{\lambda^2}{k^2}(2w^n_1 - 2w^n_0) - \frac{1}{h^n} F^n,$$
$$F^n = h^n \frac{\lambda^2}{k^2}(w^n_1 - u^n_{M^n-1}).$$

Substituting this into system (22) after expansion of the second-time derivative yields the update of the inner boundaries

$$\begin{cases} u^{n+1}_{M^n} = 2u^n_{M^n} - u^{n-1}_{M^n} + \lambda^2(u^n_{M^n-1} - 2u^n_{M^n} + w^n_1) & \text{(23a)} \\ w^{n+1}_0 = 2w^n_0 - w^{n-1}_0 + \lambda^2(u^n_{M^n-1} - 2w^n_0 + w^n_1) & \text{(23b)} \end{cases}$$

which, (again, recalling Eq. (17)) are indeed equivalent expressions for the connected point which is necessary to satisfy the rigid connection. System (20) can be shown to exhibit behaviour identical to that of the original system. In (23), w^n_1 in Eq. (23a) acts as virtual grid point $u^n_{M^n+1}$ and $u^n_{M^n-1}$ in (23b) as virtual grid point w^n_{-1}. This important fact is what the method relies on and will be extensively used in the following.

OK, in the last panel of Figure 1, the distance between u_M and w_0 looks like a full grid point...but is labelled as $\alpha = 1/2$, which I thought would be a half-grid point distance. It is! A gridpoint distance (h) is between two blue points or between two red points. This makes the distance between a blue and a red point $0.5h$ or $\alpha = 0.5$. I could make it 0.4 or 0.6 to avoid this confusion..

4.1.2. Changing the Grid

The previous section describes the case in which $\mathcal{N}^n$ is an integer and condition (12) is satisfied with equality. We now continue by varying c^n such that this is not the case.

The locations of the outer boundaries $x^n_{u_0}$ and $x^n_{w_{M_w}}$ are fixed:

$$x^n_{u_0} = x^0_{u_0} = 0 \quad \text{and} \quad x^n_{w_{M^n_w}} = x^0_{w_{M^n_w}} = L \quad \forall n.$$

If the wave speed c^n is then decreased, and consequently the grid spacing h^n according to Eq. (12) (with equality), all other points move towards their respective outer boundary (see Figure 1b). Calculating h^n this way allows this method to always satisfy the CFL condition in Eq. (11) with equality, solving issues regarding simulation quality and numerical dispersion described in Section 3.3.

As mentioned in Section 4.1.1, the state of the virtual grid points at the inner boundaries are defined as $u^n_{M^n+1} = w^n_1$ and $w^n_{-1} = u^n_{M^n-1}$ when the inner boundaries perfectly overlap (i.e., $x^n_{u_M} = x^n_{w_0}$). If this is not the case ($x^n_{u^n_M} \neq x^n_{w_0}$) a Lagrangian

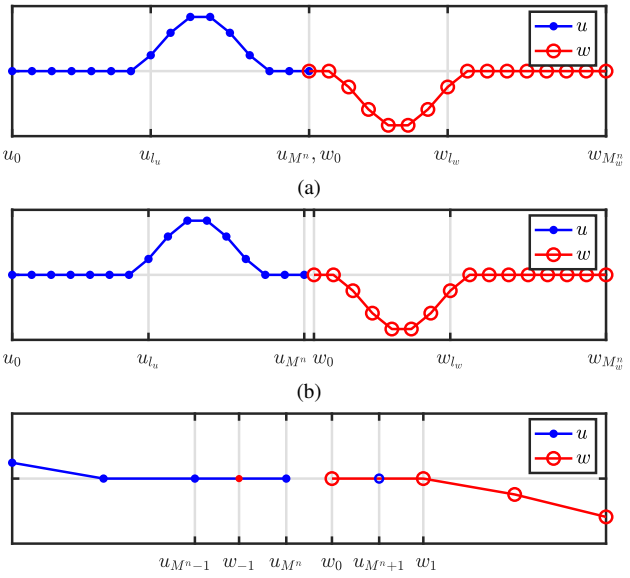

Figure 1: *Illustration of the proposed method. In all figures, the x-axis shows the location of the respective grid points, but 'x^n' is omitted for brevity. (a) Locations of the states of two (1D wave) systems connected at the inner boundaries ($\mathcal{N}^n = 30$, $x^n_{u_{M^n}} = x^n_{w_0}$). (b) When c^n – and consequently h^n – are decreased and the positions of the grid points change ($\mathcal{N}^n = 30.5$, $x^n_{u_{M^n}} \neq x^n_{w_0}$). (c) Figure 1b zoomed-in around the inner boundaries. The virtual grid points $u^n_{M^n+1}$ and w^n_{-1} are shown together with the distance between them expressed using α in Eq.* (24).

interpolator $I(x_i^n)$ at location x_i^n (in m from the left boundary) can be used to calculate the value of these virtual grid points (also see Figure 1c for reference). The interpolator I is a row-vector with the same length as $\boldsymbol{\mathcal{U}}^n$ (from Eq. (19)) and its values depend on the interpolation order. In the following, the fractional part of $\mathcal{N}^n$ is defined as

$$\alpha = \alpha^n = \mathcal{N}^n - N^n \tag{24}$$

keeping α without superscript for brevity by defining it like this (unless you think for consistency it should have the superscript anyway) and for clarity, I and $\boldsymbol{\mathcal{U}}^n$ are indexed by m. Now, consider the following quadratic interpolator

$$I_2(x_i^n) = \begin{cases} -(\alpha-1)/(\alpha+1), & m = m_i^n - 1 \\ 1, & m = m_i^n \\ (\alpha-1)/(\alpha+1), & m = m_i^n + 1 \\ 0, & \text{otherwise} \end{cases} \tag{25a}$$

and its flipped version

$$I_2^{\leftarrow}(x_i^n) = \begin{cases} (\alpha-1)/(\alpha+1), & m = (m_i^{\leftarrow})^n - 1 \\ 1, & m = (m_i^{\leftarrow})^n \\ -(\alpha-1)/(\alpha+1), & m = (m_i^{\leftarrow})^n + 1 \\ 0, & \text{otherwise} \end{cases} \tag{25b}$$

with $m_i^n = \lfloor x_i^n/h^n \rfloor$ and $(m_i^{\leftarrow})^n = \lfloor x_i^n/h^n + (1-\alpha) \rfloor$ m_i's also with superscript n here?, where the shift in the latter is necessary to transform the location x_i^n to the right indices of $\boldsymbol{\mathcal{U}}^n$. When applied to Eq. (19) this yields the definitions for the virtual grid points

$$u^n_{M^n+1} = I_2^{\leftarrow}(x^n_{u_{M^n+1}})\boldsymbol{\mathcal{U}}^n = \frac{\alpha-1}{\alpha+1}u^n_{M^n} + w^n_0 - \frac{\alpha-1}{\alpha+1}w^n_1, \tag{26a}$$

$$w^n_{-1} = I_2(x^n_{w_{-1}})\boldsymbol{\mathcal{U}}^n = -\frac{\alpha-1}{\alpha+1}u^n_{M^n-1} + u^n_{M^n} + \frac{\alpha-1}{\alpha+1}w^n_0. \tag{26b}$$

→ These definitions for the virtual grid points at the inner boundaries will replace the Neumann condition in Eq. (16b). ← One can show that when $\mathcal{N}^n$ is an integer, and thus $\alpha = 0$, Eqs. (26a) and (26b) can be substituted as w^n_1 and $u^n_{M^n-1}$ into Eqs. (23a) and (23b) respectively (as these acted as virtual grid points $u^n_{M^n+1}$ and w^n_{-1}). Then recalling Eq. (17) it can be seen that the system reduces to (23) and exhibits the same exact behaviour as the normal case.

***→Now that the virtual grid points at the inner boundaries are not determined by the Neumann boundary condition in (16b), but rather by the definitions in Eqs. (26), system (20) can simply be re-written to

$$\begin{cases} \delta_{tt}u^n_{l_u} = (c^n)^2\delta_{xx}u^n_{l_u} \\ \delta_{tt}w^n_{l_w} = (c^n)^2\delta_{xx}w^n_{l_w} \end{cases} \tag{27}$$

where the Dirichlet condition in (16a) is (still) used for the outer boundaries and the Neumann condition at the inner boundaries in (16b) is replaced by the definitions in (26)

$$u^n_{M+1} = I_2^{\leftarrow}(x^n_{u_{M^n+1}})\boldsymbol{\mathcal{U}}^n \quad \text{and} \quad w^n_{-1} = I_2(x^n_{w_{-1}})\boldsymbol{\mathcal{U}}^n. \tag{28}$$

←***

4.1.3. Adding and Removing Grid Points

When c^n, and consequently h^n, are decreased and the inner boundary points surpass the virtual points (i.e. $x^n_{u_M} \leq x^n_{w_{-1}}$ and $x^n_{w_0} \geq x^n_{u_{M+1}}$), which means that $N^n > N^{n-1}$, a point is added to the right boundary of u and the left boundary of w (for both time indices n and $n-1$) in an alternating fashion:

$$\begin{cases} \mathbf{u}^n = [(\mathbf{u}^n)^T, I_3\mathbf{v}^n]^T & \text{if } N^n \text{ is odd,} \\ \mathbf{w}^n = [I_3^{\leftarrow}\mathbf{v}^n, (\mathbf{w}^n)^T]^T & \text{if } N^n \text{ is even,} \end{cases} \tag{29}$$

where

$$\mathbf{v}^n = [u^n_{M-1}, u^n_M, w^n_0, w^n_1]^T,$$

and cubic Lagrangian interpolator

$$I_3 = \left[-\tfrac{\alpha(\alpha+1)}{(\alpha+2)(\alpha+3)} \quad \tfrac{2\alpha}{\alpha+2} \quad \tfrac{2}{\alpha+2} \quad -\tfrac{2\alpha}{(\alpha+3)(\alpha+2)} \right], \tag{30}$$

with $I_3^{\leftarrow}$ being just a flipped, not shifted, version of (30). See Figure 2. Notice that $\mathcal{N}^n$ is only going to be slightly bigger than an integer at the moment that a point is added (except in the case of extremely quick parameter variations) and Eq. (24) will return $\alpha \gtrsim 0$. This means that that $I_3 \approx [0, 0, 1, 0]$ and the displacement of the newly added point is nearly fully based on the grid point at the inner boundary of the other system.

Removing grid points happens when c^n, and consequently h^n, are increased and $x^n_{u_M} \geq x^n_{w_0}$ (or $N^n < N^{n-1}$). Grid points are

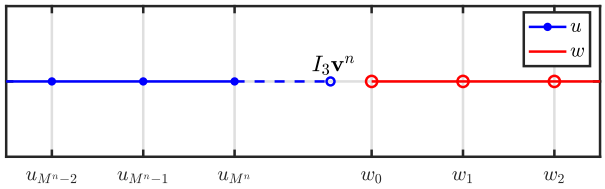

Figure 2: *The moment when a point is added to* $\mathbf{u}$ *at location* $x_{u_{M^n}} + h$ *in Eq.* (29)*. This figure shows an extreme case where this location is far from* x_{w_0}*, i.e.,* $\alpha \not\approx 0$ *in Eq.* (30)*.*

simply removed from $\mathbf{u}$ and $\mathbf{w}$ (again for both n and $n-1$) in an alternating fashion according to

$$\begin{cases} \mathbf{u}^n = [u_0^n, u_1^n ..., u_{M^n-1}^n]^T & \text{if } N^n \text{ is even,} \\ \mathbf{w}^n = [w_1^n, w_2^n ..., w_{M_w^n}^n]^T & \text{if } N^n \text{ is odd.} \end{cases} \tag{31}$$

In Eqs. (29) and (31), the even and odd conditions can be inverted. To keep the difference between u and w a maximum of one grid point, the ceiling and flooring operations when calculating M^n and M_w^n will need to be inverted as well.

Until now, only adding and removing points in the center of the original system has been considered. This location could be moved anywhere along the grid, the limit being one point from the boundary. In other words, both $u_{l_u}^n$ and $w_{l_w}^n$ need to have at least one point (excluding the grid functions at the outer boundaries). Furthermore, one does not have to add and remove points from $\mathbf{u}$ and $\mathbf{w}$ in an alternating fashion as in (29), but can just add and remove from (fx.) $\mathbf{u}$ leaving $\mathbf{w}$ the same size throughout the simulation. In the extreme case where $M^n = N^n - 1$ and $M_w^n = 1$ (leaving $w_{l_w}^n$ with only one moving grid point, w_0^n) the method still works.

4.1.4. Displacement correction

A problem that arises when increasing c^n, is that it is possible that the displacements $u_M^n \not\approx w_0^n$ at the time when a grid point needs to be removed. As the grid locations $x_{u_M}^n \approx x_{w_0}^n$ at the time of removal (except for extremely quick parameter variations), this violates the rigid connection in (17) and causes audible artefacts. A method is proposed that decreases the relative displacement of the inner boundaries the closer their grid-locations are together, i.e., the closer α in (24) is to 0. We thus extend system (27) with an artificial spring force as

$$\begin{cases} \delta_{tt} u_{l_u}^n = (c^n)^2 \delta_{xx} u_{l_u}^n + J_u(x_{u_M}^n) F_\text{c}^n, \\ \delta_{tt} w_{l_w}^n = (c^n)^2 \delta_{xx} w_{l_w}^n - J_w(x_{w_0}^n) F_\text{c}^n. \end{cases} \tag{32}$$

Using centred temporal averaging and difference operators

$$\mu_{t\cdot} q_l^n = \frac{1}{2}\left(q_l^{n+1} + q_l^{n-1}\right), \tag{33a}$$

$$\delta_{t\cdot} q_l^n = \frac{1}{2k}\left(q_l^{n+1} - q_l^{n-1}\right), \tag{33b}$$

the correction effect is defined as

$$F_\text{c}^n = \beta\left(\mu_{t\cdot}\eta^n + \sigma_0 \delta_{t\cdot}\eta^n\right). \tag{34}$$

with the difference in displacement between the inner boundaries

$$\eta^n \triangleq w_0^n - u_M^n, \tag{35}$$

and damping coefficient σ_0. Furthermore, β scales the effect of the displacement correction and is defined as

$$\beta = \beta(\alpha) = \frac{1-\alpha}{\alpha + \varepsilon}, \tag{36}$$

where $\varepsilon \ll 1$ prevents a division by 0. Despite the operators in (33) introducing states at $n+1$, it is possible to calculate the force explicitly (such as in [2] or [11]). Furthermore, it can be shown that even when $\varepsilon = 0$ this calculation is always defined. In that case, as $\alpha \to 0$, $\beta \to \infty$ which acts as a rigid connection such as Eq. (17). Essentially, the displacement correction attempts to have $\eta^n \to 0$ in Eq. (35) as $\alpha \to 0$ to satisfy the rigid connection in Eq. (17). Although the correction presented here is not based on some physical process, it could be justified by the fact that large differences in displacement between two spatially adjacent points is not physical. ← tried to justify

Notice that when c^n is decreased, the rigid connection will not be violated as $u_M^n \approx w_0^n$ when a point is added. This is due to the fact that $I_3 \approx [0, 0, 1, 0]$ and either u_M^n or w_0^n is the newly added point which almost solely based on the other (for low-speed parameter variations).

4.2. Summary

Here, Section 4.1 is summarised and describes the final version of the proposed method.

The proposed method subdivides a grid function q_l^n with N intervals into two grid functions $u_{l_u}^n$ and $w_{l_w}^n$ with M^n and M_w^n intervals respectively for a total of $N^n + 2$ grid points. Knowing that $\lambda = 1\ \forall n$, Eq. (6), written for both grid functions, becomes

$$u_{l_u}^{n+1} = u_{l_u+1}^n + u_{l_u-1}^n - u_{l_u}^{n-1}, \tag{37a}$$

$$w_{l_w}^{n+1} = w_{l_w+1}^n + w_{l_w-1}^n - w_{l_w}^{n-1}. \tag{37b}$$

Due to the Dirichlet boundary condition in (16a) imposed at the outer boundaries of the system, u_0^n and $w_{M_w}^n$ are 0 at all times and do not have to be included in the calculation. The ranges of calculation for Eq. (37a) and (37b) then become $l_u \in \{1, \ldots, M^n\}$ and $l_w \in \{0, \ldots, M_w^n - 1\}$ respectively.

The grid points at the inner boundaries are calculated by expanding (27) (ignoring the displacement correction for now)

$$u_M^{n+1} = u_{M+1}^n + u_{M-1}^n - u_M^{n-1}, \tag{38a}$$

$$w_0^{n+1} = w_{-1}^n + w_1^n - w_0^{n-1}. \tag{38b}$$

where virtual grid points u_{M+1}^n and w_{-1}^n can be calculated using Eq. (26).

Then, when $N^n > N^{n-1}$ a point is added to $\mathbf{u}^n$ and $\mathbf{u}^{n-1}$ (or $\mathbf{w}^n$ and $\mathbf{w}^{n-1}$) using Eq. (29), and when $N^n < N^{n-1}$ a point is removed from the same vectors using Eq. (31). In order to prevent audible artefacts when increasing c^n (and thus decreasing $\mathcal{N}^n$) due to a violation of the rigid connection in (17), a method in (32) is proposed to ensure that the grid points at the inner boundaries have a similar displacement when one of them is removed.

4.3. Implementation

A MATLAB implementation of the proposed method can be found via [12] and Algorithm 1 shows the order of calculation of this implementation. Especially important to take into account, is to only retrieve a change in c^n at time index n once before all other calculations. This is to ensure that $u^n_{l_u}$ and $w^n_{l_w}$ are calculated with the same α and β for all l_u and l_w.

while *application is running* **do**
- Retrieve new c^n
- Calc. h^n (Eq. (12) with equality)
- Calc. $\mathcal{N}^n$ and N^n (Eqs. (15) and (13))
- Calc. α (Eq. (24))
- **if** $N^n \neq N^{n-1}$ **then**
  - Add or remove point (Eq. (29) or (31))
  - Update M^n and M^n_w
- **end**
- Calc. $u^{n+1}_{l_u}$ and $w^{n+1}_{l_w}$ (Eqs. (37) and (38))
- Calc. and apply displacement corr. (Eq. (34))
- Retrieve output
- Update states ($\boldsymbol{\mathcal{U}}^{n-1} = \boldsymbol{\mathcal{U}}^n, \boldsymbol{\mathcal{U}}^n = \boldsymbol{\mathcal{U}}^{n+1}$)
- Update N^{n-1} ($N^{n-1} = N^n$)
- Increment n

end

Algorithm 1: *Pseudocode showing the order of calculations.*

5. ANALYSIS AND RESULTS

5.1. Modes

Writing (32) in matrix form, one can perform a modal analysis while changing c^n to obtain the frequencies and damping coefficients for each mode. As a test case, the wave speed of a system running at $f_s = 44100$ Hz is linearly varied from $c^0 = 2940$ ($\mathcal{N}^0 = 15$) to $c^{n_{\text{end}}} = 2205$ ($\mathcal{N}^{n_{\text{end}}} = 20$) where n_{end} is the simulation length in samples. Grid points are added and removed as close to the right boundary as possible, i.e., $M^n = N^n - 1$ and $M^n_w = 1$ (similar behaviour can be observed if $M^n = 1$ and $M^n_w = N^n - 1$). The results of the analysis are shown in Figure 3a where higher damping (induced by the displacement correction) is indicated using thinner and bluer lines. Figure 3b shows the resulting spectrogram, with the displacement correction deactivated, of the system excited with $u^0_1 = 1$ and the output retrieved at u^n_1, and Figure 3c shows a system with the same excitation but the change in c^n inverted ($\mathcal{N}^n = 20 \rightarrow 15$) and displacement correction activated.

In the following, the lowest mode generated by the analysis is referred to as f^n_1 and should ideally be equal to f^n_0 calculated using Eq. (3). The first thing one can observe from Figure 3a is that the frequencies of the modes decrease as c^n decreases (as desired). The lower the mode, the more linear this decrease happens. Between $\mathcal{N}^n = 15$ and $\mathcal{N}^n = 16$, f^n_1 maximally deviates by -0.15 cents. In this same interval f^n_{15} maximally deviates by -67 cents. This deviation gets less as $\mathcal{N}^n$ increases. Experiments with higher even-ordered Lagrange interpolators show that these frequency deviations become smaller, but not by a substantial amount. The quadratic interpolator has thus been chosen for being simpler and more flexible while not being substantially worse than higher order interpolators.

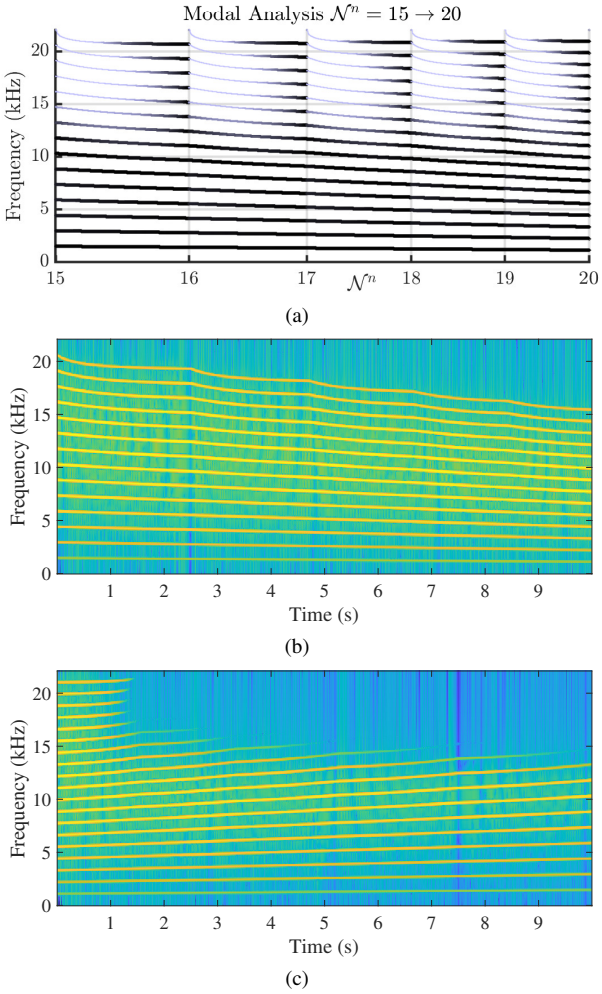

Figure 3: *Experiments showing (linearly) varying wave speed between $c^0 = 2940$ ($\mathcal{N}^0 = 15$) and $c^{n_{\text{end}}} = 2205$ ($\mathcal{N}^{n_{\text{end}}} = 20$) with $M^n = N^n - 1$ and $M^n_w = 1$ running at $f_s = 44100$ Hz for 10 s ($n_{\text{end}} = 10 f_s$). (a) Modal analysis of system* (32)*. Thinner and bluer lines indicate a higher amount of damping. (b) Output of the system while decreasing c^n ($\mathcal{N}^n = 15 \rightarrow 20$) without displacement correction, excited using $u^0_1 = 1$ and retrieved at u^n_1. The sound output follows the same pattern as predicted by the analysis shown in Figure 3a. (c) Output of the system while increasing c^n ($\mathcal{N}^n = 20 \rightarrow 15$) with displacement correction activated (essentially flipping the analysis in Figure 3a along the x-axis).*

Another observation from Figure 3a is that there are always N^n modes present, corresponding to the number of moving points of the system. As can be seen from the spectrum in Figure 3b the highest mode is not excited. If the system is excited when $\mathcal{N}^n$ is not an integer, the highest mode will also be excited. Comparing the implementation of the system using this method with integer $\mathcal{N}^n$ (without changing c^n) to a normal implementation of the 1D wave equation (shown in Section 3) with $N^n = \mathcal{N}^n$, identical outputs are observed, even though the latter has $N^n - 1$ moving

points.

5.2. Displacement Correction

In the experiments, $\sigma_0 = 1$ in Eq. (34). The displacement correction has a low-pass-comb-filtering effect on the system, where the position and amount of notches directly relates to the position of where grid points are added and removed. The best behaviour, i.e., least affecting lower frequencies, is when grid points are added and removed as close to the boundary as possible, i.e., $M^n = N^n - 1$, and only has one notch as shown in Figures 3a and 3c.

5.3. Limit on Rate of Change of c

The current implementation of the proposed method can only add or remove a maximum one point per sample using Eqs. (29) and (31). The speed of decreasing f_0^n according to (15) is thus limited by $|\mathcal{N}^n - \mathcal{N}^{n-1}| \leq 1$. Though this is the maximum limitation on speed, a much lower limitation needs to be placed to keep the system well-behaved. The usual stability and energy analyses performed on FDSs are not valid anymore in the time-varying case. Frozen coefficient analysis as in [10] could be applied here and hold for slowly varying coefficients, but is left for future work.

6. DISCUSSION

To decide whether the proposed method is satisfactory, the results presented in the previous section are compared to the method requirements listed in Section 4.

It can be argued that the frequency deviations of f_1^n from f_0^n are sufficiently small to say that the r1 is satisfied. As for r2, a fractional number of intervals $\mathcal{N}^n$ has been introduced and smooth transitions are indeed observed from Figure 3b, in the case when c^n is decreased and $\mathcal{N}^n$ is increased. When c^n is increased instead, the displacement correction prevents artefacts when grid points are removed as seen in Figure 3c. However, the filtering effect that the displacement correction has on the system (mentioned in Section 5.2) is not ideal as it creates notches in the spectrum of the output sound. The least intrusive filtering happens when points are added and removed as close to the boundary as possible, i.e., when $M^n = 1$ or $M_w^n = 1$ where the notch only occurs in the higher end of the spectrum. Although artefacts do not occur in the experiment shown in Figure 3c, higher speeds of parameter variation might cause artefacts to occur. The value of σ_0 could therefore also be made dynamic and depending on the rate of change of c^n to have a higher effect when c^n is increased faster and vice versa. Either way, as this is still not ideal, another method for reducing artefacts that less affects the frequency content of the system should be devised, if possible.

The modal analysis in Figure 3a shows that the method generates N^n rather than $N^n - 1$ modes as set by r3. However, the output does contain the correct number of modes as shown in Figure 3b due to the highest mode not being excited. This is a result of the rigid connection imposed on the inner boundaries, forcing them to have the same displacement and act as one point. The latter part of r3, however, is not satisfied. The modes deviate from integer multiples of f_0^n, moreso for higher modes. Other interpolation techniques could be investigated to improve the behaviour and decrease this deviation.

Finally, the method only adds a few extra calculations for the inner boundaries so r4 is also easily satisfied.

Although the results bring forward some drawbacks of the proposed method, such as modal frequency deviations, and filtering effects, most of these affect the higher frequencies of the output. First of all, human frequency sensitivity becomes very limited above 3000 Hz [13] making high-frequency deviations much less important perceptually. Secondly, the physical systems one usually tries to model contain high-frequency losses, causing higher modes to usually not have very high amplitudes to begin with. Finally, $\mathcal{N}^n$ is usually much bigger in the systems that one tries to model, with which frequency deviations happen to a much smaller degree.

Finally, as of now, some aspects of the proposed method still lack physical justification (such as the displacement correction), but are shown to yield the desired behaviour and fulfil the aforementioned requirements to a satisfactory degree. Despite this, further work needs to be done to physically justify the choices made in this paper.

7. CONCLUSIONS AND PERSPECTIVES

This paper presents a method to change grid configurations of finite-difference schemes to allow for dynamic parameter changes. The method allows the stability condition that these schemes rely on can be satisfied with equality at all times, minimising numerical dispersion and bandlimiting issues. Grid points are shown to be added and removed smoothly and do not cause artefacts when switching between grid configurations.

The proposed method might not provide an exact solution to the problem of time-varying systems, and not all choices are physically justified, but it does circumvent the need for upsampling and higher orders of computations necessary to approximate this solution.

Although this method has only been applied to the 1D wave equation it could be applied to many other 1D systems. Other parameters, such as material density or stiffness could also be made dynamic, going beyond what is physically possible. An application of the method that could be investigated is that of non-linear systems, such as the Kirchhoff-Carrier string model [14] where the tension is modulated based on the state of the system.

Other future work includes creating an adaptive version of the displacement correction that changes its effect depending on the speed at which the grid is changed. Finally, stability and energy analyses will have to be performed to show the limits on changes in parameters and grid configurations.

8. ACKNOWLEDGMENTS

This work is supported by NordForsk's Nordic University Hub Nordic Sound and Music Computing Network NordicSMC, project number 86892.

9. REFERENCES

[1] G. Borin, G. De Poli, and A. Sarti, "A modular approach to excitator-resonator interaction in physical models syntheses," in *Proceedings of the International Computer Music Conference*, 1989.

[2] S. Bilbao, *Numerical Sound Synthesis*, John Wiley & Sons, United Kingdom, 2009.

[3] S. Willemsen, N. Andersson, S. Serafin, and S. Bilbao, "Real-time control of large-scale modular physical models using the sensel morph," in *Proc. of the 16th Sound and Music Computing Conference*, 2019, pp. 275–280.

[4] S. Willemsen, S. Bilbao, M. Ducceschi, and S. Serafin, "A physical model of the trombone using dynamic grids for finite-difference schemes," submitted to *Proc. of the 23rd Int. Conf. on Digital Audio Effects (DAFx)*, 2021.

[5] J. D. Morrison and J.-M. Adrien, "Mosaic: A framework for modal synthesis," *Computer Music Journal*, vol. 17, no. 1, pp. 45–56, 1993.

[6] S. Mehes, M. van Walstijn, and P. Stapleton, "Towards a virtual-acoustic string instrument," in *Proceedings of the 13th Sound and Music Computing Conference (SMC)*, 2016.

[7] S. Willemsen, S. Serafin, and J. R. Jensen, "Virtual analog simulation and extensions of plate reverberation," in *Proc. of the 14th Sound and Music Computing Conference*, 2017, pp. 314–319.

[8] J.O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, no. 4, pp. 74–91, 1992.

[9] R. Michon and J.O. Smith, "A hybrid guitar physical model controller: The BladeAxe," in *Proceedings ICMC|SMC|2014*, 2014.

[10] J. Strikwerda, *Finite Difference Schemes and Partial Differential Equations*, Wadsworth & Brooks/Cole Advanced Books & Software, Pacific Grove, California, 1989.

[11] S. Bilbao, "A modular percussion synthesis environment," in *Proc. of the 12th Int. Conference on Digital Audio Effects (DAFx)*, 2009.

[12] S. Willemsen, "MATLAB implementation of the Dynamic Grid," Available at https://gist.github.com/SilvinWillemsen/e4a0e52a3df3727c58db09de1bf90e35, accessed April 13, 2021.

[13] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models*, Springer-Verlag, Berlin-Heidelberg, Germany, 1990.

[14] G. F. Carrier, "On the non-linear vibration problem of the elastic string," *Quarterly of Applied Mathematics*, vol. 3, pp. 157–165, 1945.