THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Physical modeling, algorithms and sound synthesis

Citation for published version:
Bilbao, S, Desvages, C, Ducceschi, M, Hamilton, B, Harrison, R, Torin, A & Webb, CJ 2020, 'Physical modeling, algorithms and sound synthesis: The NESS Project', *Computer Music Journal*, vol. 43, no. 2-3, pp. 15-30. https://doi.org/10.1162/COMJ_a_00516

Digital Object Identifier (DOI):
10.1162/COMJ_a_00516

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Computer Music Journal

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

OPEN ACCESS

Download date: 09. May. 2024

The NESS Project: Physical Modeling, Algorithms and Sound Synthesis

Stefan Bilbao[(1)], Charlotte Desvages[(2)], Michele Ducceschi[(1)], Brian Hamilton[(1)], Reginald Harrison-Harsley[(3)], Alberto Torin[(1)], Craig Webb[(4)]

(1): Acoustics and Audio Group/Reid School of Music, University of Edinburgh, Edinburgh, UK

(2): School of Mathematics, University of Edinburgh, Edinburgh, UK

(3): i4 Product Design, Edinburgh UK

(4): Physical Audio Ltd., London, UK

Accepted author manuscript (author's final version), Computer Music Journal, MIT Press, 20 November 2019.

Abstract

Physical modeling synthesis has a long history. As computational costs for physical modeling synthesis are often much greater than in conventional synthesis methods, most techniques currently rely on simplifying assumptions: these include digital waveguides, as well as modal synthesis methods. While such methods are efficient, it can be difficult to approach some of the more detailed behaviour of musical instruments in this way, including strongly nonlinear interactions. Mainstream time-stepping simulation methods, while computationally costly, allow for such detailed modeling. In this article, the results of a five year research project NESS (standing for *Next Generation Sound Synthesis*), are presented, with regard to algorithm design for a variety of sound-producing systems, including brass and bowed string instruments, guitars, and large-scale environments for physical modeling synthesis. In addition, 3D wave-based

modeling of large acoustic spaces is discussed, as well as the embedding of percussion instruments within such spaces for full spatialisation. This article concludes with a discussion of some of the basics of such time stepping methods, as well as issues relevant to their use in audio synthesis applications.

Introduction

Digital sound synthesis has, of course, a long history—too long to relay here in detail, but easily found in standard references (Roads 1996) and the pages of this journal. The most well-known techniques, including additive synthesis, FM, wavetable methods and granular synthesis have reached a certain level of maturity; practitioners of electronic music are familiar with them, and real-time implementations abound.

Physical modeling synthesis is somewhat younger. In principle, the idea is straightforward: beginning from a target system, which is often—but not always—an acoustic instrument or analog electronic device from the real world, develop a physical model, which is invariably a system of equations describing the input $\rightarrow$ system dynamics $\rightarrow$ output chain. From the model, one then proceeds to a discrete-time simulation algorithm which can be implemented as a sound-producing computer program. The earliest roots of physical modeling synthesis are in speech synthesis (Kelly and Lochbaum 1962), followed by early attempts at string simulation (Ruiz 1969; Hiller and Ruiz 1971a,b), and the first truly sophisticated use of physical modeling principles for musical purposes was certainly the CORDIS system, developed by Cadoz and associates (Cadoz 1979; Cadoz et al. 1983) in the late 1970s and early 1980s. Many varieties of simulation algorithms have emerged, and most notably modal synthesis (Morrison and Adrien 1993) and digital waveguides (Smith III 1992), both of which will be described in more detail in the next section. The most important benefit of physical modeling synthesis is that, in theory, it should be possible to generate sound of a very

natural and acoustic character; in addition, both instruments and control are parameterised in terms of physical quantities and constants, which should ideally be intuitive and approachable for the eventual user.

It is interesting that physical modeling synthesis has not been as widely adopted as earlier conventional synthesis techniques. There are a few good reasons for this:

Model choice: There are different levels at which an acoustic system such as a musical instrument may be modeled. In many cases, a complete model of the system is not yet available, and model simplifications can lead to sound output of an unnatural or synthetic character.

Algorithm design: The step from a model to a sound-producing algorithm operating at an audio rate is a non-trivial one, with many concerns—chief among these are perceptual artefacts and ensuring numerical stability.

Computational cost: The operation count and memory requirements for physical modeling synthesis can be much larger than for conventional synthesis algorithms.

Instrument design and control: Learning to design and play a physical model is not straightforward, and requires a lengthy acclimitisation process for the eventual user—much as in the case of learning an acoustic instrument.

The NESS Project (standing for *Next Generation Sound Synthesis*) was a recent five-year effort devoted to addressing the difficulties above. Work on the first two, at the level of models and algorithm design, and across a wide variety of instrument types, are described in this article. Work on the third, through implementation strategies in parallel hardware and, to a much more tentative level, the fourth, are detailed in a companion article (Bilbao et al. 2019b). This article is intended for a relatively nontechnical audience. For a more detailed overview, see the conference proceedings articles

(Bilbao et al. 2013, 2014).

A complete repository for all publications which have been produced during the NESS Project, tutorial material, as well as links to musical works and the NESS interface are available at the project website at `www.ness-music.eu`.

State of the Art

Most approaches to physical modeling synthesis are based, heavily, on linear system theory, and the powerful simplifications it engenders. This is not to say that a physical model of a musical instrument is a linear system; indeed, it virtually never is. The standard model coalesced with the landmark work of McIntyre, Schumacher, and Woodhouse (1983), which cemented the critical notion that a musical instrument can be divided into an excitation mechanism and a resonator. For sound synthesis purposes, the excitation mechanism, driven by an external signal supplied by the player is strongly nonlinear, but assumed to be point-like, or lumped. Examples are the interactions of the bow with the string, the hammer with the string, and the lip with the reed. The resonator is modelled as a linear system of finite spatial extent—examples are the string, bar, plate and acoustic tube. It is the linear character of the resonator which has been fruitfully exploited in modern physical modeling synthesis.

A linear (more precisely linear and time invariant) representation of a distributed system leads naturally to a description in terms of modes of vibration; with each such mode is associated a shape, frequency and damping factor. In isolation, the dynamics of such a system may be expressed completely in terms its modes, and synthesis becomes similar to additive approaches—one constructs a sound from sinusoidal components, where, in contrast to additive synthesis, there are precise physical constants determining the weightings of the various components. In implementation, a linear system

representation is attractive because the various modes evolve independently. See Figure 1. Modal approaches have been used for some time, particularly in the successful MOSAIC (later Modalys) synthesis environment, developed by IRCAM (Adrien 1991; Morrison and Adrien 1993), and have also appeared independently elsewhere (Bruyns 2006; van den Doel 2007).

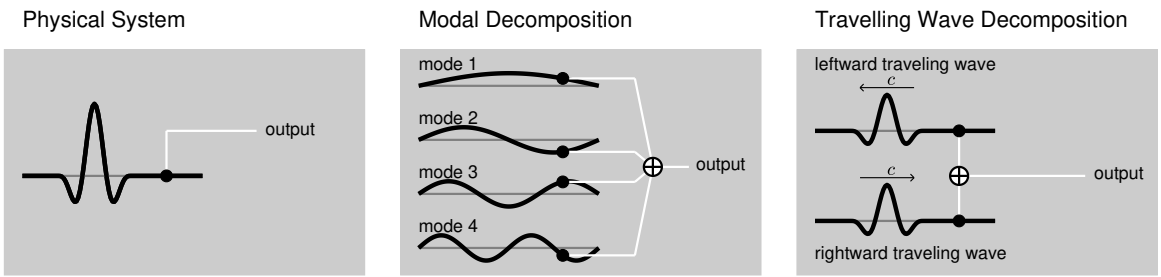

Figure 1. Physical system in 1D (left), modal decomposition (middle) and a traveling wave decomposition (right).

Under further restrictions, namely that the distributed object is uniform (or spatially homogeneous), with constant wave speed, and well-modeled in 1D (such as a simplified string model, or cylindrical or conical acoustic tubes), then another useful characterisation is in terms of traveling waves. Under such conditions, the vibration of such an object may be described completely in terms of so-called wave components, which travel throughout the medium without change in shape. Such a wave decomposition leads immediately to a very efficient discrete-time implementation in terms of delay lines. See Figure 1. Waveguides were developed by Smith at CCRMA, from the starting point of the non-physical Karplus Strong algorithm (Karplus and Strong 1983; Jaffe and Smith III 1983); the first publication on the use of digital waveguides for sound synthesis appeared at the ICMC in 1986 (Smith III 1986), though they had been proposed a year earlier for artificial reverberation purposes (Smith III 1985). They have since seen enormous application to physical modeling synthesis for stringed and wind instruments. See Smith III (1992) for an early overview

of digital waveguides.

Such methods are undeniably powerful; and yet, there are underlying limitations to their use. A major roadblock is the presence of nonlinear effects in the resonator itself. The perceptual effects of such nonlinearities range from relatively minor (as in, e.g., the case of phantom partials in heavy-gauge strings (Conklin 1999)) to dominant, as in the case of crashes in gongs and cymbals (Rossing and Fletcher 1983), and rattling in instruments such as the snare drum (Rossing et al. 1992) or fretted string instruments (Bilbao and Torin 2015). But there are other limitations even in the linear case. Modal methods rely on the availability of modal shapes and frequencies. In certain simplified cases, these are available in closed form; in most, however, they are not, and must be computed numerically, and stored—a potentially very large undertaking, particularly in the 3D setting. The efficiency advantage of digital waveguides is limited to linear 1D systems, and, more strictly to those with low dispersion—restrictions which rule out various musical components of interest such as vibrating bars, or tubes of variable cross section. For more on the limitations of such methods, see Bilbao (2009b).

Time stepping methods, whereby the various components of a musical instrument are represented over grids, and then advanced over discrete time intervals are a mainstream simulation technique with a very long history. Fleetingly used for string synthesis (Ruiz 1969; Hiller and Ruiz 1971a,b), they were later adopted as a brute force tool for the scientific study of musical instruments (Bacon and Bowsher 1978; Boutillon 1988; Chaigne and Askenfelt 1994), and finally again for synthesis purposes (Kurz and Feiten 1996; Bensa et al. 2003). Independently, time-stepping methods for lumped mass-spring networks were developed by Cadoz and associates, leading to the first modular physical modeling synthesis environment, CORDIS (Cadoz 1979; Cadoz et al. 1983). Such time-stepping methods consume more computational resources than methods such as digital waveguides, but are more general, and are able to deal

directly with complex nonlinearities, as well as time-varying behaviour through player interaction. There are many varieties of such methods; under the NESS Project, relatively simple finite difference (FD) (Strikwerda 1989) and finite volume methods (Leveque 2002) have been used. See Figure 2. One useful feature of standard FD methods is that updating at a given grid point is local—only neighbouring grid values need be employed. This leads to great simplifications, especially when dealing with connections between objects, and also yields computational structures suitable for parallelisation. For more on parallelisation aspects of the sound synthesis methods presented here, see the companion article (Bilbao et al. 2019b).

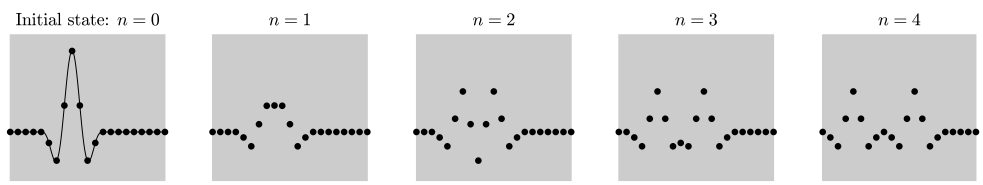

Figure 2. Time stepping method in 1D. At left: the initial state of a system, at time $n = 0$, as represented over a grid (and sampled from an underlying continuous distribution, indicated as a solid line), and then as time progresses at subsequent iterations $n = 1, \ldots, 4$.

Models

In this section, a variety of models of musical instruments of distinct types are presented, with an emphasis on detailed modeling of both highly nonlinear behaviour, as well as time-varying control.

Brass Instruments

The acoustics of a brass instrument is determined primarily by the bore profile. See, e.g., Caussé et al. (1984). A note is generated by buzzing the lips to set up oscillations within the tube, the fundamental frequency of which is close to one of the natural resonance frequencies of the bore. To modify the resonant frequencies, additional

lengths of tubing can be introduced, such as those in the valve sections of a trumpet. See Figure 3, showing a hypothetical brass instrument with additional lengths of tubing, to be activated by valves (not pictured).

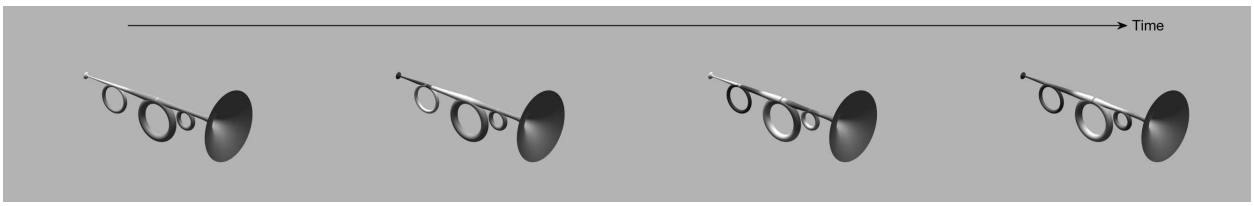

Figure 3. Three-valved brass instrument model, excited with an impulse under partial valved conditions. Light-coloured regions indicate high pressure, and dark-coloured regions indicate low pressure.

The synthesis of brass instrument sounds has been approached using several methods—from the early AM synthesis of Risset (1966) to FM synthesis (Morrill 1977) and then later physical modeling work (Cook 1991) using digital waveguides. Under the NESS project, a fully articulated brass instrument environment has been developed using FD methods (Bilbao and Harrison 2016), and the algorithm design resembles the very early speech synthesis work by Kelly and Lochbaum (1962). The user has complete control over the instrument design, including the specification of the bore profile, valve positions and lengths of valve sections, and lip parameters. The instrument can be played through the manipulation of several time-varying control streams including mouth pressure, lip frequency, and multiple valve depression positions. In addition to generating note transitions, this model can also produce sounds with multiphonic timbre caused by partially open valve configurations, a novel feature of this work. Because execution times are very small, brass synthesis has been a mainstay for composers using the NESS system. See Bilbao et al. (2019b).

Additional information on the brass instrument environment can be found in an earlier publication in this journal that also documents its implementation in the

Composers Desktop Project (Harrison et al. 2015). In addition, a multi-platform software release and tutorial files are available at `www.ness-music.eu`.

Bowed String Instruments

The oscillations of a bowed string arise from the strongly nonlinear friction interaction between the bow hair, coated in rosin, and the string surface at the bowing point. Under certain excitation parameter choices (e.g. bow force, position, velocity), the string vibrations settle into a periodic stick-slip regime, known as Helmholtz motion McIntyre and Woodhouse (1979). Other, less musically pleasant oscillation regimes are found elsewhere in the playing parameter space, some of which are characterised by screeching noise or overtone jumps. The left-hand fingers of the musician are used to clamp the string against the fingerboard and to transition between stopped notes, often with added effects (*vibrato* being the most well-known example).

Existing bowed string physical models have relied on travelling-wave representations which go back to work by Smith (Smith III 1986) (see, e.g., Mansour et al. (2016)). In this framework, however, the implementation of time-varying or distributed nonlinear interactions is non-trivial, thus drastically restricting the range of reproducible bowing gestures.

A FD scheme for the bowed string system is presented in (Desvages and Bilbao 2016) (see Figure 4). A two-polarisation linear string is coupled to a stopping finger, which allows users to play different notes along the neck, and to execute certain gestures (e.g. *glissando*, *legato*, *vibrato*). The fingerboard is modelled as a rigid barrier underneath the string. Other gestures are enabled through a dynamic nonlinear bow model, which can bounce against the string to simulate, e.g., *spiccato* bowing. The nonlinear friction force applied transversally by the bow onto the string depends on the relative velocity between string and bow Smith and Woodhouse (2000).

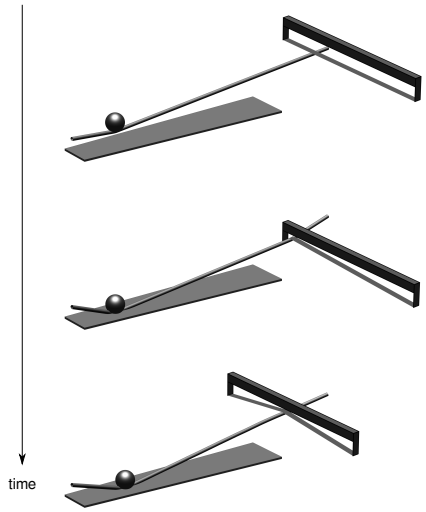

Figure 4. A bow is lowered onto a string, which is pinned between a finger and the fingerboard. The bow is then pushed across the string to set it into motion.

Guitars

Like bowed string instruments, guitar-like instruments are enormously complex constructions, consisting of a set of strings coupled via a bridge to a body which then radiates acoustic energy to the listener; the linear behaviour of the body and radiation characteristics has undergone intensive numerical investigation through time-stepping methods (Derveaux et al. 2003; Bader 2005). Synthesis methods for linear guitar string models include digital waveguides, often accompanied by a filter summarising the effects of the body and radiation (Laurson et al. 2001).

Much less investigated has been the strongly nonlinear collision interaction between the strings and fretboard, particularly under the action of stopping or tapping fingers. Such nonlinear behaviour leads to delicate twanging and rattling effects, particularly when the fingers are able to move. Under simple plucked and unstopped conditions, the strings will bounce off the raised frets, leading to highly amplitude-dependent timbres (Evangelista and Eckerholm 2010; Rabenstein and Trautmann 2004). The dynamics of

the stopping fingers may be modelled separately; when they are present, it is possible to emulate chord changes, sliding barre chords, as well as the ability to play harmonics when the fingers touch the string very lightly. See Figure 5.

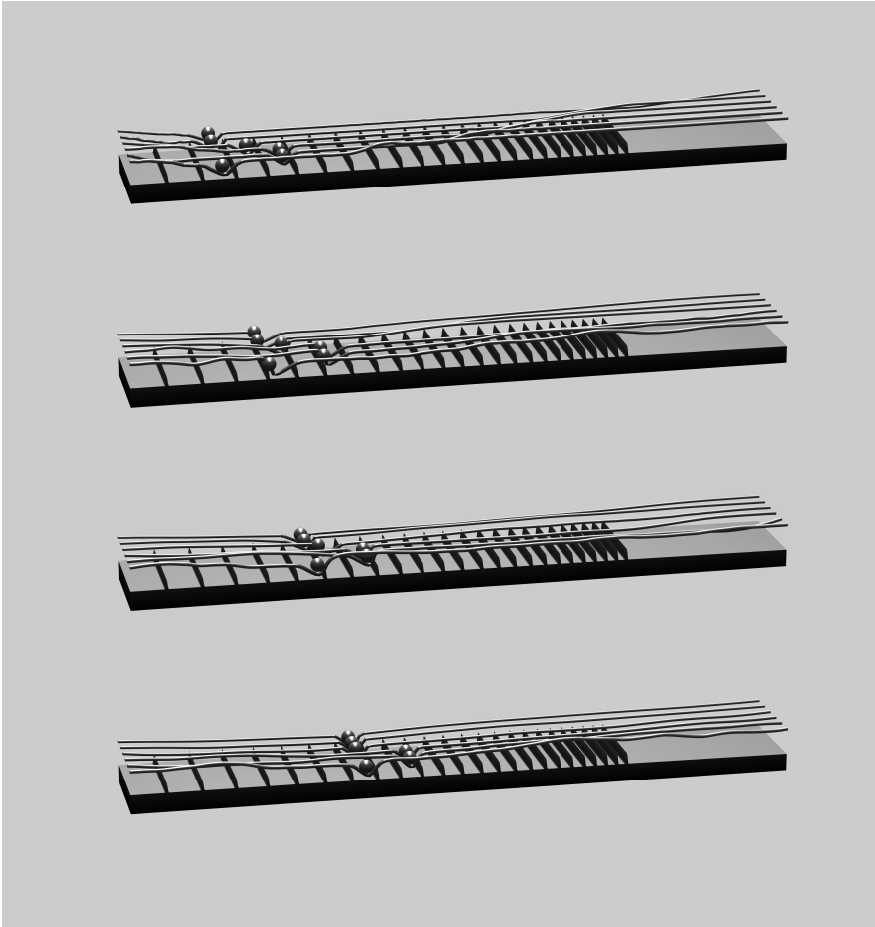

Figure 5. Six-string guitar model, in the course of a time-varying gesture including fretboard and finger interactions.

A complete system for the finger/string/fretboard interaction has been developed using FD methods, allowing for an arbitrary number of strings, a user-defined fretboard, and stopping fingers, all with independent time-varying control over finger positions and forcing. The details of the model and algorithm are presented in Bilbao and Torin (2015); not present in the current model are the body and radiation.

3D Wave-based Room Acoustics Simulation

One important target system in the NESS project has been 3D room acoustics, with the overarching goal of full-wave simulations at audio rates. As opposed to image source and ray tracing techniques (Savioja and Svensson 2015), which are high-frequency approximations based on geometrical acoustics, wave simulation is valid across all audible frequencies and can be viewed as a complete approach to room acoustics simulation.

Room acoustic wave simulations were first attempted in the 1990s using finite difference methods (Chiba et al. 1993; Botteldooren 1994, 1995) as well as the digital waveguide mesh paradigm applied in an equivalent finite-difference form (Savioja et al. 1994). In the NESS project, the main developments were with respect to the modelling of complex geometries and frequency-dependent boundaries (Bilbao et al. 2016), air absorption effects and acceleration over parallel computing hardware (Webb and Bilbao 2011), and the use of non-Cartesian spatial grids (Hamilton and Bilbao 2013) for computational efficiency. Such wave-based simulations are typically parallelisable over the underlying spatial grid. This, with the help of modern parallel computing hardware (such as graphics processing unit (GPU) devices), has made it possible to carry out large-scale wave simulations of room acoustics at audio rates such as 44.1 kHz (Webb and Bilbao 2011). This is illustrated here in the case of a concert hall of approx. 14,000 cubic metres in volume (see Figure 6). In Figure 7, snapshots of the time evolution of the acoustic field in response to a point-source excitation are shown. What is notable in these images is the diffraction that is faithfully reproduced at many points in the scene (e.g., steps, balcony, and seats) – such effects are largely impractical to reproduce within the geometrical acoustics paradigm.

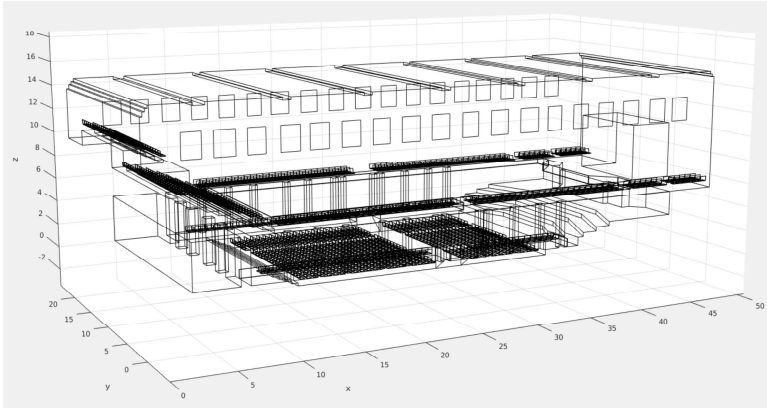

Figure 6. 3D concert hall model with axis units in metres.

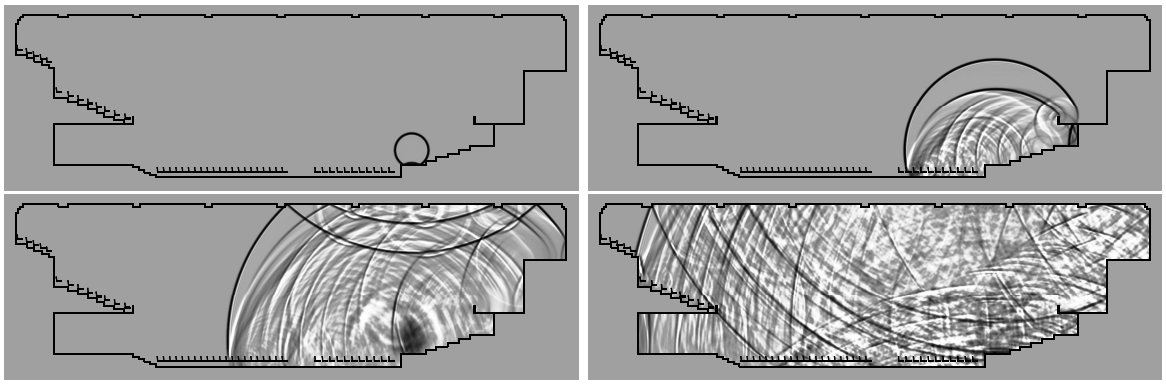

Figure 7. Snapshots of a simulated 3D acoustic field within large concert hall, as two-dimensional x-z–slices (at $y = 8.5$ m) at times: 5 ms, 25 ms, 50 ms, and 100 ms (left-to-right, top-to-bottom).

Percussion Instruments

Timpani are an example of percussion instruments that are well-suited to the type of large-scale 3D simulations attempted in the project. By combining the room models detailed in the previous section with embedded membranes and shells, it is possible to create a complete simulation of multiple timpani in a virtual space (Bilbao and Webb 2013).

The time-stepping model used here is similar to that employed by Rhaouti et al. (1999), consisting of simplified non-linear membrane and boundary reflection from the body of the instrument. The instrument is played by applying a time-varying force at a point on the membrane, representing a mallet or drumstick strike. The position of the strike leads to variations in timbre, while higher amplitudes lead to characteristic pitch glide effects. Figure 8 shows a slice of a full simulation using four timpani being played in a room. Audio output can be drawn from any location (or multiple locations for spatialised output).

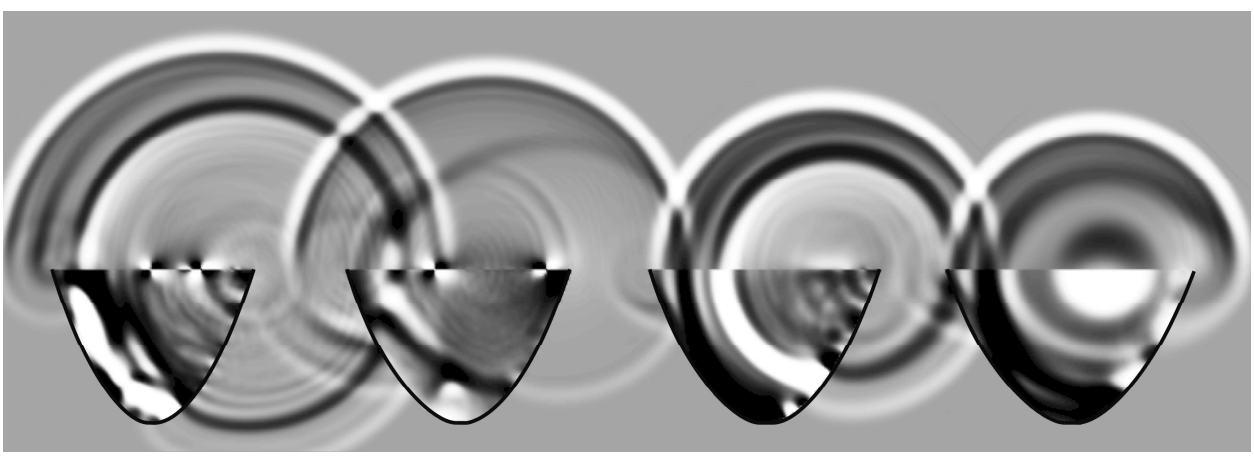

Figure 8. A 2D slice of the acoustic field from a simulation of four timpani drums being played in a room model.

Snare drums are another interesting application of time-stepping methods. These, in fact, are able to capture all the complex interactions (Rossing et al. (1992)) that take place

among the different components of the system, as already anticipated in the introduction. The snare drum model developed during the NESS project is composed of two membranes, connected together by a rigid cavity. The bottom membrane is in contact with a set of stiff snares and the drum is embedded in a 3D room. Here, the excitation mechanism is a drumstick, explicitly included as a lumped object. The details of the model can be found in (Torin et al. 2014).

Figure 9 shows the typical initial excitation phase of a snare drum. In the first instants, when the mallet travels against the membrane, the drum and the surrounding acoustic field are at rest. During the interaction with the membrane, there is a build-up of positive pressure inside the cavity, that pushes the bottom membrane and the snares downwards. When the snares collide against the membrane, a wave front is created, which propagates inside the cavity and excites the upper membrane. This behaviour continues until all the energy of the system is dissipated. These repeated collisions give the snare drum its characteristic rattling sound which, like the rest of the instruments presented here, can be captured at any of the points inside the virtual room.

Finally, another drum model that has been created is the bass drum. This model shares many similarities with the snare drum, but apart from lacking the snares, it has one fundamental difference: both membranes include nonlinearities, in the form described by the von Kármán equations. This virtual model allows composers to produce the dramatic attacks and the pitch glide effects typical of bass drums and has been used in several compositions during the NESS project.

Modular Synthesis Environments

Beyond modeling real-world instruments, or variants of them, an ultimate goal of physical modeling synthesis is to model instruments which, although lacking real-world counterparts, still behave according to physical principles. In this way, it is hoped, the

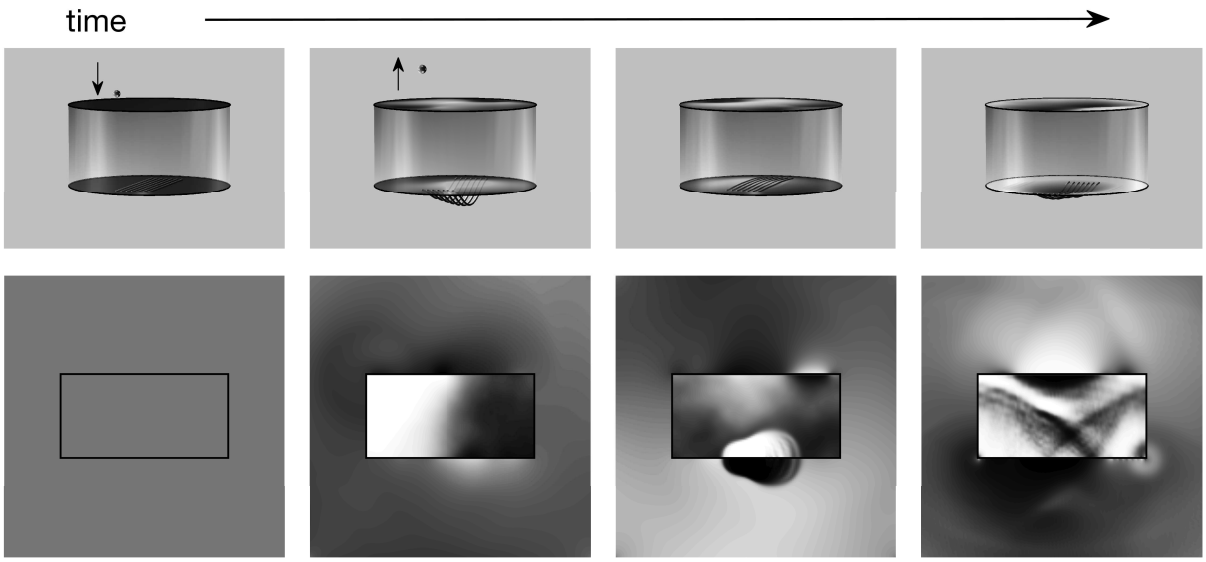

Figure 9. Evolution of the snare drum simulation (above) and corresponding 2D slice of the acoustic field (below), at four different time steps. The arrows next to the drumstick indicate the direction of travel.

door may be opened to new classes of synthetic sound with an acoustic character.

One approach is through the use of modular constructions: an instrument may be assembled given a set of elementary of canonical objects which obey certain physical laws, as well as connections between them. Such ideas have a long history, and were first explored by Cadoz, leading ultimately to the CORDIS-ANIMA environment (Cadoz et al. 1993), for which the canonical elements are masses and springs. Modal synthesis environments also allow for modular instrument construction, and other modular formalisms have also been proposed (Rabenstein et al. 2007).

In the NESS Project, distributed canonical elements, such as strings, bars, or plates have been employed. Each instance can be represented over a grid, and then time-advanced using an FD method. A connection, in the simplest case, can be idealised as a pointwise link between two given elements, at given locations. A given connection element may have its own internal dynamics, and may be characterised by a mass, damping, and a stiffness, which could be nonlinear—reflecting hardening spring

behaviour or even intermittent loss of contact, leading to highly nonlinear responses.

Input to such an instrument can take a variety of forms. Perhaps the simplest form of excitation is a series of plucks or strikes, in which case for a given component, and at a given location, a pulse-like force input signal is sent, where the user has control over the duration of the pulse (which is generally short, and on the order of 1-5 milliseconds), as well as the amplitude, in Newtons. Another approach is to treat the instrument as an effect, and to send in audio input. In either case, for a nonlinear instrument design, the resulting timbres will depend heavily on the input amplitude.

There are great opportunities for multichannel synthesis from such modular constructions. For a given instrument, which will in general consist of multiple interacting components, outputs may be drawn simultaneously from “virtual pickups" placed at different locations on distinct components. For a given input, then, there will be a natural degree of coherence among the various outputs, and thus a holistic approach to spatialisation is possible. See Figure 10. For more on the use of such environments in a multichannel setting, see the companion article (Bilbao et al. 2019b). Various modular frameworks have emerged throughout the course of the NESS Project. The first complete environment, which was ported to GPU, was the so-called *zero code*, which allowed for the nonlinear interconnection of plates, and percussive input; it was later refined to allow audio input. A later iteration, called *net1*, involved the interconnections of strings and bars, using rattling nonlinear connections. Both have been used in a multichannel setting to generate complete pieces of music by various artists. See Figure 11. For more on the technical considerations of designing such modular synthesis systems, see the proceedings articles (Bilbao 2009a; Bilbao et al. 2019a).

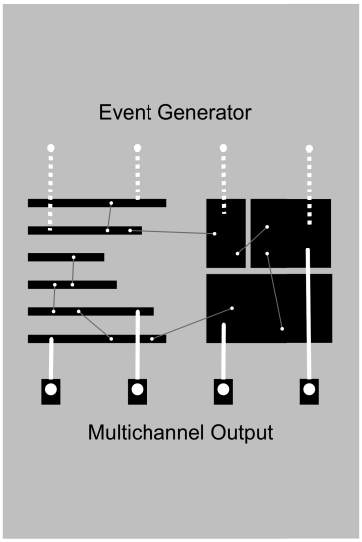

Figure 10. Functional diagram of a modular network constructed from interconnected bar and plate elements, subject to input excitations from an event generator (score) and yielding multi-channel output.

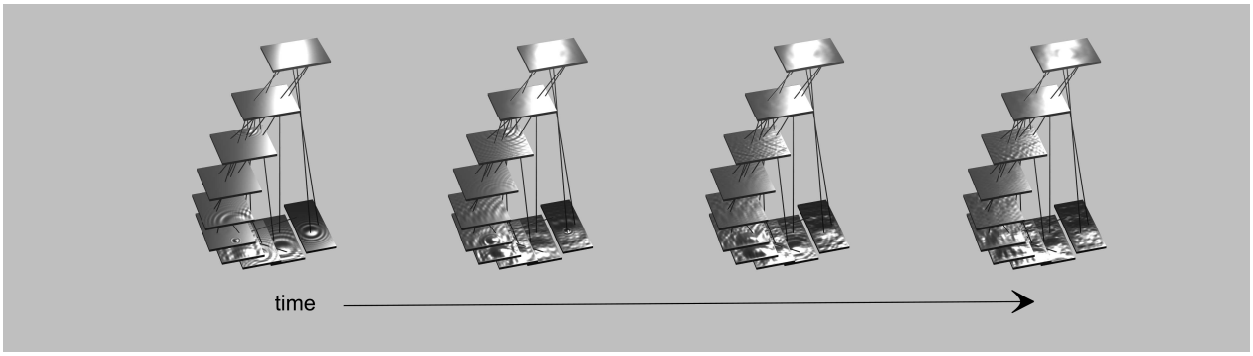

Figure 11. Snapshots of the time evolution of a connected network of plates.

Time-stepping Algorithms: Basics

The main advantage of time-stepping methods relative to other physical modeling sound synthesis techniques is generality; such methods are able to handle a large variety of musical instrument types, including the emulation of strongly nonlinear effects. One price to be paid for such generality is computational cost—always a concern, but perhaps currently less of one due to the availability of fast parallel hardware. Deeper concerns are at the algorithmic level—poorly designed time-stepping methods can produce sound of poor quality, due to perceptual artefacts and, in some cases, may not produce a meaningful solution. A major part of the algorithm design effort under the NESS Project has been concerned with attacking such difficulties.

A Simple Example: an FD Scheme for the 1D Wave Equation

Perhaps the very simplest system of interest in physical modeling, and one which may be familiar to the reader, is the 1D wave equation:

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

Here, the function $u(x,t)$, for a spatial coordinate $x \in [0, L]$, for some length L, and for time $t \geq 0$ represents an unknown of interest, and Equation (1) describes its time evolution. If Equation (1) is intended to represent the dynamics of an ideal string, then $u(x,t)$ represents string displacement; in the case of a lossless cylindrical acoustic tube, it could represent the pressure field. In either case, c is the wave speed and is assumed constant. The 1D wave equation must be supplemented by two initial conditions, as well as a boundary condition at each end—for simplicity, assume that $u(0,t) = u(L,t) = 0$, which has the interpretation of a “fixed" termination in the case of a string. The 1D wave equation (1) is the starting point for digital waveguide synthesis methods (Smith III 1992).

The first step in the design of a time-stepping method is the definition of a grid. See Figure 12. The numerical solution will be calculated at multiples of a given time step T (in seconds), or at times $t_n = nT$, for integer $n \geq 0$; in audio applications, $F_s = 1/T$ is the sample rate. In space, the solution is approximated at spatial intervals of X (in metres), or at locations $x_l = lX$, for integer l. Because the spatial domain is of finite extent, it is simplest to set $l = 0, \ldots, N$, for an integer N such that $L/X = N$. The grid function u_l^n, then, represents an approximation to $u(x_l, t_n)$.

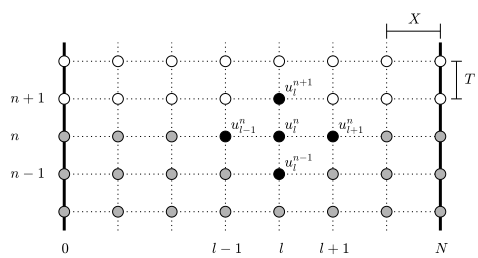

Figure 12. Spatiotemporal grid for scheme in Equation (3) *for the 1D wave equation. Given values of the grid function u_l^n, known through time step n (shown in grey), values at time step $n+1$ may be updated, at a given spatial index l, with reference to neighbouring values (shown in black).*

Consider the following approximations:

$$\left.\frac{\partial^2 u}{\partial t^2}\right|_{x=x_l, t=t_n} \approxeq \frac{1}{T^2}\left(u(x_l, t_{n+1}) - 2u(x_l, t_n) + u(x_l, t_{n-1})\right) \tag{2a}$$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=x_l, t=t_n} \approxeq \frac{1}{X^2}\left(u(x_{l+1}, t_n) - 2u(x_l, t_n) + u(x_{l-1}, t_n)\right) \tag{2b}$$

Identifying $u(x_l, t_n)$ with u_l^n leads to the finite difference scheme

$$u_l^{n+1} = 2u_l^n - u_l^{n-1} + \lambda^2\left(u_{l+1}^n - 2u_l^n + u_{l-1}^n\right) \tag{3}$$

which approximates Equation (1). The parameter $\lambda = cT/X$, sometimes referred to as the Courant number (Strikwerda 1989) plays an important role in the eventual

behaviour of the scheme, as will be discussed momentarily. Given values u_l^{n-1} and u_l^n, the scheme computes an update to the values at the next time step, u_l^{n+1}, a process which will be repeated within a run time loop operating at a sample rate of F_s. See Figure 12. The update above holds for values of l with $l = 1, \ldots, N-1$. At the endpoints $l = 0$ and $l = N$, it appears to require values of the grid function from outside the domain; this can be addressed by imposing the boundary conditions $u_0^n = u_N^n = 0$, corresponding to fixed termination. This basic scheme was used to generate the numerical results shown in Figure 2.

Dispersion and Bandlimiting

Consider first the operation of the scheme in Equation (3) subject to a static initial condition in the form of a peaked distribution. Assume $L = 1$ m and $c = 882$ m/s, and that $F_s = 44,100$ Hz. In the ideal case, one should expect the distribution to split into two wave-like solutions, traveling to the left and right with speed c, and maintaining the initial shape. This is indeed what scheme in Equation (3) yields, when the Courant number λ is set to 1. See Figure 13 at top. If one reads, say, an output displacement anywhere along the string, one should expect, also, that the frequency response will consist of equally spaced frequencies, up to the Nyquist limit, at multiples of $c/2L = 441$ Hz, the fundamental; this is also true for scheme given in Equation (3) when $\lambda = 1$. The same is not true if λ is chosen differently: see Figure 13 at bottom, where results are plotted for scheme given in Equation (3) with $\lambda = 0.6$. The initial distribution no longer travels coherently—it is dispersed. Accompanying this is a great reduction in bandwidth (down to approximately one-third of the Nyquist limit), and, furthermore, the partials are no longer equally spaced (they are now inharmonic). Such anomalous behaviour (known as numerical dispersion) is a major difficulty in the design of time-stepping methods, and can lead to a great reduction in sound quality, through the loss of bandwidth, smearing of responses, as well as other perceptually undesirable features

such as chirps.

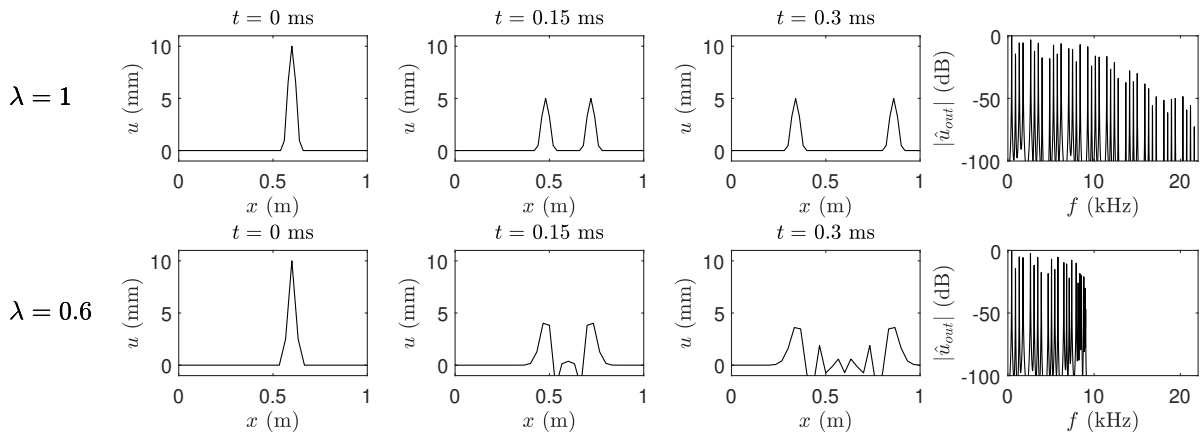

Figure 13. Time evolution of an initial distribution according to the scheme given in Equation (3)*, and resulting output magnitude spectrum $\hat{u}_{out}$, in dB for $\lambda = 1$ (top) and $\lambda = 0.6$ (bottom).*

Numerical Stability and Energy Balances

One property of the 1D wave equation under fixed termination is that it is lossless; that is, it possesses a conserved quantity $E(t)$, defined to within a constant multiplicative factor as

$$E(t) = \int_0^L \frac{1}{2}\left(\frac{\partial u}{\partial t}\right)^2 + \frac{c^2}{2}\left(\frac{\partial u}{\partial x}\right)^2 dx' = \text{constant} \geq 0 \tag{4}$$

The total energy of the system remains constant; furthermore, it is non-negative, providing for bounds on the state itself.

The scheme in Equation (3) possesses a similar conserved quantity, $E^{n+\frac{1}{2}}$, which may be defined as

$$E^{n+\frac{1}{2}} = \sum_{l=0}^{N} \frac{X}{2T^2}\left(u_l^{n+1} - u_l^n\right)^2 + \sum_{l=0}^{N-1} \frac{c^2}{2X}\left(u_{l+1}^{n+1} - u_l^{n+1}\right)\left(u_{l+1}^n - u_l^n\right) = \text{constant} \tag{5}$$

This quantity is clearly a discrete approximation to the total energy of the system, from Equation (4). It may be shown that this quantity is non-negative only under the

condition $\lambda \leq 1$, which is sometimes referred to as the Courant-Friedrichs-Lewy condition. Indeed, when $\lambda > 1$, the scheme is numerically unstable—spurious oscillations, usually at the resolution of the grid itself, grow exponentially. See Figure 14.

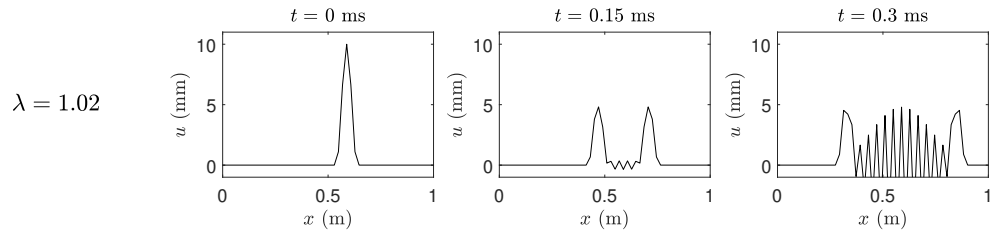

Figure 14. Time evolution of an initial distribution according to Equation (3)*, illustrating numerical instability when* $\lambda = 1.02$*.*

The numerical energy conservation property is a useful one for two reasons: first, it allows the determination of conditions for numerical stability, and second, it provides a useful debugging tool—by monitoring the numerical energy in the run-time loop, any deviations on the order of more than machine precision indicate a programming error. The same ideas extend to the notion of an energy balance, where, for more realistic models of musical instruments, stored energy is related to integrated power loss and supplied power. It is also possible to approach stability for relatively complex systems, involving strongly nonlinear and coupled systems in this way, and thus the numerical energy balance has been used as a design principle for all computer codes in the NESS project.

Concluding Remarks

Mainstream time domain numerical simulation techniques offer a general approach to the simulation of complex musical instruments for physical modeling synthesis, allowing for the simulation of a system according to the most basic laws of physics, and without recourse to simplifying assumptions—which may be unphysical and ultimately degrade sound quality. And yet, two major challenges remain. One is the step to real

time performance: while already possible for some of the systems described here (in particular brass instruments and modular networks, depending on their complexity), the remainder of the algorithms here are available only in offline mode. Here, code parallelisation, either on GPU or in multicore CPU offers a partial solution, provided that the underlying operations are parallelisable, forming an additional design constraint not discussed here. Another deeper issue is that of learning to play these instruments, a process which can be very difficult, just as learning to play an acoustic instrument is. In this case, the experience of the musician, informing control strategies and user interface design, is an essential step towards the mature musical use of these synthesis algorithms. Both of these challenges are discussed in detail in a companion article (Bilbao et al. 2019b).

Acknowledgment

This work was supported by the European Research Council, under grants 2011-StG-279068-NESS and 2016-PoC-737574-WRAM. M. Ducceschi was supported by the Newton International Fellow program, through the Royal Society and British Academy, and a Early Career Fellowship from the Leverhulme Trust.

References

Adrien, J.-M. 1991. "The Missing Link: Modal Synthesis." In G. DePoli, A. Picialli, and C. Roads, (editors) *Representations of Musical Signals*. Cambridge, Massachusetts: MIT Press, pp. 269–297.

Bacon, R., and J. Bowsher. 1978. "A Discrete Model of a Struck String." *Acustica* 41:21–27.

Bader, R. 2005. *Computational Mechanics of the Classical Guitar*. Berlin Heidelberg: Springer-Verlag.

Bensa, J., et al. 2003. "The Simulation of Piano String Vibration: From Physical Models to Finite Difference Schemes and Digital Waveguides." *Journal of the Acoustical Society of America* 114(2):1095–1107.

Bilbao, S. 2009a. "A Modular Percussion Synthesis Environment." In *Proceedings of the 12th International Digital Audio Effects Conference*. Como, Italy, pp. 321–328.

Bilbao, S. 2009b. *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Chichester, UK: John Wiley and Sons.

Bilbao, S., M. Ducceschi, and C. Webb. 2019a. "Large-scale Real-time Modular Physical Modeling Sound Synthesis." In *Proceedings of the 22nd International Digital Audio Effects Conference*. Birmingham, UK.

Bilbao, S., et al. 2016. "Finite volume time domain room acoustics simulation under general impedance boundary conditions." *IEEE/ACM Transactions on Audio Speech and Language Processing* 24(1):161–173.

Bilbao, S., et al. 2013. "Large Scale Physical Modeling Synthesis." In *Proceedings of the Stockholm Musical Acoustics Conference*. Stockholm, Sweden, pp. 593–600.

Bilbao, S., and R. Harrison. 2016. "Passive time-domain numerical models of viscothermal wave propagation in acoustic tubes of variable cross section." *Journal of the Acoustical Society of America* 140:728–740.

Bilbao, S., et al. 2019b. "The NESS Project: Large Scale Physical Modeling Synthesis, Parallel Computing and Musical Experimentation."

Bilbao, S., and A. Torin. 2015. "Numerical Modeling and Sound Synthesis for Articulated String/Fretboard Interactions." *Journal of the Audio Engineering Society* 63(5):336–347.

Bilbao, S., et al. 2014. "Modular Physical Modeling Synthesis Environments on GPU." In

Proceedings of the International Computer Music Conference. Athens, Greece, pp. 1396–1403.

Bilbao, S., and C. J. Webb. 2013. "Physical modeling of timpani drums in 3D on GPGPUs." *Journal of the Audio Engineering Society* 61(10):737–748.

Botteldooren, D. 1994. "Acoustical finite-difference time-domain simulation in a quasi-Cartesian grid." *Journal of the Acoustical Society of America* 95(5):2313–2319.

Botteldooren, D. 1995. "Finite-difference time-domain simulation of low-frequency room acoustic problems." *Journal of the Acoustical Society of America* 98(6):3302–3308.

Boutillon, X. 1988. "Model for Piano Hammers: Experimental Determination and Digital Simulation." *Journal of the Acoustical Society of America* 83(2):746–754.

Bruyns, C. 2006. "Modal Synthesis for Arbitrarily Shaped Objects." *Computer Music Journal* 30(3):22–37.

Cadoz, C. 1979. "Synthèse sonore par simulation de mécanismes vibratoires." Thèse de Docteur Ingénieur, I.N.P.G. Grenoble, France.

Cadoz, C., A. Luciani, and J.-L. Florens. 1983. "Responsive Input Devices and Sound Synthesis by Simulation of Instrumental Mechanisms." *Computer Music Journal* 8(3):60–73.

Cadoz, C., A. Luciani, and J.-L. Florens. 1993. "CORDIS-ANIMA: A Modeling and Simulation System for Sound and Image Synthesis." *Computer Music Journal* 17(1):19–29.

Caussé, R., J. Kergomard, and X. Lurton. 1984. "Input Impedance of brass musical instruments - Comparison between experiment and numerical models." *Journal of the Acoustical Society of America* 75(1):241–254.

Chaigne, A., and A. Askenfelt. 1994. "Numerical Simulations of Struck Strings. I. A Physical Model for a Struck String Using Finite Difference Methods." *Journal of the Acoustical Society of America* 95(2):1112–1118.

Chiba, O., et al. 1993. "Analysis of sound fields in three dimensional space by the time-dependent finite-difference method based on the leap frog algorithm." *Journal of the Acoustical Society of Japan* 49:551–562.

Conklin, H. 1999. "Generation of partials due to nonlinear mixing in a stringed instrument." *Journal of the Acoustical Society of America* 105(1):536–545.

Cook, P. 1991. "Tbone: An interactive waveguide brass instrument synthesis workbench for the NeXT machine." In *Proceedings of the 1991 International Computer Music Conference*. Montreal, Canada, pp. 297–299.

Derveaux, G., et al. 2003. "Time-domain simulation of a guitar: Model and method." *Journal of the Acoustical Society of America* 114(6):3368–3383.

Desvages, C., and S. Bilbao. 2016. "Two-polarisation physical model of bowed strings with nonlinear contact and friction forces, and application to gesture-based sound synthesis." *Applied Sciences* 6(5):135.

Evangelista, G., and F. Eckerholm. 2010. "Player Instrument Interaction Models for Digital Waveguide Synthesis of Guitar: Touch and Collisions." *IEEE Transactions on Audio Speech and Language Processing* 18(4):822–832.

Hamilton, B., and S. Bilbao. 2013. "On finite difference schemes for the 3-D wave equation using non-Cartesian grids." In *Proceedings of the Sound and Music Computing Conference*. Stockholm, Sweden, pp. 592–599.

Harrison, R. L., et al. 2015. "An Environment for Physical Modeling of Articulated Brass Instruments." *Computer Music Journal* 29(4):80–95.

Hiller, L., and P. Ruiz. 1971a. "Synthesizing Musical Sounds by Solving the Wave Equation for Vibrating Objects: Part I." *Journal of the Audio Engineering Society* 19(6):462–470.

Hiller, L., and P. Ruiz. 1971b. "Synthesizing Musical Sounds by Solving the Wave Equation for Vibrating Objects: Part II." *Journal of the Audio Engineering Society* 19(7):542–550.

Jaffe, D., and J. O. Smith III. 1983. "Extensions of the Karplus-Strong Plucked String Algorithm." *Computer Music Journal* 7(2):56–68.

Karplus, K., and A. Strong. 1983. "Digital Synthesis of Plucked-String and Drum Timbres." *Computer Music Journal* 7(2):43–55.

Kelly, J., and C. Lochbaum. 1962. "Speech Synthesis." In *Proceedings of the Fourth International Congress on Acoustics*. Copenhagen, Denmark, pp. 1–4. Paper G42.

Kurz, M., and B. Feiten. 1996. "Physical Modeling of a Stiff String by Numerical Integration." In *Proceedings of the International Computer Music Conference*. Hong Kong, pp. 361–364.

Laurson, M., et al. 2001. "Methods for Modeling Realistic Playing in Acoustic Guitar Synthesis." *Computer Music Journal* 25(3):38–49.

Leveque, R. 2002. *Finite Volume Methods for Hyperbolic Problems*. Cambridge, UK: Cambridge University Press.

Mansour, H., J. Woodhouse, and G. P. Scavone. 2016. "Enhanced wave-based modelling of musical strings. Part 2: Bowed strings." *Acta Acustica united with Acustica* 102(6):1082–1093.

McIntyre, M., R. Schumacher, and J. Woodhouse. 1983. "On the Oscillations of Musical Instruments." *Journal of the Acoustical Society of America* 74(5):1325–1345.

McIntyre, M. E., and J. Woodhouse. 1979. "On the fundamentals of bowed-string dynamics." *Acustica* 43(2):93–108.

Morrill, D. 1977. "Trumpet Algorithms For Computer Composition." *Computer Music Journal* 1(1):46–52.

Morrison, D., and J.-M. Adrien. 1993. "MOSAIC: A Framework for Modal Synthesis." *Computer Music Journal* 17(1):45–56.

Rabenstein, R., et al. 2007. "Block-based Physical Modeling for Digital Sound Synthesis." *IEEE Signal Processing Magazine* 24(2):42–54.

Rabenstein, R., and L. Trautmann. 2004. "Multirate Simulations of String Vibrations Including Nonlinear Fret-String Interactions Using the Functional Transformation Method." *EURASIP Journal on Advances in Signal Processing* 2004:745924.

Rhaouti, L., A. Chaigne, and P. Joly. 1999. "Time-domain modeling and numerical simulation of a kettledrum." *Journal of the Acoustical Society of America* 105(6):3545–3562.

Risset, J.-C. 1966. "Computer Study of Trumpet Tones." Technical report, Bell Technical Laboratories, Murray Hill, New Jersey.

Roads, C., (editor) . 1996. *The Computer Music Tutorial*. Cambridge, Massachusetts: MIT Press.

Rossing, T., and N. Fletcher. 1983. "Nonlinear Vibrations in Plates and Gongs." *Journal of the Acoustical Society of America* 73(1):345–351.

Rossing, T. D., et al. 1992. "Acoustics of snare drums." *Journal of the Acoustical Society of America* 92(1):84–94.

Ruiz, P. 1969. "A Technique for Simulating the Vibrations of Strings with a Digital Computer." Master's thesis, University of Illinois.

Savioja, L., T. J. Rinne, and T. Takala. 1994. "Simulation of room acoustics with a 3-D finite difference mesh." In *Proceedings of the International Computer Music Conference*. pp. 463–466.

Savioja, L., and U. P. Svensson. 2015. "Overview of geometrical room acoustic modeling techniques." *Journal of the Acoustical Society of America* 138(2):708–730.

Smith, J. H., and J. Woodhouse. 2000. "The tribology of rosin." *Journal of the Mechanics and Physics of Solids* 48:1633–1681.

Smith III, J. O. 1985. "A New Approach to Digital Reverberation Using Closed Waveguide Networks." In *Proceedings of the International Computer Music Conference*. Vancouver, Canada, pp. 47–53.

Smith III, J. O. 1986. "Efficient simulation of the reed-bore and bow-string mechanisms." In *Proceedings of the International Computer Music Conference*. The Hague, The Netherlands, pp. 275–280.

Smith III, J. O. 1992. "Physical Modelling Using Digital Waveguides." *Computer Music Journal* 16(4):74–91.

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

Torin, A., B. Hamilton, and S. Bilbao. 2014. "An energy conserving finite difference scheme for the simulation of collisions in snare drums." In *Proceedings of the 17th International Conference on Digital Audio Effects*. Erlangen, Germany, pp. 145–152.

van den Doel, K. 2007. "Modal Synthesis for Vibrating Objects." In K. Greenebaum, (editor) *Audio Anecdotes III*. Natick, Massachusetts: A. K. Peters.

Webb, C., and S. Bilbao. 2011. "Computing room acoustics with CUDA - 3D FDTD

schemes with boundary losses and viscosity." In *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pp. 317–320.