Computational Thinking through Modular Sounds Synthesis

Andrew M. Olney

2022-08-28

Contents

Welcome

This is the official website for "Computational Thinking through Modular Sound Synthesis". This book will teach you computational thinking through modular sound synthesis (hereafter *modular*). You'll learn how to trigger sounds, create sounds, and modify sounds to solve specific sound design problems and create compositions. Along the way, you'll learn computational thinking practices that transcend modular and can be applied to a variety of problem-solving domains, but which are particularly relevant to information processing domains like computing.

If you're wondering whether this is a book about computational thinking, or a book about modular, the answer is both: on the surface, most content is about modular, but computational thinking is a style of thinking reflected in the presentation of the material and gives it additional coherence. As you work through the book, you'll become more proficient in computational thinking practices like decomposition, algorithmic design, evaluation of solutions, pattern recognition, and abstraction.

This book is *interactive*, which is why it is an e-book rather than a paper book. Throughout you will encounter examples, simulations, and exercises that run in your browser to demonstrate and reinforce key concepts. Don't skip the interactive activities!

Chapter 1

Introduction

1.1 Why this book?

Let's start with why I'm writing it. I got into electronic music in the 1990s when I lived in London but never transitioned from DJing to making music, though several of my friends did. A few years ago, they started talking about modular, and in talking to them and trying to find out more about it, I realized a few things:

- The best books (to me) were from the 1970s and 1980s[1]
- Modular synthesis is really well aligned with *computational thinking*

If you've never heard of computational thinking and/or modular, that last point won't make a lot of sense, so let's break it down.

Modular sound synthesis (modular) creates sound by connecting modules that each perform some function on sound. Different sounds are created by combining modules in different ways.

Computational thinking creates runnable models to solve a problem or answer a question. Models can be scientific models (e.g. meteorology), statistical models (e.g. statistics/data science), computation models (computer science), and perhaps other kinds of models.

How are they connected? Modular involves computational thinking when we:

[1] Old books that I like are Crombie (1982) and Strange (1983). Newer books of note are Bjørn and Meyer (2018), which gives a great overview of module hardware and history, Eliraz (2022), which gives a broader overview of issues related to musical equipment and production, and Dusha et al. (2020), which gives a modern but briefer introduction to modular than the older books. There are also some online courses (paid), but since I haven't taken them, I'm not listing them here.

- Simulate an instrument by reverse engineering its sound
- Create new sounds based on models of signal processing

Why should you read this book? This book is about modular, but it approaches modular in a way that highlights computational thinking. I believe this deeper approach to modular will help you do more with modular, other synthesizers, and studio production tools. Additionally, the computational thinking approach should help accelerate your learning of computational-thinking domains in the future. Since computational thinking involves problem solving, this book is full of interactive activities that will let you hone your modular skills - something you won't find in most books!

The next sections give some background on computational thinking and modular to better explain where this book is coming from.

1.2 Computational thinking

Tedre and Denning (2016) present a nice overview of the history of computational thinking. Here's a brief summary.

When the field of computing was taking off in the 1950s, there was interest and discussion about how it was different from other fields (e.g. math). One argument was that computing involved *algorithmic thinking*, which is designing algorithms to solve problems (cf. programming), and this kind of thinking was unique to computing. Some even thought that this kind of thinking could improve thinking generally.

Computational thinking appears to have been coined in the 1980s by Seymour Papert and popularized in his book *Mindstorms* (Papert, 1980). Papert was a mathematician by training, and his approach was much broader than the algorithmic thinking approach that came before. Papert's approach was empirical and embraced model building, which he implemented using simulated microworlds containing robots (LEGO Mindstorms takes its name from this work). It was revolutionary in its time and received a lot of attention from educators and policy makers of widely different backgrounds.

Unfortunately, today it's very hard to get agreement on what computational thinking is, so definitions tend to be squishy. This is likely due to the widespread use of computers and the tendency for everyone to frame computational thinking in terms of what *they* do with computers. Some want to reduce it to computer literacy, others to basic programming, and yet others to discovery learning with computers, etc.

I take a more unified view of computational thinking based on model building and problem solving. I define computational thinking as building a *runnable* model to solve a problem:

- For an algorithmic problem, this is a model of computation (the original computer science view)
- For data science/statistics, this is a statistical model
- For general scientific fields, this is a scientific model of a phenomenon or process

The model doesn't need to run on a computer, but to be a runnable model, it needs to be mechanistic. One of my favorite examples of a non-computer model is MENACE (Michie, 1963), which plays tic-tac-toe (AKA noughts and crosses). MENACE plays tic-tac-toe using matchboxes full of colored beads as shown in Figure 1.1. Each possible board position (starting with a blank board) is represented by a matchbox, and each move is represented by one of nine colored beads. To make a move, a human assistant selects the correct box for the current board position and randomly samples a colored bead, which determines where MENACE makes its move. If MENACE wins the game, the chosen bead from each box is replaced along with extra beads of the same color, and if MENACE loses, the chosen bead is removed. Over time, these bead adjustments make winning moves more likely and losing moves less likely.

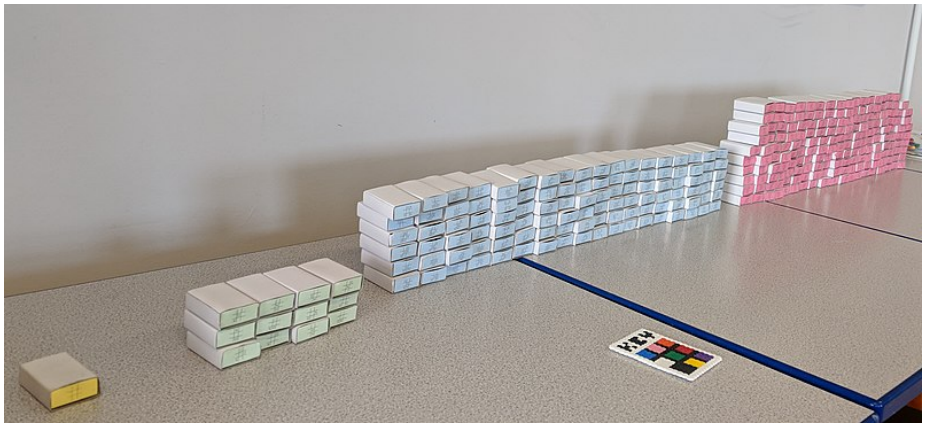

Figure 1.1: Machine Educable Noughts and Crosses Engine (MENACE). Each matchbox corresponds to a possible board position and is full of colored beads corresponding to moves. The color key in the foreground shows the board location indicated by each colored bead. Image © Matthew Scroggs/CC-BY-SA-4.0.

MENACE is a nice example of computational thinking without computers because algorithmic game playing has a long history in computer science and AI. However MENACE is not "an exception to the rule" - teaching computer science without computers has been part of the model curriculum for almost 20 years (Tucker, 2003; Bell, 2021). We really don't need computers for computational thinking!

So how do we *learn* to build runnable models to solve problems (i.e., how do we learn computational thinking)? Well, models are made of interacting elements, so we need to learn those elements, and we need to learn how the elements interact.

Once we know those things, we can customize general problem solving, which has the same basic steps (Polya, 2004):

- Understand the problem
- Make a plan
- Implement the plan
- Evaluate the solution

For any new domain, the big things to learn are the "understand the problem" and the "make a plan" steps of problem solving. That's the approach of this book - for the domain of modular synthesis.

1.3 Modular synthesis

While we can pinpoint the invention of modular synthesis with some precision, it is useful to consider it in a broader context. This section briefly overviews the history of synthesis and how modular fits into it.

Humans have long been interested in musical instruments that incorporate automation or in reproducing sounds by mechanical means. Wind chimes, which play a series of notes when disturbed by wind, appeared in the historical record thousands of years ago. Even before the complete electrification of instruments (synthesizers are electric by definition), there were numerous attempts to partially automate or model sounds, such as barrel organs, player pianos, or speech synthesis using bellows (Dudley and Tarnoczy, 1950) as shown in Figure 1.2.

Consider the difference between wind chimes or a player piano and this speaking machine. Neither of the former is a model of the sound but rather uses mechanical means to trigger the sound (later we will refer to this as sequencing). In contrast, the speaking machine is a well-considered model of the human speech mechanism.

Synthesizers using electricity appeared in the late 19th century.[2] Patents were awarded just a few years apart to Elisha Gray, whose synthesizer comprised simple single note oscillators and transmitted over the telegraph, and Thaddeus Cahill, whose larger Telharmonium could sound like an organ or various wood instruments but weighed 210 tons!

[2]There is some difference of opinion on what qualifies as usage of electricity in this context. For a fuller history of synthesizers, see https://120years.net/wordpress/

Figure 1.2: YouTube video of Wolfgang von Kempelen's speaking machine circa 1780. Image © Fabian Brackhane.

The modular synthesizer was developed by Harald Bode from 1959-1960 (Bode, 1984), and this innovation quickly spread to other electronic music pioneers like Moog and Buchla. The key idea of modular is flexibility. This is achieved by refactoring aspects of synthesis (i.e. functions on sound) into a collection of modules. These modules may then be combined to create a certain sound by patching them together and adjusting module parameters (e.g. by turning knobs or adjusting sliders). An example modular synthesizer is shown in Figure 2.3.

In the 1970s, *semi-modular* synthesizers were developed that did not require patching to make a sound. Instead, semi-modulars were pre-set with an invisible default patch, meaning that the default patch wiring was internal and not visible to the user. Users could then override this default patch by plugging in patch cables. Most semi-modulars from this period also included an integrated keyboard. Arguably, theses changes made semi-modulars more approachable to typical musicians. An example semi-modular synthesizer is shown in Figure 1.4.

Digital technology began replacing the analog technology of synthesizers in the 1980s. As a result, synthesizers got smaller and cheaper. Digital synthesizers made increasing use of preset sounds so that most users never needed to create custom sounds. In comparison to digital synthesizers, modular synthesizers were more expensive and harder to use. An example digital synthesizer is shown in Figure 1.5.

By the 1990s the digital transformation was complete, such that computers could be used to create and produce music in software. Although computers were still relatively expensive at this time, they provided an all-in-one solution that included editing, mixing, and other production aspects. Over the next few

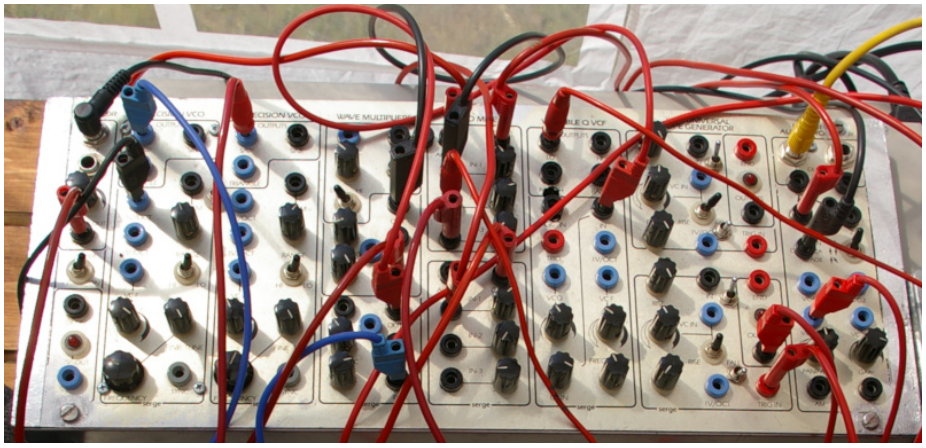

Figure 1.3: A Serge modular system based on a 1970s design. Each module is labeled at the top edge, e.g. "Wave Multiplier," and extends down to the bottom edge in a column. Note that although the modules have the same height, they have different widths. Image © mikael altemark/CC-BY-2.0.

Figure 1.4: A Minimoog semi-modular system from the 1970s. Patch points are primarily on the top edge and hidden from view. Image public domain.

Figure 1.5: A Yamaha DX7 from the 1980s. Note the menu-based interface and relative lack of controls compared to modular and semi-modular synthesizers. Image public domain.

decades as personal computers and portable computing devices became common household items, the costs associated with computer-based music making became dominated by the cost of software and associated audio and MIDI interfaces. Figure 1.6 shows digital audio workstation (DAW) software commonly used in music production.

Figure 1.6: Logic Pro digital audio workstation software. Additional functionality is provided by 3rd-party plugins showing as additional windows on the screen. In the foreground are an audio interface and a MIDI keyboard used for recording/playing audio and entering note information respectively. Image © Musicianonamission/CC-BY-SA-4.0.

The computer-centric approach dominated synthesis for a decade or more, but by the 2010s, improved electronics manufacturing, smartphone technology, and the open-source movement led to lower cost modular synthesizers. Additionally, the Eurorack standard (Doepfer Musikelektronik, 2022a,b) was widely adopted,

leading to +10,000 interoperable modules.[3] As a result, modular synthesis saw a resurgence in popularity. Figure 1.7 shows a Eurorack modular synthesizer.

Figure 1.7: A Eurorack modular synthesizer. The different modules designs and logos reflect the adoption of the Eurorack standard which makes modules from different manufacturers interoperable. Image © Paul Anthony/CC-BY-SA-4.0.

It is perhaps surprising that some 60 years after its creation, modular synthesis is more popular than ever. One possible reason is the reduction in price over time, shown in Table 1.1. However, other trends seem to be at work. While the modular synthesizer was simplified for wider adoption early in its history, first with semi-modular and later with digital synthesizers, the culmination of this trend led to large preset and sample banks that transformed the task of creating a specific sound to searching for a pre-made sound. It's plausible that as the search for sounds became more intensive, the time savings of presets diminished, making the modular approach more attractive. An intersecting trend is a commonly-expressed dissatisfaction with using computers for every aspect of music making and a corresponding return to hardware instruments, including modular.

Table 1.1: The cost of modular, semi-modular, and computer synthesizers over time. Prices are in 2022 dollars.

Decade	Synthesizer	Cost
1960s	Moog modular synthesiser	$96,000
1970s	Minimoog semi-modular	$10,000
1980s	Yamaha DX7	$6,000
1990s	Gateway computer with Cubase	$8,000
2010s	ALM System Coupe modular	$2,400

[3] https://www.modulargrid.net/

Decade	Synthesizer	Cost
...	VCVRack virtual modular	Free

Earlier in this chapter, I argued that a computational thinking approach to modular could help with other synthesizers and studio production tools. Hopefully this brief history helps explain why: modular represents the building blocks of synthesis that later approaches have appropriated and presented in their own way. A square wave oscillator in modular is fundamentally the same as that in another hardware synth or DAW software. If you understand these building blocks in modular, you should understand them everywhere.

1.4 Moving forward

Our next stop is *Sound* where the focus is to "understand the problem". Chapter 2 addresses both the physics of sound and our perception of it, which perhaps surprisingly, are not the same. From there we move into sounds commonly found in music and their properties, ranging from harmonic sounds in Chapter 3 to inharmonic sounds like percussion in Chapter 4.

The remainder of the book alternates between learning model elements (modules), how they interact (patches), problem solving (sound design). The progressive *Modules* and *Sound Design* sections build up from basic approaches to the more complex. By the time we're done, you should have a good foundation to create patches to solve new sound design problems.

Part I

Sound

Chapter 2

Physics and Perception

From the outset, it's important to understand that the physics of sound and how we perceive it are not the same. This is a simple fact of biology. Birds can see ultraviolet, and bats can hear ultrasound; humans can't do either. Dogs have up to 40 times more olfactory receptors than humans and correspondingly have a much keener sense of smell. We can only perceive what our bodies are equipped to perceive.

In addition to the limits of our perception, our bodies also *structure* sensations in ways that don't always align with physics. A good example of this is equal loudness contours. As shown in Figure 2.1, sounds can appear equally loud to humans across frequencies even though the actual sound pressure level (a measure of sound energy) is not constant. In other words, our hearing becomes more sensitive depending on the frequency of the sound.

Why do we need to understand the physics of sound *and* perception of sound? Although ultimately we hear the sounds we're going to make, the process of making the sounds is based in physics. So we need to know how both the physics and perception of sound work, at least a bit.

2.1 Waves

Have you ever noticed a dust particle floating in the air, just randomly wandering around? That random movement is known as Brownian motion, and it was shown by Einstein to be evidence for the existence of atoms - that you can see with your own eyes! The movement is caused by air molecules[1] bombarding the dust particle from random directions, as shown in Figure 2.2.

[1] In what follows, we will ignore that air is a mixture of gases because it is irrelevant to the present discussion.

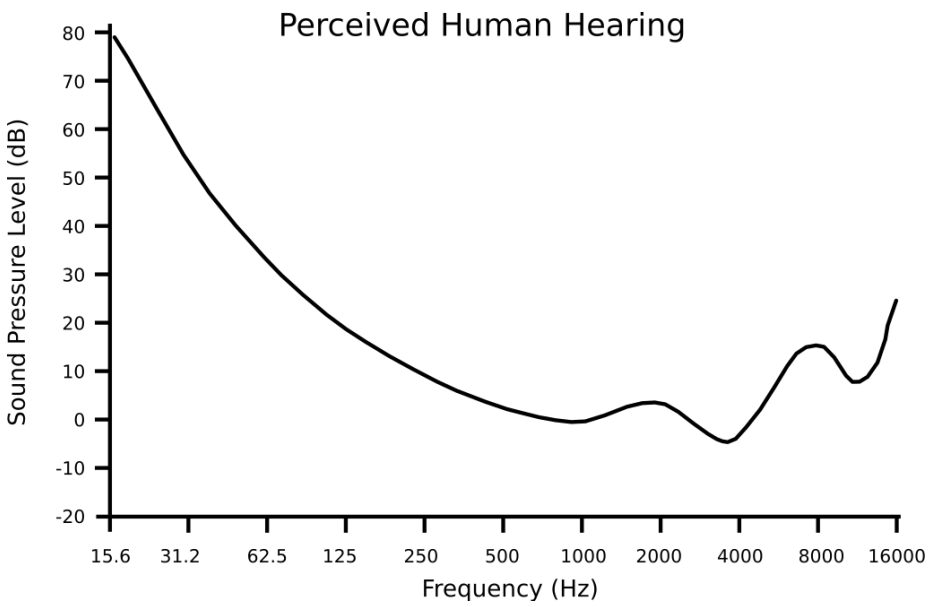

Figure 2.1: An equal loudness contour showing improved sensitivity to frequencies between 500Hz and 4KHz, which approximately matches the range of human speech frequencies. Image public domain.

Air molecules are always whizzing around like this. However, in much of the discussion below, we'll largely ignore this fact and focus instead on the properties of waves passing through air.

Amazingly, it is also possible to see sound waves moving through the air, using a technique called Schlieren photography. Schlieren photography captures differences in air pressure, and sound is just a difference in air pressure that travels as a wave. Take a look at the video in Figure **??**.

TODO https://www.youtube.com/embed/BspouwCTXRo?start=76&end=120

Here is some garbage text. Figure 2.4.

2.2 Frequency and pitch

hz

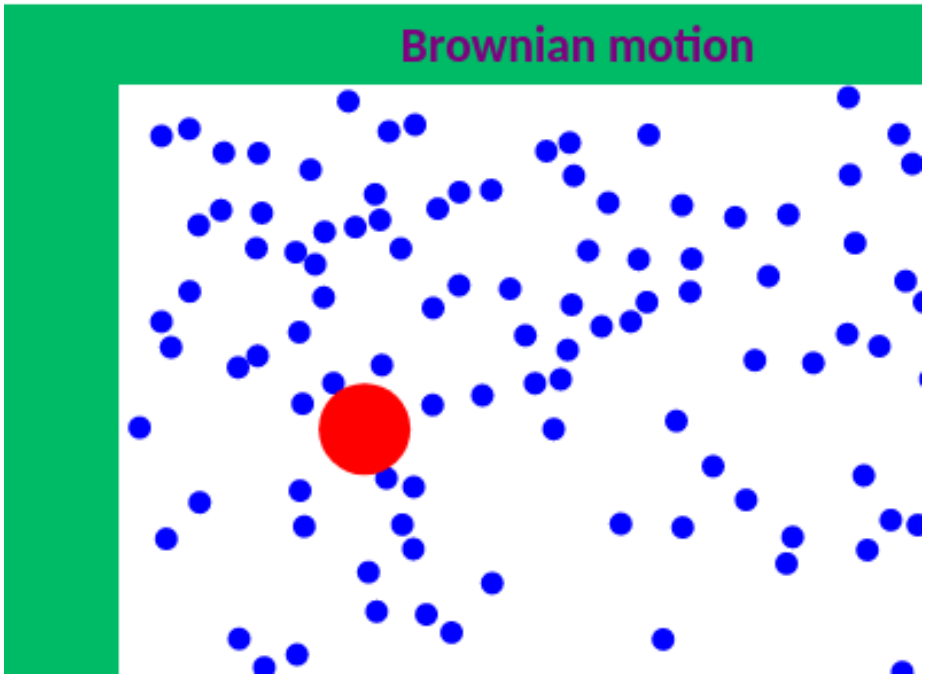

Figure 2.2: Simulation of Brownian motion. Press `Pause` to stop the simulation. © Andrew Duffy/CC-BY-NC-SA-4.0.

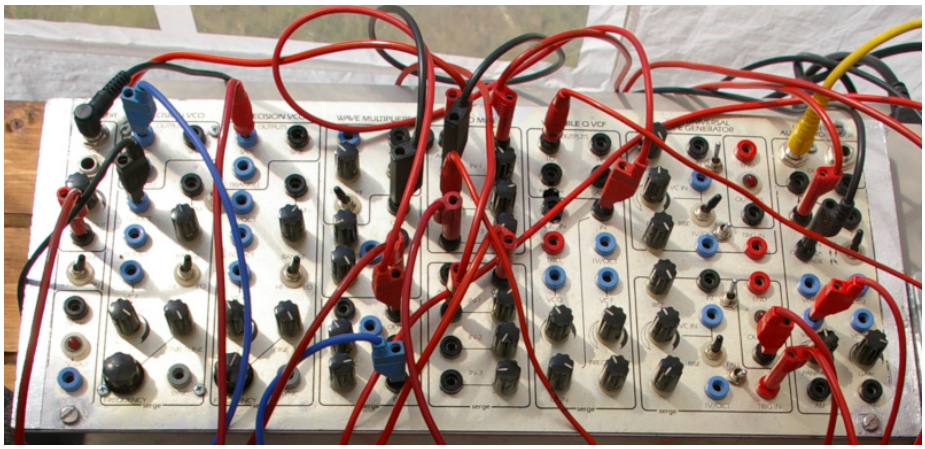

Figure 2.3: A Serge modular system based on a 1970s design. Each module is labeled at the top edge, e.g. "Wave Multiplier," and extends down to the bottom edge in a column. Note that although the modules have the same height, they have different widths. Image © mikael altemark/CC-BY-2.0.

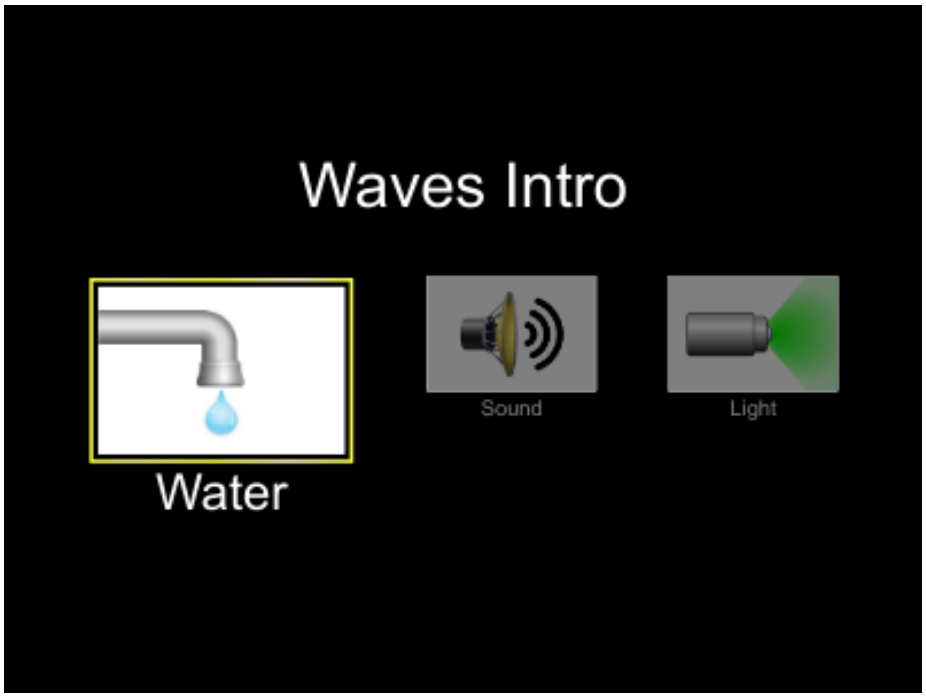

Figure 2.4: Simulation of sound waves. Simulation by PhET Interactive Simulations, University of Colorado Boulder, licensed under CC-BY-4.0.

2.3 Amplitude and loudness

db

2.4 Waveshape and timbre

2.5 Phase and ... phase?

You might think that we only need to understand how humans perceive sound and not the physics of it...

Chapter 3

Harmonic Sounds

Chapter 4

Inharmonic Sounds

Part II

Fundamental Modules

Chapter 5

Basic Concepts

Chapter 6

Trigger

Chapter 7

Create

Chapter 8

Modify

Part III

Sound Design 1

Chapter 9

Kick & Cymbal

Chapter 10

Lead & Bass

Part IV

Complex Modules

Chapter 11

Trigger

Chapter 12

Create

Chapter 13

Modify

Part V

Sound Design 2

Chapter 14

Minimoog & 303

Bibliography

Bell, T. (2021). CS unplugged or coding classes? *Communications of the ACM*, 64(5):25–27.

Bjørn, K. and Meyer, C. (2018). *Patch & Tweak: Exploring Modular Synthesis*. Bjooks. Google-Books-ID: xmrVvQEACAAJ.

Bode, H. (1984). History of electronic sound modification. 32(10):730–739.

Crombie, D. (1982). *The Complete Synthesizer: A Comprehensive Guide*. Omnipress Books.

Doepfer Musikelektronik (2022a). Construction Details A-100.

Doepfer Musikelektronik (2022b). Technical details A-100.

Dudley, H. and Tarnoczy, T. H. (1950). The speaking machine of Wolfgang von Kempelen. 22(2):151–166. Publisher: Acoustical Society of America.

Dusha, T., Martonova, A., and Johnston, P. (2020). *Modular Sound Synthesis On the Moon*. Modular Moon. Google-Books-ID: llA4zgEACAAJ.

Eliraz, Z. (2022). Loopop's In-Complete Book of Electronic Music.

Michie, D. (1963). Experiments on the Mechanization of Game-Learning Part I. Characterization of the Model and its parameters. 6(3):232–236.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, New York.

Polya, G. (2004). *How to solve it: A new aspect of mathematical method*. Princeton University Press.

Strange, A. (1983). *Electronic Music: Systems, Techniques, and Controls*. William C Brown Publishers, 2 edition.

Tedre, M. and Denning, P. J. (2016). The long quest for computational thinking. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, Koli Calling '16, page 120–129, New York, NY, USA. Association for Computing Machinery.

Tucker, A. (2003). A model curriculum for K–12 computer science: Final report of the ACM K–12 task force curriculum committee. Technical report, New York, NY, USA.