Contents

Foreword: New Music and Science

With the use of computers and digital devices, the processes of music composition and its production have become intertwined with the scientific and technical resources of society to a greater extent than ever before. Through extensive application of computers in the generation and processing of sound and the composition of music from levels of the microformal to the macroformal, composers, from creative necessity, have provoked a robust interdependence between domains of scientific and musical thought. Not only have science and technology enriched contemporary music, but the converse is also true: problems of particular musical importance in some cases suggest or pose directly problems of scientific and technological importance, as well. Each having its own motivations, music and science depend on one another and in so doing define a unique relationship to their mutual benefit.

The use of technology in music is not new; however, it has reached a new level of pertinence with the rapid development of computer systems. Modern computer systems encompass concepts that extend far beyond those that are intrinsic to the physical machines themselves. One of the distinctive attributes of computing is programmability and hence programming languages. High-level programming languages, representing centuries of thought about thinking, are the means by which computers become accessible to diverse disciplines.

Programming involves mental processes and rigorous attention to detail not unlike those involved in composition. Thus, it is not surprising that composers were the first artists to make substantive use of computers. There were compelling reasons to integrate some essential scientific knowledge and concepts into the musical consciousness and to gain competence in areas which are seemingly foreign to music. Two reasons were (and are) particularly compelling: (1) the generality of sound synthesis by computer, and (2) the power of programming in relation to the musical structure and the process of composition.

Sound Synthesis

Although the traditional musical instruments constitute a rich sound space indeed, it has been many decades since composers' imaginations have conjured up sounds based on the interpolation and extrapolation of those found in nature but that are not realizable with acoustical or analog electronic instruments. A loudspeaker controlled by a computer is the most general synthesis medium in existence. Any sound, from the simplest to the most complex, that can be produced through a loudspeaker can be synthesized with this medium. This generality of computer synthesis implies an extraordinarily larger sound space, which has an obvious attraction to composers. The reason is that computer sound synthesis is the bridge between that which can be imagined and that which can be heard.

With the elimination of constraints imposed by the medium on sound production, there nonetheless remains an enormous barrier that the composer must overcome in order to make use of this potential. That barrier is the lack of knowledge—the knowledge required for the composer to be able to effectively instruct the computer in the synthesis process. To some extent this technical knowledge relates to computers; this is rather easily acquired. But it mostly has to do with the physical description and perceptual correlates of sound. Curiously, the knowledge required does not exist, for the most part, in those areas of scientific inquiry where one would most expect to find it, that is, physical acoustics and psychobiology; these disciplines tend to provide either inexact or no data at those levels of detail with which a composer is ultimately most concerned. In the past, scientific data and conclusions were used in attempts to replicate natural sounds as a way of gaining information about sound in general. Musicians and musician-scientists were quick to point out that most of the conclusions and data were insufficient. The synthesis of sounds that approach in aural complexity the simplest natural sound demands detailed knowledge about the temporal evolution of the various components of the sound.

Physics, psychology, computer science, and mathematics have, however, provided powerful tools and concepts. When these concepts are integrated with musical knowledge and aural sensitivity, they allow musicians, scientists, and technicians, working together, to carve out new concepts and physical and psychophysical descriptions of sound at levels of detail that are of use to the composer in meeting the exacting requirements of the ear and imagination.

As this book shows, some results have emerged: there is a much deeper understanding of timbre, and composers have a much richer sound palette

with which to work; new efficient synthesis techniques have been discovered and developed that are based on modeling the perceptual attributes of sound rather than the physical attributes; powerful programs have been developed for the purposes of editing and mixing synthesized and/or digitally recorded sound; experiments in perceptual fusion have led to novel and musically useful research in sound source identification and auditory images; and finally, special-purpose computer-synthesizers are being designed and built. These real-time performance systems incorporate many advances in knowledge and technique.

Programming and Composition

Because one of the fundamental assumptions in designing a computer programming language is generality, the range of practical applications of any given high-level language is enormous and obviously includes music. Programs have been written in a variety of programming languages for various musical purposes. Those that have been most useful and with which composers have gained the most experience are programs for the synthesis and processing of sound and programs that translate musical specifications of a piece of music into physical specifications required by the synthesis program.

The gaining of some competence at programming can be rewarding to a composer because it is the key to a general understanding of computer systems. Although systems are composed of programs of great complexity and written using techniques not easily learned by nonspecialists, programming ability enables the composer to understand the overall workings of a system to the extent required for its effective use. Programming ability also gives the composer a certain independence at those levels of computing in which independence is most desirable: synthesis. Similar to the case in traditional orchestration, the choices made in the synthesis of tones, having to do with timbre and microarticulation, are often highly subjective. The process is greatly enhanced by the ability of the composer to alter synthesis algorithms freely.

The programming of musical structure is another opportunity that programming competence can provide. To the extent that compositional processes can be formulated in a more or less precise manner, they may be implemented in the form of a program. A musical structure that is based upon some iterative process, for example, might be appropriately realized by means of programming.

But there is a less tangible effect of programming competence that results from the contact of the composer with the concepts of a programming language. Whereas the function that a program is to perform can influence the

choice of language in which the program is written, it is also true that a programming language can influence the conception of a program's function. In a more general sense, programming concepts can suggest functions that might not occur to one outside of the context of programming. This is of signal importance in music composition, because the integration of programming concepts into the musical imagination can extend the boundaries of the imagination itself. That is, the language is not simply a tool with which some preconceived task or function can be accomplished; it is an extensive basis of structure with which the imagination can interact, as well. Although computer synthesis of sound involves physical and psychophysical concepts derived from the analysis of natural sounds, when joined with higher-level programming of musical structure the implications extend far beyond timbre. Unlike the condition that exists in composition for traditional instruments under which the relation of vibrational modes of an instrument is largely beyond compositional influence, computer synthesis allows for the composition of music's microstructure.

In the context of computing, then, the microstructure of music is not necessarily of predetermined form, that is, associated with a specific articulation of a particular instrument. Rather, it can be subjected to the same thought processes and be as freely determined in the imagination of the composer as every other aspect of the work.

John Chowning

Preface to the Second Edition

Here is *The Computer Music Tutorial, Second Edition.* When we first submitted a book proposal for *The Computer Music Tutorial* to the MIT Press in 1979, I was editor of *Computer Music Journal.* Three previous editors joined me in this proposal. John Snell was a music hardware engineer, and Curtis Abbott and John M. Strawn were digital signal processing mavens. My research interests at the time revolved around algorithmic composition and granular synthesis. Our general plan had been to divide up the subject matter according to each author's interests and expertise. However, the project went slower than expected, and by 1983, my partners had dropped out due to better opportunities. I was faced with learning about areas of research that were unfamiliar to me. After a sustained period during which I was also working full time, I finished the manuscript in Paris in 1993.

The production process was long, and the book did not appear until 1996. I planned the original as a three-volume set; however, The MIT Press favored a single volume. The size and weight of the book was a practical problem, although it did not deter sales. The scope reflected my insistence on breadth. I felt it important to represent many directions of the field and not just a select few, as favored by certain people and institutions.

I have always been interested in the legacy of electronic and computer music, so a strong thread of the first edition was the history of the field. This historical scholarship was a major endeavor in itself and runs throughout the text.

Later, for the French edition published by Dunod, my colleague Jean de Reydellet suggested a more logical organization into many short chapters (Roads 1998). We used this organization in the second and third French editions (Roads 2007, 2016). I have adopted a similar organization here.

Many chapters can be read independently of one another. However, in certain cases it is helpful to read one before another. This is mentioned in the relevant chapters.

Publishing Context

In this revised edition, the reader will find new chapters and also extensive updates based on recent research. In certain cases, it was necessary to reframe the discussion in the light of all that has transpired since the original edition was published.

Electronic and computer music is developing rapidly. The field rides waves of technological innovation. Faster processors and networks, better displays, and clever controllers have all made their mark. Improvements in audio equipment advance the field. These create a hardware foundation for innovations in software. Today, thousands of companies compete to deliver products to musicians.

Many fundamentals, however, have not changed. Vintage gear and carefully preserved software from decades past can sound wonderful today. The laws of physics that govern digital signal processing are immutable. Many new products are clever reworkings of ancient principles dressed up in fresh interfaces. Yet the power of an interface is not to be underestimated. A novel design can completely transform the method of working. The digital audio workstation (DAW) with its graphical time line interface for editing and mixing was one such innovation. Graphical patching of modules was another (e.g., Max), as was the integration of real-time spectral displays. The modular Eurorack format inspired a giant wave of creativity in synthesis and control. In each of these cases, a novel step quickly propagated and changed the field.

When it was published, *The Computer Music Tutorial* was one of only a handful of books devoted to the subject. Many more books have since been published. If we look solely at the catalog of MIT Press books published since 1996 and narrow it to technical books on computer music, we see a range of texts that expand on subtopics introduced in this volume: *Composing Interactive Music, Microsound, The Audio Programming Book, The Csound Book, The SuperCollider Book, Designing Sound, Sonic Interaction Design, Virtual Music, Music Cognition and Computerized Sound, Computer Models of Musical Creativity, Musimathics 1 and 2, Beyond MIDI, Music Query, Music and Probability,* and *Musical Networks.* Dozens of other books are available from different publishers.

Publishing has been profoundly transformed by the internet. The internet is an extraordinary resource if you know what to look for. A goal of this text is to foster this curiosity. Some 2,000 references support the descriptions.

What Is New in the Second Edition?

To begin, this edition incorporates corrections. Second, after an extensive review of the literature, the content has been thoroughly updated and rewritten for greater clarity. This includes hundreds of new figures and hundreds of new references.

New chapters include virtual analog, pulsar synthesis, concatenative synthesis, spectrum analysis by atomic decomposition, Open Sound Control, spectrum editors, instrument and patch editors, and an appendix on machine learning.

New sections cover MIDI 2.0, piano models, single-sideband modulation, wavefolding, dynamic convolution, immersive sound (VBAP, ambisonics, and wave field synthesis), sidechain control and adaptive effects, additive synthesis based on machine learning, transmission formats for multichannel sound, filter banks and vocoders, fractal interpolation synthesis, scanned synthesis, digital audio workstations and audio middleware, live coding and live notation, and telematic music.

Topics Deleted or Omitted

The original edition had an introductory chapter on computer programming as a general topic. Although it was well-intended, programming is hard to summarize in a chapter. Many tomes cover programming in depth and detail. Identical considerations came into play when I considered a chapter specifically on audio programming. By now many books focus specifically on this topic. For example, *The Audio Programming Book* (Boulanger and Lazzarini 2011) contains over 3,000 pages of text and thousands of lines of code, including code listings for Csound, cmusic, and Music V. Each audio language has its own reference text. Among these are *The Csound Book* (Boulanger 2000), *The SuperCollider Book* (Wilson, Cottle, and Collins 2011), *Programming for Musicians and Digital Artists: Creating Music with ChucK* (Kapur et al. 2015), *Electronic Music and Sound Design, Volumes 1 and 2* (Max) (Cipriani and Giri 2010a, b), *The Theory and Technique of Electronic Music* (PureData) (Puckette 2007), *Designing Audio Objects for Max/MSP and Pd* (Lyon 2012), *Hack Audio* (MATLAB) (Tarr 2019), and *Nyquist Reference Manual* (Dannenberg 2018), among others. Languages such as Faust and AudioKit are documented online. Practical how-to texts such as *Linux Sound Programming* (Newmarch 2017), *Designing Audio Effects Plugins in C++*(Pirkle 2019) and *Designing Software Synthesizer Plugins in C++*(Pirkle 2015) are available. See also *Generating Sound & Organizing Time: Thinking with gen~* (Wakefield and

Taylor 2022). The Audio Developer Conference is another resource, as is the Audio Programmer video channel on YouTube.

Thus our presentation of audio programming in chapter 8 is merely a pointer to a large domain.

The original edition had an appendix on psychoacoustics, a topic that is not specific to computer music. Other texts offer a more comprehensive treatment of this area (Loy 2006; Howard and Angus 2017; Bader 2018).

The old appendix on the mathematics of Fourier analysis is gone, but I merged the essential information into chapter 37, "Spectrum Analysis by Fourier Methods." For those who want a more extended treatment, Loy (2007) devotes over one hundred pages to the same subject; Smith (2011) offers nearly six hundred pages.

I excised the dated chapter "Internals of Digital Signal Processors" with regret, especially the historical section. The pioneers of *digital signal processing* (DSP) hardware in the 1970s and 1980s, such as Sydney Alonso, Harold Alles, Peter Samson, and Giuseppe Di Giugno, were heroes of synthesizer engineering. Fortunately, many of these stories are told elsewhere. Joel Chadabe (1997) tells the saga of the Synclavier and several other pioneering systems. Mark Vail (2000a, b) recounts the early days of analog and digital synthesis. Bjørn and Meyer (2018) interview several modular synthesizer developers. Loy (2013a, b) tells the story of the Samson Box from an historical perspective. Giordano (2020) documents the colorful career of Di Giugno.

The situation of digital audio hardware today is quite different. DSP chips can be found in audio effects hardware, synthesizers from Yamaha, Roland, and other companies, digital Eurorack modules, and unique products such as the Kyma/Pacarana and the Universal Audio Apollo interfaces. However, most audio software today runs on standard microprocessors without DSP hardware support. Thus the topic of DSP architecture per se did not seem as central to this book as it did in the 1990s, when many personal computers had a dedicated DSP circuit board for audio processing.

Ironically, present-day audio computing faces a challenge, as *central processing unit* (CPU) clock speeds stalled years ago (Asanovíc et al. 2006). The short-term solution, adding more cores to the CPU chip, does not benefit real-time audio processing, which is not ideally suited to multicore architectures (Thall 2019). As a solution, some have predicted that audio DSP chips could make a comeback even as *graphical programming units* (GPUs) and *tensor processing units* (TPUs) are also being deployed for audio (Storer 2018; Anderson 2020). However, if these chips do not become standard, their impact will be limited. Time will tell.

At one point I thought it would be useful to have a survey chapter on the broad area of *music information retrieval* (MIR). Fortunately, a textbook called

Fundamentals of Music Processing (Müller 2015) appeared, which covers this topic in a tutorial manner. Müller's book describes specific methods for analysis-based applications, discussed in part V of this book. These include score following, parsing music structure, chord recognition, tempo and beat tracking, content-based audio retrieval, harmonic-percussive separation, and melody tracking. An associated website provides example code in Python (https://www.audiolabs-erlangen.de/news/articles/FMP), and Lerch (2012) covers related subjects. Also refer to Polotti and Rocchesso (2008).

Certain chapters of the book mention mechanical automation, but a full-bore treatment of the topic of musical robots was not possible. Mechanical automation has a deep history dating to the age of eighteenth-century *androids* built by Jacques de Vaucanson (Sousa 1906; Leichtentritt 1934; Ord-Hume 1973, 1984; Buchner 1978; Kapur 2005). Although some musical robots are little more than mechanical sequencers (such as player pianos), others are capable of human interaction by means of machine listening. Specialized conferences and books such as those by Solis and Ng (2011) focus entirely on this area of experimentation.

In the original edition, I did not specifically focus on composition. Neither do I here. It is a topic that demands its own platform, and to this end I have devoted a part of my book *Microsound* (Roads 2001a) and all of *Composing Electronic Music: A New Aesthetic* (Roads 2015) to this end.

Intended Audience

The intended audience of *The Computer Music Tutorial, Second Edition* is the same as the original: music students, but also engineers and scientists seeking orientation to computer music. The word *orientation* is key, as is the word *tutorial* in the title.

Many of the topics discussed in the fifty-two chapters herein could be expanded into book-length treatises. The goal of this book was the opposite: to sift through the research literature, sort out the most fundamental facts, and craft a clear and concise technical narrative that would be understandable by a novice. This tutorial is not intended as a comprehensive resource for advanced developers of computer music algorithms. We aim to introduce the field, explain its motivations, put topics into context, and provide references for further study.

For more than twenty-five years I have used *The Computer Music Tutorial* in my year-long introductory course on electronic and computer music. Judging from this experience, I can testify that the pace and the level of this book are well matched to an introductory course.

Generic versus Specific

In the first edition of the book I was careful to make the explanations generic and not tied to specific hardware and software, which can become obsolete. The field is more stable now. Thus in this revised edition, I still keep the theory generic, but I decided to cite more products as examples in order to make the descriptions more concrete. Many of these products have been around for more than two decades. Of course, although certain products are mentioned as examples, this text is not meant to be a product survey.

Diversity

One of the reviewers of the manuscript brought up the thorny issue of diversity in computer music. This has long been a concern in the field, as pointed by Mary Simoni's 1995 survey of gender issues in electronic and computer music. These concerns echo in more recent studies such as Xambó (2018) and Sofer (2022). Similar issues can be raised concerning other underrepresented groups. *The Computer Music Tutorial, Second Edition* is focused primarily on technical research. It reflects the scholarly literature, which historically has not been as diverse as one would hope. Thus I have made an explicit effort to mention and cite diverse contributors when possible.

Acknowledgments

The Computer Music Tutorial was published immediately before I started teaching at the University of California, Santa Barbara. I would like to thank my UCSB colleagues at the Center for Research in Electronic Art Technology (CREATE). I also greatly appreciate the support from my colleagues in the Media Arts and Technology (MAT) Graduate Program and the Department of Music at UCSB.

For this edition, my former student and research colleague, Professor Bob L. T. Sturm (KTH Royal Institute of Technology, Stockholm) kindly agreed to assist with revisions. Specifically, he revised several chapters and contributed two new chapters.

I would also like to thank my former CREATE colleague Dr. Mathew J. Wright (Center for Computer Research in Music and Acoustics, Stanford) for his collaboration on chapter 52, "Open Sound Control."

Chapter reviewers included Aaron Anderson, Clarence Barlow, Stefan Bilbao, Thom Blum, Ludger Brümmer, Nick Collins, Jean de Reydellet, Rodney Duplessis, Kramer Elwell, Stewart Engart, Tom Erbe, Yuan-Yi Fan, Susan Frykberg, Stefanie Ku, JoAnn Kuchera-Morin, Elizabeth Hambleton, Michael Hetrick, Francisco Iovino, Christopher Jette, Garry Kling, Lawrence Kolasa, Ryan McGee, João Pedro Oliveira, Brian O'Reilly, Robert Owens, Chris Ozley, Brandon Rolle, David Romblom, Giorgio Sancristoforo, Ron Sedgwick, Atau Tanaka, Bruce Wiggins, Michael Winter, and Karl Yerkes. Dr. Tim Wood wrote scripts to verify the citations and references. Thanks to Federico Llach for his consultation on the Max example in chapter 45.

This book includes corrections to the original edition kindly supplied by Keiji Hirata, Takafumi Hikichi, James McCartney, and Graham Hadfield.

I am grateful to Professor Keiji Hirata of the University of Tokyo for organizing the Japanese edition, which appeared in 2001. Alongside Professor Hirata were translators Tatsuya Aoyagi, Naotoshi Osaka, Masataka Goto, Takefumi Hikichi, Saburo Hirano, Yasuo Horiuti, and Toshiaki Matsushima.

I thank Dr. Ken Fields, Adjunct Professor in Media Arts and Technology at UCSB and Professor at the Central Conservatory of Music in Beijing for organizing the Chinese edition, which appeared in 2011. I thank the translators of the Chinese edition, Chang Wei, Chen Yang, Cheng Yibing, Hu Ze, Huang Zhipeng, Jiang Hao, Li Sixin, Li Yueling, Qi Gang, Yang Renying, Zhang Ruibo (Mungo), and the proofreaders, Jin Ping, Li Sixin, and Qi Gang.

Finally, I would like to thank the teams at MIT Press and Westchester Publishing Services for their assistance with the production of this book.

Preface to the Original Edition (1996)

Music changes: new forms appear in infinite variety, and reinterpretations infuse freshness into old genres. Waves of musical cultures overlap, diffusing new stylistic resonances. Techniques for playing and composing music meander with these waves. Bound with the incessant redevelopment in music making is an ongoing evolution in music technology. For every music there is a family of instruments, so that today we have hundreds of instruments to choose from, even if we restrict ourselves to the acoustic ones.

In the twentieth century, electronics turned the stream of instrument design into a boiling rapids. Electrification transformed the guitar, bass, piano, organ, and drum (machine) into the folk instruments of industrial society. Analog synthesizers expanded the musical sound palette and launched a round of experimentation with sound materials. But analog synthesizers were limited by a lack of programmability, precision, memory, and intelligence. By virtue of these capabilities, the digital computer provides an expanded set of brushes and implements for manipulating sound color. It can listen, analyze, and respond to musical gestures in sophisticated ways. It lets musicians edit music or compose according to logical rules and print the results in music notation. It can teach interactively and demonstrate all aspects of music with sound and images. New musical applications continue to spin out of computer music research.

In the wake of ongoing change, musicians confront the challenge of understanding the possibilities of the medium and keeping up with new developments. *The Computer Music Tutorial* addresses the need for a standard and comprehensive text of basic information on the theory and practice of computer music. As a complement to the reference volumes *Foundations of Computer Music,* (edited with John M. Strawn, MIT Press, 1985) and *The Music Machine* (MIT Press, 1989), this book provides the essential background necessary for advanced exploration of the computer music field. While *Foundations of Computer Music* and *The Music Machine* are anthologies, this textbook contains all new material directed toward teaching purposes.

Intended Audience

The intended audience for this book is not only music students but also engineers and scientists seeking an orientation to computer music. Many sections of this volume open technical "black boxes," revealing the inner workings of software and hardware mechanisms. Why is technical information relevant to the musician? Our goal is not to turn musicians into engineers but to make them better informed and more skillful users of music technology. Technically naive musicians sometimes have unduly narrow concepts of the possibilities of this rapidly evolving medium; they may import conceptual limitations of bygone epochs into a domain where such restrictions no longer apply. For want of basic information, they may waste time dabbling, not knowing how to translate intuitions into practical results. Thus one aim of this book is to impart a sense of independence to the many musicians who will eventually set up and manage a home or institutional computer music studio.

For some musicians, the descriptions herein will serve as an introduction to specialized technical study. A few will push the field forward with new technical advances. This should not surprise anyone who has followed the evolution of this field. History shows time and again that some of the most significant advances in music technology have been conceived by technically informed musicians.

Interdisciplinary Spirit

The knowledge base of computer music draws from composition, acoustics, psychoacoustics, physics, signal processing, synthesis, performance, computer science, and electrical engineering. Thus, a well-rounded pedagogy in computer music must reflect an interdisciplinary spirit. In this book, musical applications motivate the presentation of technical concepts, and the discussion of technical procedures is interspersed with commentary on their musical significance.

Heritage

One goal of our work has been to convey an awareness of the heritage of computer music. Overview and background sections place the current picture into historical context. Myriad references to the literature point to sources for further study and also highlight the pioneers behind the concepts.

Concepts and Terms

Every music device and software package uses a different set of protocols—terminology, notation system, command syntax, button layout, and so on. These differing protocols are built on the fundamental concepts explained in this volume. Given the myriad incompatibilities and the constantly changing technical environment, it seems more appropriate for a book to teach fundamental concepts than to spell out the idiosyncrasies of a specific language, software application, or synthesizer. Hence, this volume is not intended to teach the reader how to operate a specific device or software package—that is the goal of the documentation supplied with each system. But it will make this kind of learning much easier.

Use of This Book in Teaching

The Computer Music Tutorial has been written as a general textbook, aimed at presenting a balanced view the field in its current state. It is designed to serve as a core text and should be easily adaptable to a variety of teaching situations. In the ideal situation, this book should be assigned as a reader in conjunction with a studio environment where students have ample time to try out the various ideas within. Every studio favors particular tools (such as computers, software, synthesizers, etc.), so the manuals for those tools, along with studio-based practical instruction, should round out the educational equation.

Composition

Notwithstanding the broad scope of this book, it was impossible to compress the art of composition into a single part. Instead, readers will find many citations to composers and musical practices interwoven with technical discussions. Chapters 18 and 19 present the technical principles behind algorithmic composition, but this is only one facet of a vast—indeed open-ended—discipline, and is not necessarily meant to typify computer music composition as a whole.

We have surveyed composition practices in other publications. *Composers and the Computer* focuses on several musicians (Roads1985a). During my tenure as the editor of *Computer Music Journal* (1978–1989), we published many reviews of compositions as well as interviews with and articles by composers. These include a "Symposium on composition," with fourteen

composers participating (Roads 1986b), and a special issue on composition, 5(4) 1981. Some of these articles are available in a widely available text, *The Music Machine* (MIT Press 1989). Issue 11(1) 1987 featured microtonality in computer music composition. Many other periodicals and books contain informative articles on compositional issues in electronic and computer music.

References and Index

In a tutorial volume that covers many topics, it is essential to supply pointers for further study. This book contains extensive citations and a reference list of more than 1,400 entries compiled at the back of the volume. As a further service to readers, we have invested much time to ensure that both the name and subject indexes are comprehensive.

Mathematics and Coding Style

Since this *Tutorial* is addressed primarily to a musical audience, we chose to present technical ideas in an informal style. The book uses as little mathematical notation as possible. It keeps code examples brief. When mathematical notation is needed, it is presented with operators, precedence relations, and groupings specified explicitly for readability. This is important because the idioms of traditional mathematical notation are sometimes cryptic at first glance or are incomplete as algorithmic descriptions. For the same reasons, the book usually uses long variable names instead of the single-character variables favored in proofs. With the exception of a few simple Lisp examples, code examples are presented in a Pascal-like pseudocode.

Corrections and Comments Invited

In a large book covering a new field, there will inevitably be errors. We welcome corrections and comments, and we are always seeking further historical information. Please send comments and corrections via email to the author at clangtint@gmail.com.

Acknowledgments

This book was written over a period of many years. I wrote the first draft from 1980 to 1986, while serving as a research associate in computer music at the Massachusetts Institute of Technology and as editor of *Computer Music Journal* for The MIT Press. I am grateful to many friends for their assistance during the period of revisions that followed.

Major sections of part IV ("Mixing and Signal Processing") and part V ("Sound Analysis") were added during a 1988 stay as visiting professor in the Department of Physics at the Università di Napoli Federico II, thanks to an invitation by Professor Aldo Piccialli. I am deeply grateful to Professor Piccialli for his detailed comments and generous counsel on the theory of signal processing.

Valuable feedback on part III ("Sound Synthesis") came from composition students in the Department of Music at Harvard University, where I taught in 1989, thanks to Professor Ivan Tcherepnin. I thank Professors Conrad Cummings and Gary Nelson for the opportunity to teach at the Oberlin Conservatory of Music in 1990, where I presented much of the book in lecture form, which led to clarifications in the writing.

During spare moments I worked on part VI ("The Musician's Interface") in Tokyo at the Center for Computer Music and Music Technology, Kunitachi College of Music, in 1991, thanks to the center's director Cornelia Colyer, Kunitachi chairman Bin Ebisawa, and a commission for a composition from the Japan Ministry of Culture. Final refinements to the book were carried out in Paris. I presented the first courses based on the completed text in 1993 and 1994 at Les Ateliers UPIC, thanks to Gerard Pape and Iannis Xenakis, and the Music Department of the University of Paris 8, thanks to Professor Horacio Vaggione.

John M. Strawn, formerly my editorial colleague at *Computer Music Journal,* contributed substantially to this project for several years. In between his duties as a doctoral student at Stanford University, he contributed much to part I ("Digital Audio"). Later, he reviewed drafts of most chapters with characteristic thoroughness. Throughout this marathon effort, John consulted on myriad details via electronic mail. I am grateful to him for sharing his wide musical and technical knowledge and sharp wit.

Many kind individuals helped by supplying information, documentation, and photographs or by reading chapter drafts. I am profoundly indebted to these generous people for their myriad suggestions, criticisms, and contributions to this book: Jean-Marie Adrien, Jim Aiken, Clarence Barlow, François Bayle, James Beauchamp, Paul Berg, Nicola Bernardini, Peter Beyls, Jack

Biswell, Thom Blum, Richard Boulanger, David Bristow, William Buxton, Wendy Carlos, René Caussé, Xavier Chabot, John Chowning, Cornelia Colyer, K. Conklin, Conrad Cummings, James Dashow, Philippe Depalle, Mark Dolson, Giovanni De Poli, Gerhard Eckel, William Eldridge, Gianpaolo Evangelista, Ayshe Farman-Farmaian, Adrian Freed, Christopher Fry, Guy Garnett, John W. Gordon, Philip Greenspun, Kurt Hebel, Henkjan Honing, Gottfried Michael Koenig, Paul Lansky, Otto Laske, David Lewin, D. Gareth Loy, Max V. Mathews, Stephen McAdams, Dennis Miller, Diego Minciacchi, Bernard Mont-Reynaud, Robert Moog, F. R. Moore, James A. Moorer, Peter Nye, Robert J. Owens, Alan Peevers, Aldo Piccialli, Stephen Pope, Edward L. Poulin, Miller Puckette, Thomas Rhea, Jean-Claude Risset, Craig Roads, Xavier Rodet, Joseph Rothstein, William Schottstaedt, Marie-Hélène Serra, John Snell, John Stautner, Morton Subotnick, Martha Swetzoff, Karen Tanaka, Stan Tempelaars, Daniel Teruggi, Irène Thanos, Barry Truax, Alvise Vidolin, Dean Wallraff, David Waxman, Erling Wold, and Iannis Xenakis.

I would also like to express my thanks to the staff of The MIT Press Journals—Janet Fisher, manager—publishers of *Computer Music Journal.* This work would have been nigh impossible without their backing over the past fourteen years.

I will always be grateful to Frank Urbanowski, director of The MIT Press, and executive editor Terry Ehling for their extraordinarily patient and kind support of this project.

I am also indebted to Sandra Minkkinen and the production staff of The MIT Press for their fine editing and production labors.

This book is dedicated to my mother, Marjorie Roads.

I Digital Audio

1 *History of Digital Audio*

History of Analog Audio Recording

Experimental Digital Audio Recording

Digital Sound for the Public

- **Digital Audio Discs**
- **Downloadable Audio File Formats**

Digital Sound for Musicians

Origins of Digital Multitrack Recording

The Art of Recording

This chapter presents a basic introduction to the history and technology of digital audio recording and playback.

History of Analog Audio Recording

Sound recording has a rich history, beginning with Léon Scott's phonautograph of 1857, which could record a sound waveform but not play it back. Thomas Edison's experiments in the 1870s combined sound recording with playback on wax cylinders. Emile Berliner's gramophone (1887) recorded on rotating discs, the precursor to the long-play (LP) vinyl discs still being made (Read and Welch 1976). Early audio recording was a mechanical process (figure 1.1). Air vibrations caused a membrane to vibrate, and these vibrations were traced in a soft medium such as a rotating wax cylinder by a stylus attached to the membrane.

Although the invention of the triode vacuum tube in 1906 launched the era of electronics, electronically produced recordings did not become practical until 1924 (Keller 1981). Figure 1.2 depicts one of the horn-loaded loudspeakers that were common in the 1920s.

Optical sound recording on film was first demonstrated in 1922 (Ristow 1993). Sound recording on tape coated with powdered magnetized material was developed in the 1930s in Germany (figure 1.3) but did not reach the rest of the world until after World War II. The German

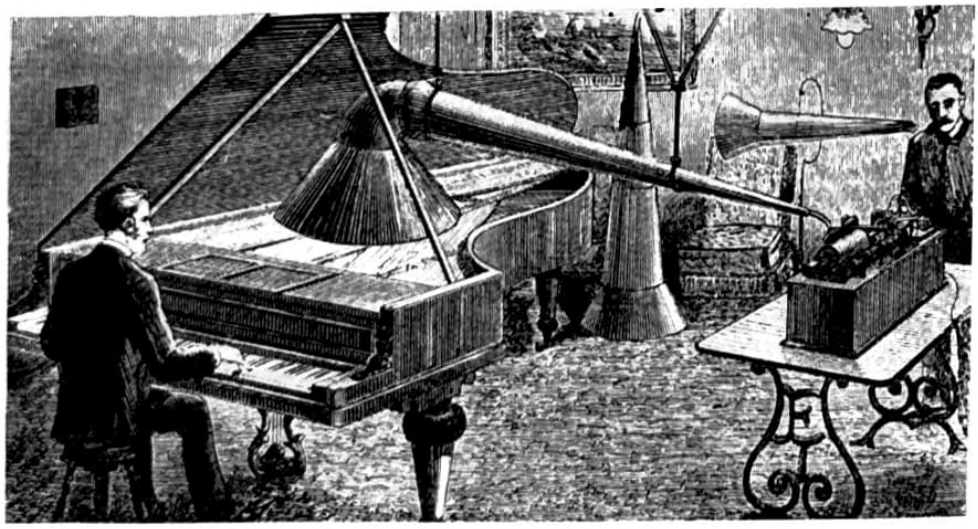

Figure 1.1 Mechanical recording session before 1900. Sound vibrations picked up by the large cone over the piano were transduced into vibrations of a cutting stylus piercing a rotating wax cylinder.

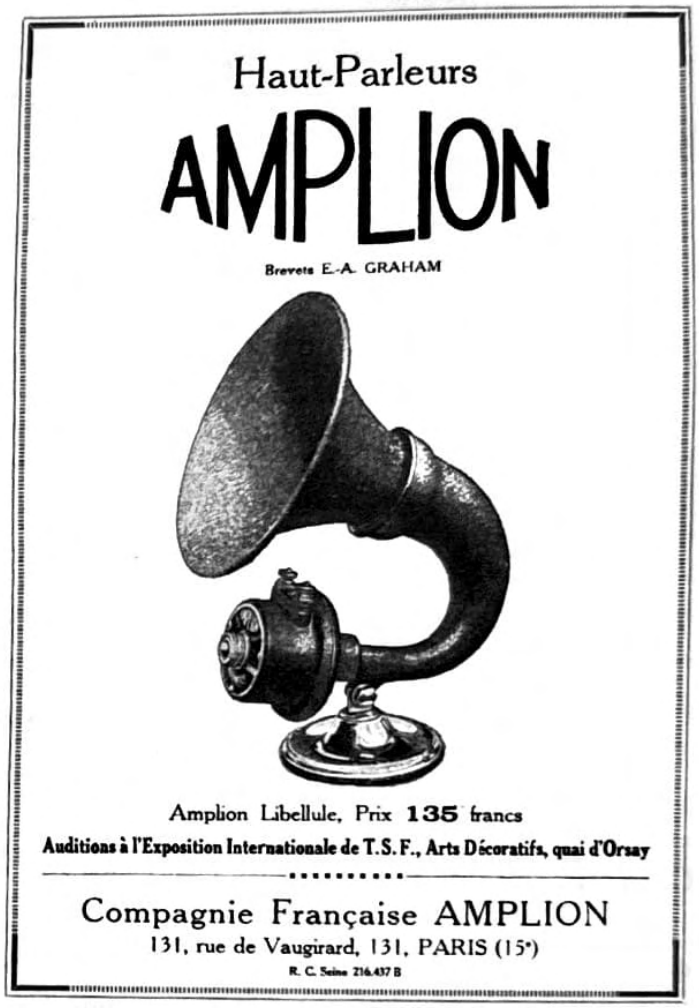

Figure 1.2 Amplion loudspeaker, as advertised in 1925.

magnetophon tape recorders were a great advance over previous wire and steel band recorders, which required hard soldering or welding to make a splice.

The magnetophons and their descendants were *analog* recorders, so called because the waveform encoded on tape is a close *analogy* to the original sound waveform picked up by a microphone. If we could view the magnetized particles on the tape, they would form a pattern resembling the sound waveform. Analog recording is still favored by some for its special quality of sound. However, analog recording faces fundamental physical limits. These limits are most apparent in copying from one analog medium to another—additional noise is inescapable.

For more on the history of analog recording, see chapter 26 on mixing.

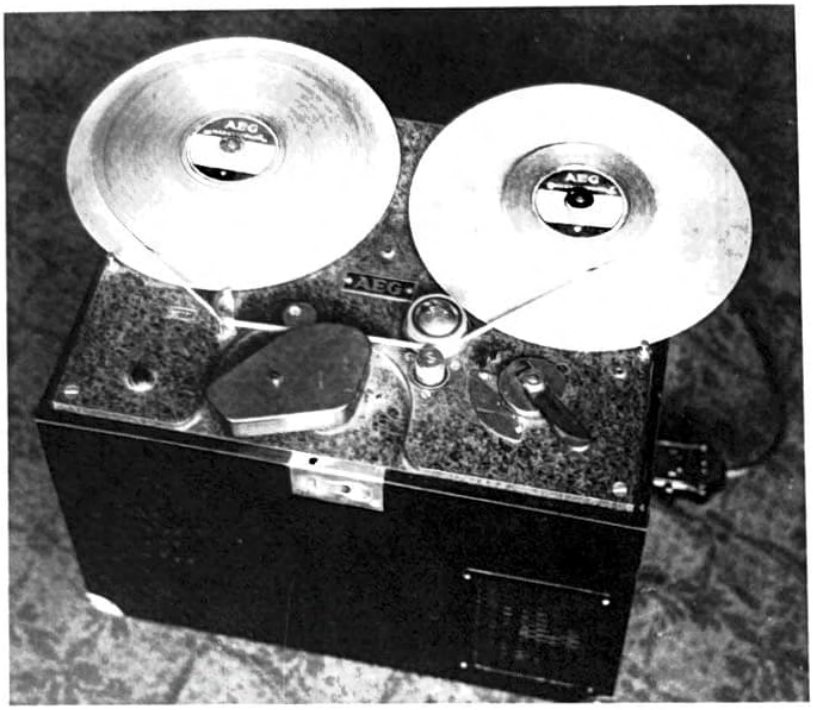

Figure 1.3 Prototype of a portable *magnetophon* tape recorder from 1935, made by AEG. (Photograph courtesy of BASF Aktiengesellschaft.)

Experimental Digital Audio Recording

The core concept in digital audio recording is *sampling*, that is, converting continuous analog signals (such as those coming from a microphone) into discrete *time-sampled* signals (*samples*). Each sample is nothing more than a number—a snapshot of a sound waveform (figure 1.4).

The theoretical underpinning of sampling is the *sampling theorem*, which specifies the relation between the sampling rate and the audio bandwidth. It is also called the *Nyquist theorem* after the work of Harold Nyquist of Bell Telephone Laboratories (Nyquist 1928), but another form of this theorem was first stated in 1841 by the French mathematician Augustin Louis Cauchy (1789–1857). The British researcher A. Reeves developed the first patented *pulse-code-modulation* (PCM) system for transmission of messages in *amplitude-dichotomized, time-quantized* (digital) form (Reeves 1938; Licklider 1950; Black 1953). Even today, digital recording is sometimes called *PCM recording*. The development of *information theory* contributed to the understanding of digital audio transmission (Shannon 1948). Solving the difficult problems of converting between analog signals and digital signals took decades and is still being improved. Chapters 3 and 4 describe the conversion processes.

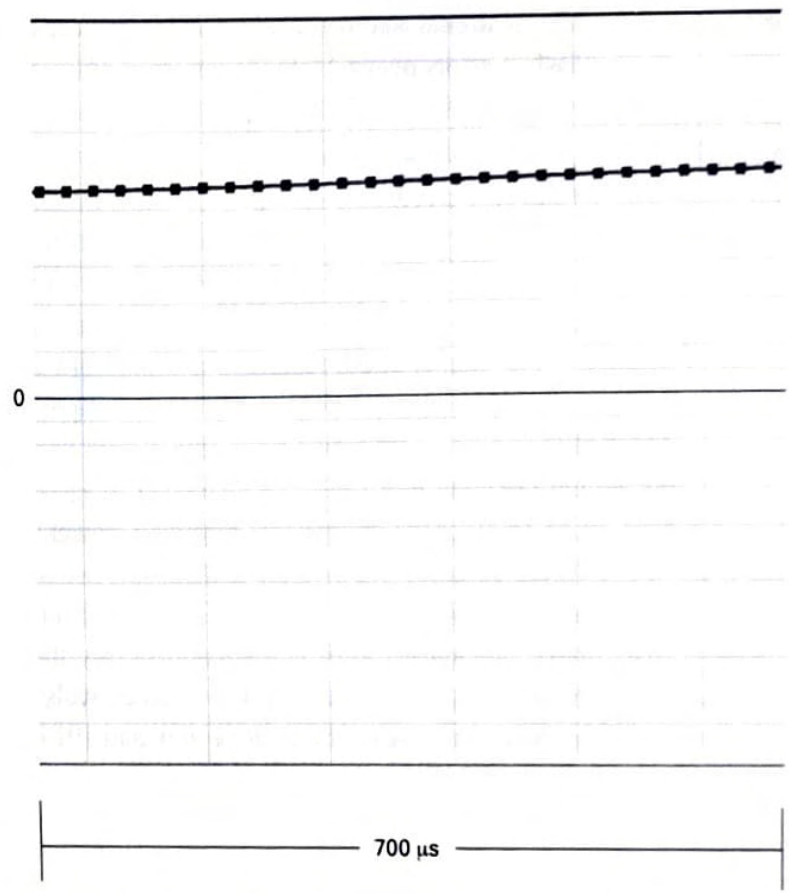

Figure 1.4 Zoomed in to show individual samples as they appear in a sound editor. The sound editor drew a line through them to enhance the display. The samples are all positive in amplitude; the line in the center represents 0 amplitude. The time span shown is about 700 μs (less than a thousandth of a second).

Primitive methods of generating sounds, such as placing a radio next to a computer and running programs that loop at audio frequencies to create melodies, date to the early days of electronic computing (Doornbusch 2005). These experiments were very limited in their sound quality.

In the late 1950s, Max Mathews and his group at Bell Telephone Laboratories generated the first sample-based sounds from a digital computer (David, Mathews, and McDonald 1959). Using samples to represent waveforms allowed generating any possible waveshape. The samples were written by the computer to expensive and bulky reel-to-reel computer tape storage drives. The production of sound from the numbers was a separate process of playing back the tape through a custom-built 12-bit vacuum tube *digital-to-sound converter* developed the Epsco Corporation (Roads 1980). Today we call a playback device that translates from digital numbers into analog voltages a *digital-to-analog converter* (DAC). By contrast, an *analog-to-digital converter* (ADC) encodes an analog waveform into digital numbers.

A practical digital audio recording medium needs to be robust, that is, resilient when errors occur. Methods for *error detection and correction* were developed in the 1950s and 1960s and widely used in communications. Later, Sato, Blesser, Stockham, and Doi applied error correction in the first practical digital audio recorders and players. The first dedicated one-channel digital audio recorder (based on a videotape mechanism) was demonstrated by NHK, the Japanese broadcasting company (Nakajima et al. 1983). Soon thereafter, Denon developed an improved version (figure 1.5), and the race began to bring digital audio recorders to market (Iwamura et al. 1973).

By 1977 the first commercial recording system appeared, the Sony PCM-1, designed to encode 13-bit digital audio signals onto Sony videocassette recorders. Within a year this was displaced by 16-bit PCM encoders such as the $40,000 Sony PCM-1600 for the professional recording market (Nakajima et al. 1978).

The Audio Engineering Society established two standard sampling frequencies in the 1980s: 44.1 and 48 kHz. This became known as the AES/EBU or AES3 standard after it was endorsed by the European Broadcast Union. Since then, the standard has been successively revised to incorporate the higher sampling rates of 88.2, 96, 176.4, and 192 kHz, and so on (Audio Engineering Society 2008). A variety of high-resolution recorders are now available, including hand-held *field recorders* with built-in microphones.

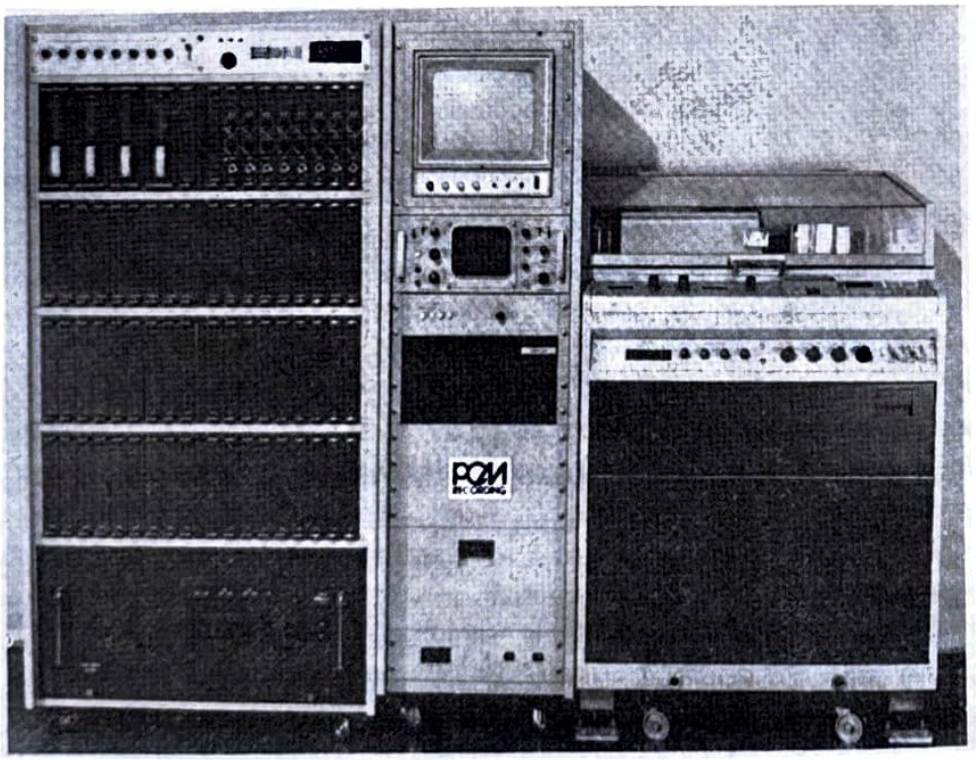

Figure 1.5 Nippon Columbia (Denon) digital audio recorder made in 1973, based on a 1-inch videotape recorder (right).

Digital Sound for the Public

In 1980, computer sound research was still in its infancy, and the standard consumer format for recorded music was the vinyl long-play (LP) record, introduced in 1948. Apart from a small number of audiophile-quality LPs, the majority of LPs manufactured by record companies in the 1970s and 1980s were not of outstanding quality, due to compromised mastering and manufacturing practices. Thus, the market was ripe for an improved format.

Digital Audio Discs

Digital sound first reached the general public in 1982 by means of the *compact disc* (CD) format, a 12 cm optical disc read by a laser. The CD format was developed jointly by the Philips and Sony corporations after years of development. It was a tremendous commercial success, selling over 1.35 million players and tens of millions of discs within two years (Pohlman 1989a). Part of the success of the CD can be traced to its adoption by the computer industry as a general means of distributing software and storing data, the CD read-only memory (CD-ROM).

Introduced in 2000, the specialized DVD-audio (DVD-A) and super audio compact disc (SACD) optical disc formats offered improved sound quality, but neither format gained a foothold in the market. The Blu-ray disc format, first introduced in 2002, is same physical size as the CD and DVD. Blu-ray supports high-definition 3D video, up to 128 gigabytes of data and up to eight channels of high-resolution digital audio (up to 24 bits per sample at a 96 kHz sampling rate). Blu-ray as an audio-only format was announced in 2013 but was not successful.

Downloadable Audio File Formats

The rise of the internet fostered the evolution of downloadable audio file formats. Throughout the 1990s, most internet users relied on slow connections based on telephone modems with sluggish transmission speeds. As a response to these limitations, the popular MP3 medium was developed under the official moniker MPEG Audio Layer III, an international standard. MP3 uses a *lossy compression* algorithm to drastically reduce the size of a sound file (Pohlmann 2005). To reduce the amount of data, audio information is discarded in the encoding phase. This results in a loss of audio fidelity.

By contrast, *lossless compression* reduces the size of the file for the purpose of storage and transmission while allowing the original file to be reconstructed

exactly. The free lossless audio codec (FLAC) is an example of a lossless compression format. Digital audio files compressed by FLAC's algorithm can be reduced in size by around 50 percent (Xiph 2015).

New media formats are constantly appearing, partly because there are commercial incentives to introduce proprietary systems. See chapter 4 for more on audio file formats.

Digital Sound for Musicians

High-quality DACs attached to personal computers came on the scene around 1988. At the same time, standard formats for audio files appeared, including Sound Designer II (SDII), audio interchange file format (AIFF), and waveform audio format (WAVE). These developments heralded a major new era for computer music. In a short period, sound synthesis, recording, and processing by personal computer became widespread. Many different *digital audio workstations* (DAWs) reached the musical marketplace, including Pro Tools, Cubase, Digital Performer, Logic, and others. DAWs let musicians record music onto a hard disk connected to a personal computer. This music could be precisely edited using a timeline display on the computer screen with playback from the hard disk.

Origins of Digital Multitrack Recording

Multitrack recorders have a number of discrete channels or *tracks* that can be recorded at different times. Each track can record, for example, a separate instrument, which allows flexibility when the tracks are later mixed together. Another advantage of multitrack machines is that they let musicians build recordings in several layers; each new layer is an accompaniment to previously recorded layers.

The British Broadcasting Company (BBC) developed an experimental 10-channel digital tape recorder in 1976. Two years later, the 3M company introduced the first commercial 32-track digital recorder (figure 1.6) as well as a rudimentary digital tape editor (Duffy 1982). The first computer disk–based random-access sound editor and mixer was developed by the Soundstream company in Salt Lake City, Utah (see figure 16.38). Their pioneering system allowed mixing of up to eight tracks or *sound files* stored on computer disk at a time (Ingebretsen and Stockham 1984).

Early digital multitrack recording was a very expensive enterprise. The Studer digital recorder (figure 1.7) sold for $270,000 in 1991. Then within a

Figure 1.6 3M 32-track digital tape recorder, introduced in 1978.

Figure 1.7 Studer D820-48 DASH digital multitrack recorder, introduced in 1991 with a retail price of about $270,000. To make a backup copy of the tape required the use of two machines.

Figure 1.8 Sony PCM-D100 field recorder.

short time, software DAWs replaced tape recorders in most studios. It became possible to record, edit, and mix on portable laptop computers.

High-quality digital field recorders with built-in microphones became popular (figure 1.8).

To some extent, the functionality of portable field recorders can be achieved on mobile phones with accessories for an audio interface and high-quality stereo microphones. The downside of mobile devices is rapid obsolescence, which afflicts anything connected to them.

The Art of Recording

The art of recording requires more than proper equipment. Serious students of recording have much to learn through apprenticeship. A number of schools offer four-year Tonmeister degrees in recording engineering. Tonmeisters study music as well as applied physics. They learn about the acoustics of rooms (reflection, absorption, and diffraction of sound waves) and instruments, microphone types, microphone techniques, and audio media production methods.

2 *Basics of Sound Signals*

This chapter introduces basic concepts and terminology for describing sound signals, including frequency, amplitude, and phase.

Frequency and Amplitude

Sound reaches listeners' ears after being transmitted through air from a source. We hear sound because the air pressure is changing slightly in our ears, causing the eardrum to vibrate. If the pressure varies according to a repeating pattern, we say that the sound has a *periodic waveform.* If there is no discernible pattern it is called *noise.* In between these two extremes is a vast domain of quasiperiodic and quasinoisy sounds.

One repetition of a periodic waveform is called a *cycle,* and the *fundamental frequency* of the waveform is the number of cycles that occur per second. In the rest of this book we substitute *Hz* for *cycles per second* in accordance with standard acoustical terminology. (Hz is an abbreviation for *hertz*, the unit of measurement named after the German acoustician Heinrich Hertz.)

As the length of the cycle (called the *period*) decreases, the frequency in cycles per second increases, and vice versa. Table 2.1 shows the relationship between frequency and period.

Another descriptive term is *wavelength*, which is the measure of the physical distance between periods. Because sound travels at about 343 m/s at 20° Celsius, a wave at 1 Hz unfolds over about 343 m, whereas a wave at 20 kHz unfolds over about 0.017 m or about 1.7 cm.

Time-Domain Representation

A simple method of depicting sound waveforms is to draw them in the form of a graph of air pressure versus time (figure 2.1). This pressure graph is called a *time-domain* representation. When the curved line is near the bottom of the graph, the air pressure is lower; when the curve is near the top of the

Table 2.1 Relationship of frequency to period

Frequency	Period
1 Hz	1 second (s)
10 Hz	0.1 s or 100 milliseconds (ms)
100 Hz	0.01 s or 10 ms
1000 Hz	0.001 s or 1 ms
10000 Hz	0.0001 s or 100 microseconds (μs)

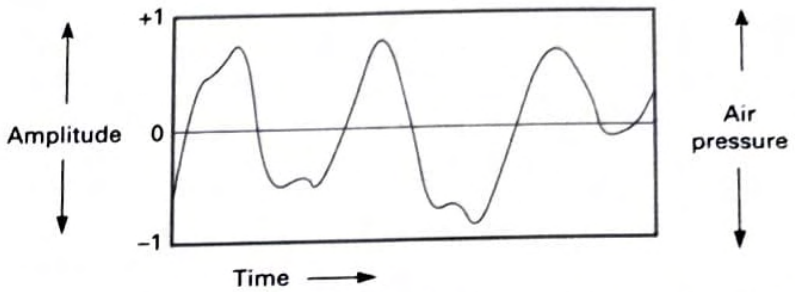

Figure 2.1 Time-domain representation of a signal. The vertical dimension shows the air pressure. When the curved line is near the top of the graph, the air pressure is greater. Below the solid horizontal line, the air pressure is reduced. Atmospheric pressure variations heard as sound can occur quickly; for musical sounds, this entire graph might last no more than one-thousandth of a second (1 ms).

graph, the air pressure has increased. The *amplitude* of the waveform is the amount of air pressure change; we can measure amplitude as the vertical distance from the zero pressure point to the highest (or lowest) points of a given waveform segment.

An acoustic instrument creates sound by emitting vibrations that change the air pressure around the instrument. A loudspeaker creates sound by moving back and forth according to voltage changes in an electronic signal. When the loudspeaker moves *in* from its position at rest, then the air pressure decreases. As the loudspeaker moves *out*, the air pressure near the loudspeaker is raised. To create an audible sound, these in/out vibrations must occur at a frequency in the range of about 20 to 20,000 Hz.

Frequency-Domain Representation

Besides the fundamental frequency, there can be many frequencies present in a waveform. A *frequency-domain* or *spectrum* representation shows the frequency content of a sound. The individual frequency components of the spectrum can be referred to as *harmonics* or *partials*. Harmonic frequencies are simple integer multiples of the fundamental frequency. Assuming a fundamental or *first harmonic* of 100 Hz, its second harmonic is 200 Hz, its third harmonic is 300 Hz, and so on. More generally, any frequency component can be called a partial, whether or not it is an integer multiple of a fundamental. Indeed, many sounds have no particular fundamental frequency.

The frequency content of a waveform can be displayed in many ways. A standard way is to plot each partial as a line along an x-axis. The height of each line indicates the strength (or amplitude) of each frequency component. The purest signal is a *sine* waveform, so named because it can be calculated

using trigonometric formulae for the sine of an angle. A pure sine wave represents just one frequency component or one line in a spectrum. Figure 2.2 depicts the time-domain and frequency-domain representations of several waveforms. Notice that the spectrum plots are labeled *harmonics* on their horizontal axis because the analysis algorithm assumes that its input is exactly one period of the fundamental of a periodic waveform. In the case of the noise signal in figure 2.2g, this assumption is not valid, so we relabel the partials as *frequency components.*

As we discuss in chapters 36–39, there are many ways to plot the spectrum of a sound.

Phase

The starting point of a periodic waveform on the y or amplitude axis is its *initial phase.* For example, a typical sine wave starts at the amplitude point 0 and completes its cycle at 0. If we displace the starting point by $\pi/2$ radians or 90° on the horizontal axis then the sinusoidal wave starts and ends at 1 on the amplitude axis. By convention this is called a *cosine wave.* In effect, a cosine is equivalent to a sine wave that is *phase shifted* by 90° (figure 2.3).

When two signals start at the same point, they are said to be *in phase* or *phase aligned.* This contrasts to a signal that is slightly delayed with respect to another signal, in which the two signals are *out of phase.* When a signal A is the exact opposite phase of another signal B (i.e., it is 180° out of phase, so that for every positive value in signal A there is a corresponding negative value for signal B), we say that B has *reversed polarity* with respect to A. We could also say that B is a *phase-inverted* copy of A. Figure 2.4 portrays the effect when two signals in inverse phase relationship sum: they cancel out each other.

Importance of Phase

It is sometimes said that phase is insignificant to the human ear, because two signals that are exactly the same except for their initial phase are difficult to distinguish. However, research indicates that 180° differences in absolute phase or *polarity* can be distinguished by some people under laboratory conditions (Greiner and Melton 1991). For more on phase perception, refer to Laitinen, Disch, and Pulkki (2013).

Even apart from special cases, phase is an important concept for several reasons. Every filter uses phase shifts to alter signals. A filter shifts the phase

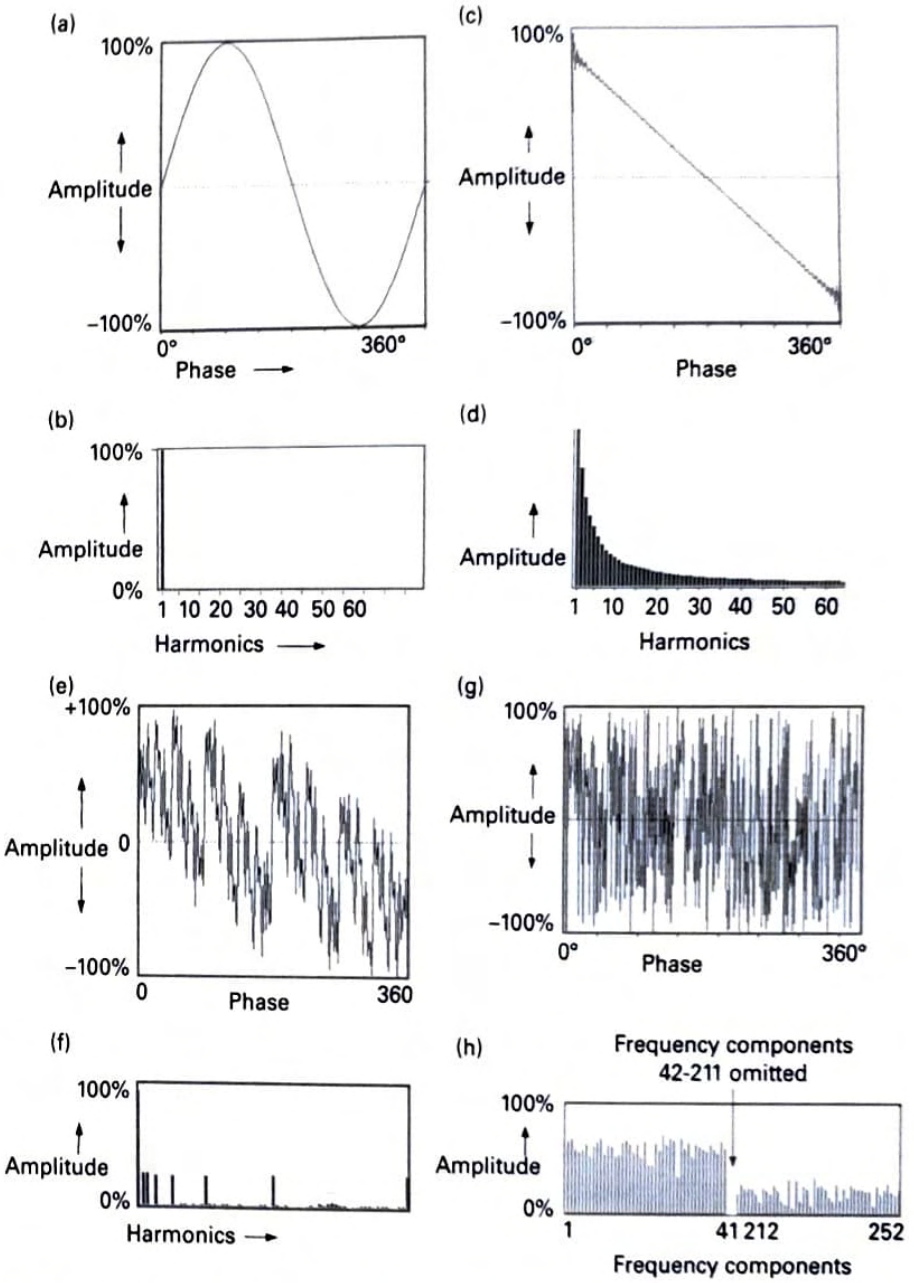

Figure 2.2 Time-domain and frequency-domain representations of four signals. (a) Time-domain view of one cycle of a sine wave. (b) Spectrum of the one frequency component in a sine wave. (c) Time-domain view of one cycle of a sawtooth waveform. (d) Spectrum showing the exponentially decreasing frequency content of a sawtooth wave. (e) Time-domain view of one cycle of a complex waveform. Although the waveform looks complex, when it is repeated over and over its sound is actually simple—like a thin reed organ sound. (f) The spectrum of waveform (e) shows that it is dominated by a few frequencies. (g) A random noise waveform. (h) If the waveform is constantly changing (each cycle is different from the last cycle), then we hear noise. The frequency content of noise is very complex. In this case the analysis extracted 252 frequencies. This snapshot does not reveal how their amplitudes are constantly changing over time.

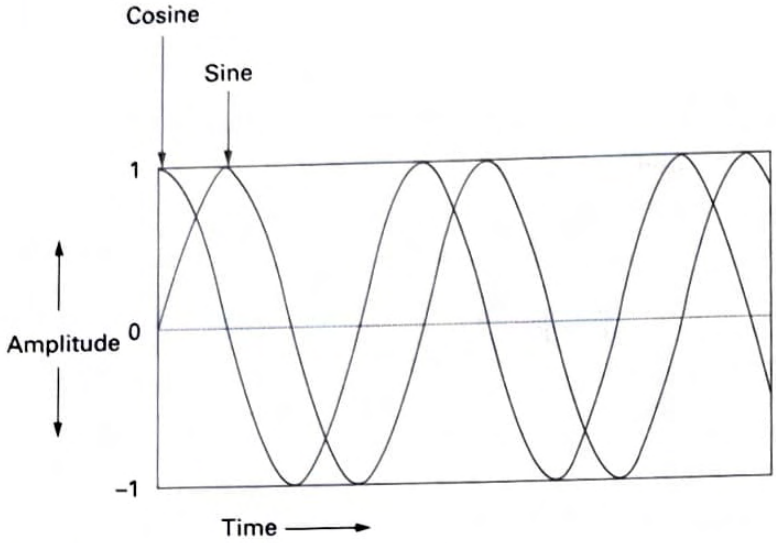

Figure 2.3 A sine waveform is equivalent to a cosine waveform that has been delayed and hence phase shifted slightly.

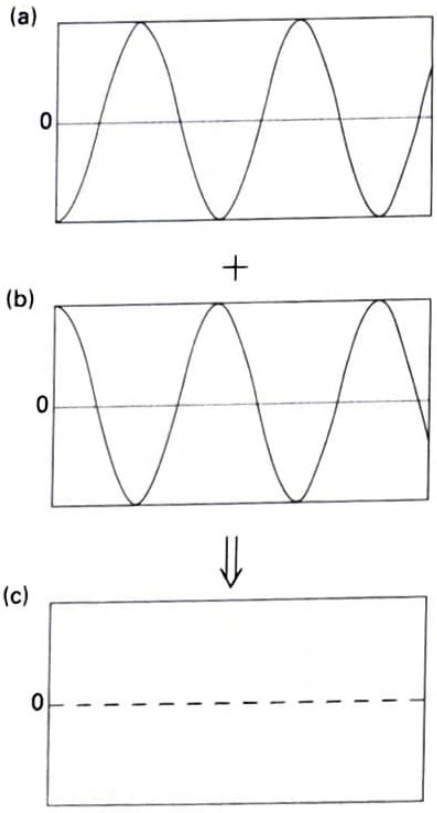

Figure 2.4 The effects of phase inversion. (b) is a phase-inverted copy of (a). If the two waveforms are added together, (c) they sum to zero.

of a signal by delaying its input for a short time and then combines the phase-shift version with the original signal. This creates *frequency-dependent phase cancelation* to attenuate certain frequencies and *phase reinforcement* to boost certain frequencies. By *frequency-dependent* we mean that not all frequency components are affected equally. When the phase shift is time varying, the affected frequency bands also vary, creating the sweeping sound effect called *phasing* or *flanging* (see chapter 30).

Phase is also important in systems that resynthesize sound on the basis of an analysis of an existing sound. In particular, these systems need to know the starting phase of each frequency component in order to assemble the different components in the right order (M.-H. Serra 1997; X. Serra 1997). Phase data is particularly critical in reproducing short, rapidly changing *transient* sounds, such as the onset of a percussive tone.

Finally, much attention has been invested in audio components that shift the phases of their input signals as little as possible, because frequency-dependent phase shifts distort musical signals audibly and interfere with loudspeaker *imaging*. (Imaging is the ability of a set of loudspeakers to create a stable *audio picture* in which each audio source is localized to a specific place within the picture.) Unwanted phase shifting is called *phase distortion*. To make a visual analogy, a phase-distorted signal is *out of focus*.

Sound Magnitude

We all have an intuitive notion of *sound level* or *sound magnitude*. Even a small child understands the function of a volume knob. The *decibel* is a unit of measurement for relationships of magnitude, including voltage levels, intensity, or power, particularly in audio systems. In acoustic measurements, the decibel scale indicates the ratio of one level to a *reference level*, according to the relation

$$\textit{number of decibels} = 10 \times \log_{10} (\textit{level/reference level})$$

where the *reference level* is usually the threshold of hearing (10^{-12} watts per square meter). The logarithmic basis of decibels means that if two notes sound together and each note is 60 dB, the increase in level is just 3 dB. A millionfold increase in intensity results in only a 60 dB boost.

Figure 2.5 shows the decibel scale and some estimated acoustic power levels relative to 0 dB.

Chapter 4 contains a more thorough discussion of sound magnitude and decibels.

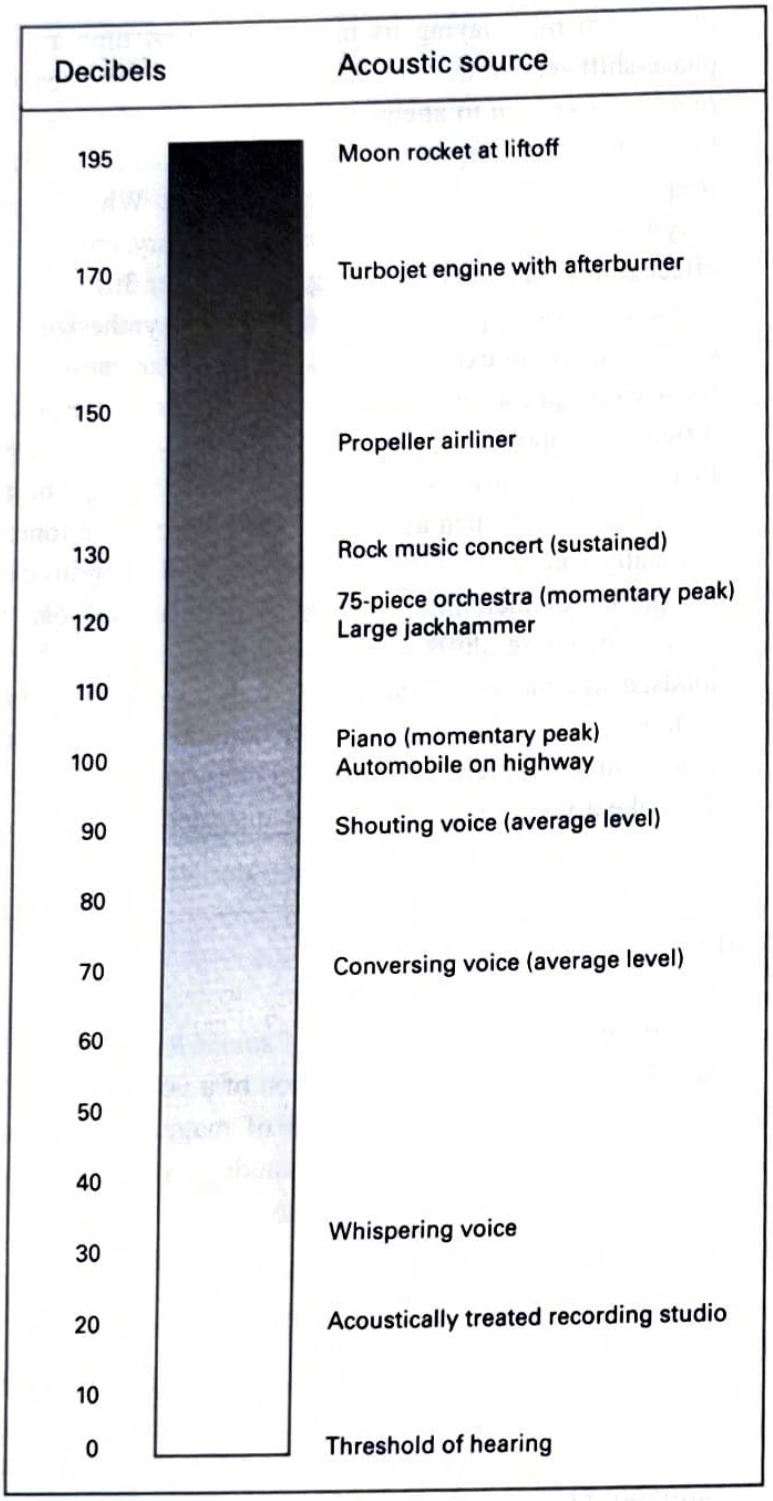

Figure 2.5 Typical acoustic power levels for various acoustic sources. All values are relative to 0 dB.

Dynamic Range

Dynamic range is the ratio between the loudest and the softest sound that can be handled by a system without distortion. Two important facts describe the dynamic range requirements of a digital audio system:

- The range of human hearing extends from approximately 0 dB, roughly the level at which the softest sound can be heard, to around 125 dB, which is roughly the threshold of pain for sustained sounds.
- A difference of somewhat less than one dB between the amplitude levels of two sounds corresponds to the smallest difference in amplitude that can be heard.

In recording music, we want to reproduce the full expressive power of the music. Thus it is important to capture the widest possible dynamic range. In a live orchestra concert, for example, the dynamic range can vary from silence to a tutti (full orchestra) section exceeding 110 dB.

Every recording device (such as a microphone, mic preamplifier, mixer, or recorder) can handle only a certain dynamic range before it distorts. For example, the dynamic range of analog tape equipment is dictated by the physics of the analog recording process. That range is around 80 dB for a 1 kHz tone using professional reel-to-reel tape recorders without noise-reduction devices. By contrast, a high-quality digital audio recorder can have a dynamic range of around 120 dB.

Chapter 4 goes into more detail about dynamic range in digital audio systems.

3 Theory of Sampling

Curtis Roads with John M. Strawn

This chapter begins by explaining differences between analog and digital audio systems. We then step through the basics of the digital audio recording and playback chain. For further technical study, consult Pohlmann (2010).

Analog Representations of Sound

Just as air pressure varies according to sound waves, so can the electrical property called *voltage* in a wire connecting an amplifier with a loudspeaker. We do not need to define voltage here. For the purposes of this chapter, we simply assume that it is possible to modify an electrical property in a fashion that closely matches changes in air pressure. We can say that air pressure and voltage are analogous to each other. That is, a graph of the air pressure variations picked up by a microphone looks similar to a graph of the variations in the loudspeaker position when that sound is played back. The term *analog* means that these properties can vary in a similar manner.

Figure 3.1 shows an analog audio playback chain. The curve of an audio waveform can be inscribed along the groove of a traditional phonograph record. The walls of the grooves on a phonograph record contain a *continuous-time* representation of the sound stored in the record. As the needle glides through the groove, it moves back and forth in lateral motion. This lateral motion is then changed into voltage, which is amplified and eventually reaches the loudspeaker.

Analog recording and reproduction of sound has been taken to a high level, but it faces fundamental physical limits. Specifically, when one copies an analog recording onto another analog recorder, the copy is never as good as the original. The reason is that the analog recording process always adds noise. On a *first-generation* or original recording made with a high-quality tape recorder, this noise is not objectionable. But if we copy the first-generation tape onto another tape and then copy the copy, the noise increases noticeably. In contrast, digital technology can create any number of generations of perfect (noise-free) clones of an original digital recording, as we show further on.

Digital Representations for Sound

This section introduces the most basic concepts associated with digital signals, including the conversion of audio signals into binary numbers, and comparison of audio data with MIDI data.

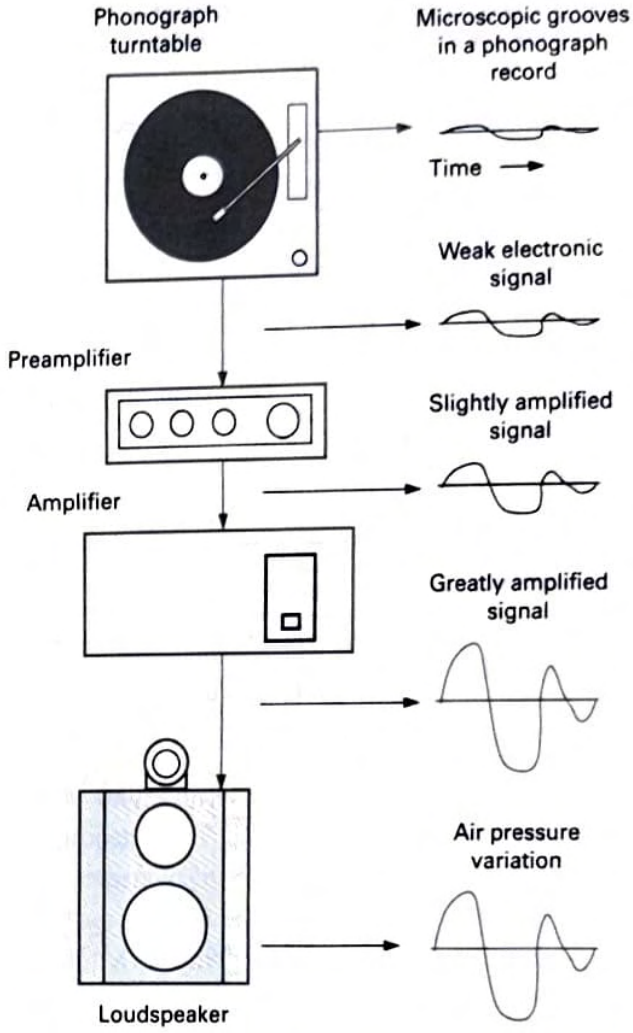

Figure 3.1 The analog audio playback chain, starting from an analog waveform transduced from the grooves of a phonograph record to a voltage sent to a preamplifier, amplifier, and loudspeaker and projected into the air.

Analog-to-Digital Conversion

Let us look at the process of digitally recording sound and then playing it back. Rather than the continuous-time signals of the analog world, a digital recorder handles *discrete-time* signals. Figure 3.2 shows a diagram of the digital audio recording and playback process. In this diagram, a microphone transduces air pressure variations into electrical voltage variations, which are then passed through an *antialiasing filter* and then to an *analog-to-digital converter* (ADC). (We discuss the function of the antialiasing filter in a subsequent section.) The ADC samples and converts the voltage variations into a string of *binary numbers* at each uniform period of the sample

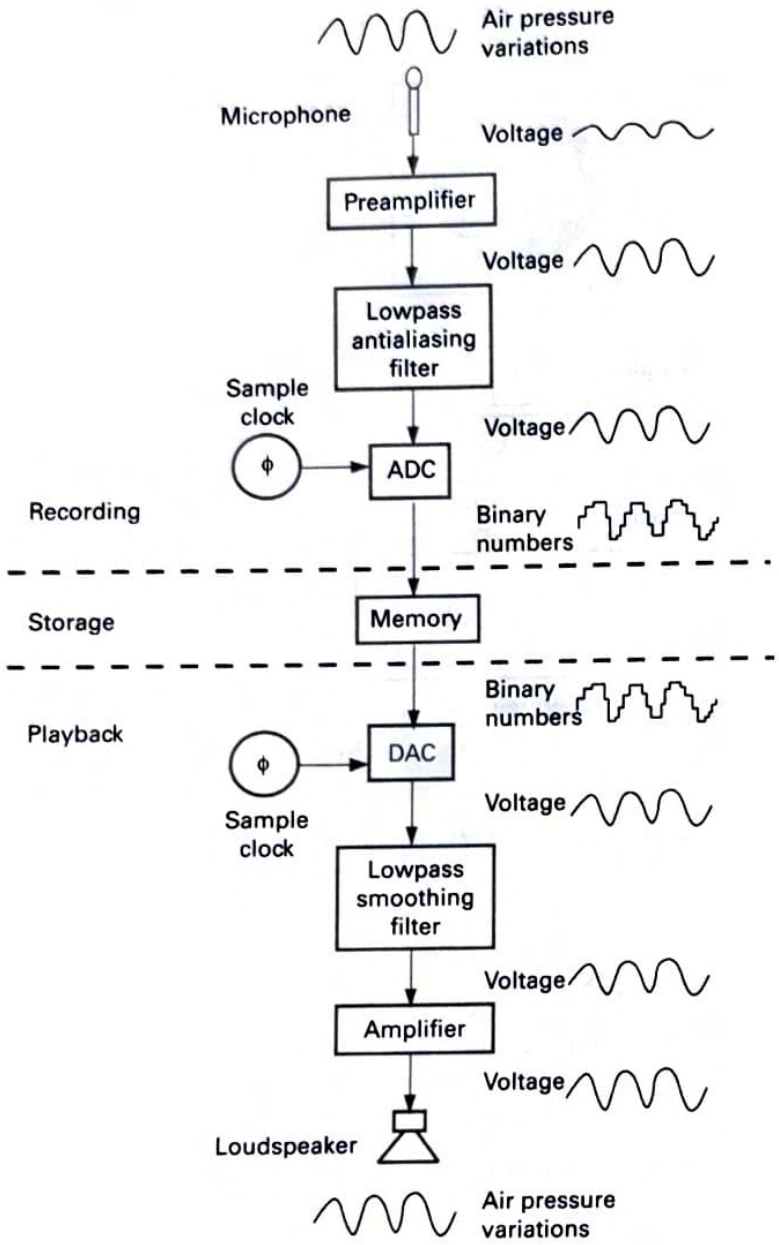

Figure 3.2 Overview of digital recording and playback.

clock. The binary numbers are then stored in a digital recording medium—a type of memory.

Binary Numbers

In contrast to decimal (or *base ten*) numbers, which use the ten digits 0–9, binary (or *base two*) numbers use only two digits: 0 and 1. The term *bit* is an abbreviation of *binary digit*. Table 3.1 lists binary numbers and their

Table 3.1 Binary numbers and their decimal equivalents

Binary	Decimal
0	0
1	1
10	2
11	3
100	4
1000	8
10000	16
100000	32
1111111111111111	65,535

decimal equivalents. In many digital systems the leftmost bit is interpreted as a sign indicator, with a 1 indicating a positive integer and a 0 indicating a negative integer. (Real decimal numbers such as 8.476 can also be represented in binary as *floating-point numbers*, but we will not explain this scheme here.)

The way a bit is physically encoded in a recording medium depends on the properties of that medium. On a magnetic disc, for example, a 1 might be represented by a positive magnetic charge, whereas a 0 is indicated by the absence of such a charge. This is different from an analog magnetic tape recording, in which the signal is represented as a continuously varying magnetic charge. On an optical medium such as a compact disc, binary data might be encoded as variations in the reflectance at a particular location. *Solid-state* or *flash memory* encodes a bit as an electrical charge in a memory cell made out of transistors.

Digital-to-Analog Conversion

Figure 3.3 depicts the result of converting an audio signal (a) into a digital signal (b). When the listener wants to hear the sound again, the binary numbers are read one by one from the digital storage and passed through a *digital-to-analog converter* (DAC). This device, driven by a sample clock, changes the stream of numbers into a series of voltage levels. From there, the process is the same as shown in figure 3.2; that is, the series of voltage levels are lowpass filtered into a continuous-time waveform (figure 3.3c), amplified, and routed to a loudspeaker, whose vibration causes the air pressure to change. Voilà—the signal sounds again.

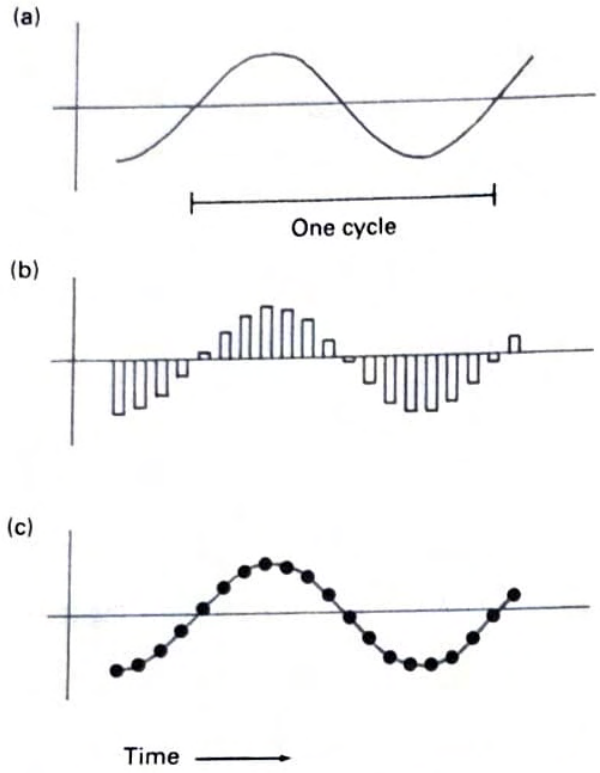

Figure 3.3 Analog and digital representations of a signal. (a) Analog sine waveform. The horizontal bar below the wave indicates one period or cycle. (b) Sampled version of the sine waveform in (a), as it might appear at the output of an ADC. Each vertical bar represents one sample. Each sample is stored in memory as a number that represents the height of the vertical bar. (c) Reconstruction of the sampled version of the waveform in (b). In effect, the samples are connected by the lowpass smoothing filter to form the waveform that eventually reaches the listener's ear.

In summary, we can change a sound in the air into a string of binary numbers that can be stored digitally. The central component in this conversion process is the ADC. When we want to hear the sound again, a DAC can change those numbers back into sound.

Digital Audio Samples Are Not MIDI Data

This section may clear up confusion. The string of numbers generated by the ADC is not related to Musical Instrument Digital Interface (MIDI) data. (For more information on MIDI, see chapter 52.) Digital audio recording samples the sound waveform whereas MIDI recording captures the data played on a controller, such as a keyboard. MIDI note information includes only the start and ending time, pitch (a number), and amplitude at the beginning of the note (a number). If a MIDI note is transmitted back to the

synthesizer on which it was originally played, this causes the synthesizer to play the sound as it did originally, similarly to a piano roll recording. If the musician plays four quarter notes at a tempo of sixty beats per minute on a MIDI synthesizer, just sixteen pieces of information capture this four-second sound (four starts, ends, pitches, and amplitudes).

By contrast, if we record the same sound with a microphone connected to a digital audio recorder set to a sampling frequency of 44.1 kHz, 352,800 pieces of information (in the form of audio samples) are recorded for the same sound (44,100 × 2 channels × 4 seconds). The storage requirements of digital audio recording are large. Using 16-bit samples, it takes over 700,000 bytes to store a four-second sound. This is 44,100 times more data than is stored by MIDI.

The advantage of a digital audio recording is that it can capture any sound that can be recorded by a microphone, including the human voice. By contrast, MIDI sequence recording is limited to recording control signals that indicate the start, end, pitch, and amplitude of a series of note events.

Sampling

The digital signal shown in figure 3.3b is significantly different from the original analog signal shown in figure 3.3a. First, the digital signal is defined only at certain points in time. This happens because the signal has been *sampled* at certain times. Each vertical bar in figure 3.3b represents one *sample* of the original signal. The samples are stored as binary numbers; the higher the bar in figure 3.3b, the larger the number.

The number of bits used to represent each sample is called the *quantization* of the system. Quantization determines both the noise level and the amplitude range that can be handled by the system. For example, a compact disc has a quantization of 16 bits. That is, every sample is represented by a 16-bit number. More bits of quantization mean better amplitude resolution. This translates into lower noise and more dynamic range. Fewer bits result in the opposite. We return to this subject in chapter 4 when we discuss quantization at greater length.

The rate at which samples are taken—the *sampling frequency*—is expressed in terms of samples per second. This is an important specification of digital audio systems. It is often called the *sampling rate* and is expressed in terms of hertz (Hz). Simplifying the measurement 1,000 Hz to 1 kHz, we say, "The sampling rate of a compact disc recording is 44.1 kHz," where the *k* is derived from the metric term *kilo*, which means thousand.

Reconstruction of the Analog Signal

Sampling frequencies around 50 kHz are common in digital audio systems, although both lower and higher frequencies can also be found. In any case, 50,000 numbers per second is a rapid stream of numbers; it means there are 6,000,000 samples for one minute of stereo sound.

The digital signal in figure 3.3b does not show the value between the bars. The duration of a bar is extremely narrow, perhaps lasting only 0.00002 s (two hundred-thousandths of a second). This means that if the original signal changes between the bars, the change is not reflected in the height of a bar until the next sample is taken. In technical terms, we say that the signal in figure 3.3b is defined at *discrete* times, each such time represented by one sample (vertical bar).

Part of the magic of digital audio is that if the signal is *bandlimited,* the DAC and associated hardware can exactly reconstruct the original signal from these samples. We call a signal bandlimited if it has frequencies only within a finite range. This means that, given certain conditions, the missing part of the signal between the samples can be restored. This happens when the numbers are passed through the DAC, which includes the lowpass smoothing filter. The lowpass smoothing filter is designed precisely to re-create the missing part of the signal between the discrete samples (see the dotted line in figure 3.3c). Thus, the signal sent to the loudspeaker looks and sounds like the original signal.

Aliasing

The process of sampling is not quite as straightforward as it might seem. Just as an audio amplifier can introduce distortion, sampling can play tricks with sound. Figure 3.4 gives an example. Using the input waveform shown in figure 3.4a, suppose that a sample of this waveform is taken at each point in time shown by the vertical bars in figure 3.4b (each vertical bar creates one sample). As before, the resulting samples of figure 3.4c are stored as numbers in digital memory. However, when we attempt to reconstruct the original waveform, as shown in figure 3.4d, the result is something completely different.

In order to understand better the problems that can occur with sampling, we look at what happens when we change the *wavelength* (the length of one cycle) of the original signal without changing the length of time between samples. Figure 3.5a shows a signal with a cycle eight samples long, figure 3.5d shows a cycle two samples long, and figure 3.5g shows a waveform with eleven cycles per ten samples.

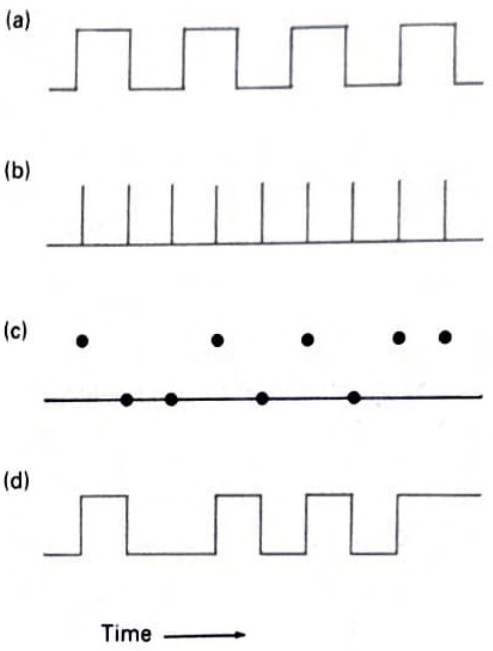

Figure 3.4 Problems in sampling. (a) Waveform to be recorded. (b) The sampling pulses; whenever a sampling pulse occurs, one sample is taken. (c) The waveform as sampled and stored in memory. (d) When the waveform from (c) is sent to the DAC, the output might appear as shown here (after Mathews 1969).

Again, as each of the sets of samples is passed through the DAC and associated hardware, a signal is reconstructed (figures 3.5c, f, and i) and sent to the loudspeaker. The signal shown by the dotted line in figure 3.5c is reconstructed more or less accurately. The results of the sampling in figure 3.5f are potentially less satisfactory; one possible reconstruction is shown there. But in figure 3.5i, the resynthesized waveform is completely different from the original in one important respect. Namely, the wavelength (length of the cycle) of the resynthesized waveform is different from that of the original. In the real world, this means that the reconstructed signal sounds at a pitch different from that of the original signal. This kind of distortion is called *aliasing*.

The frequencies at which this aliasing occurs can be predicted. Suppose, just to keep the numbers simple, that the sampling rate is 1,000 Hz. Then the signal in figure 3.5a has a frequency of 125 Hz (because there are eight samples per cycle, and 1,000/8 = 125). In figure 3.5d, the signal has a frequency of 500 Hz (because 1,000/2 = 500). The frequency of the input signal in figure 3.5g is 1,100 Hz.

Notice how the frequency of the output signal is different. In figure 3.5i you can count ten samples per cycle of the output waveform. In actuality, the output waveform occurs at a frequency of 1,000/10 = 100 Hz. Thus the

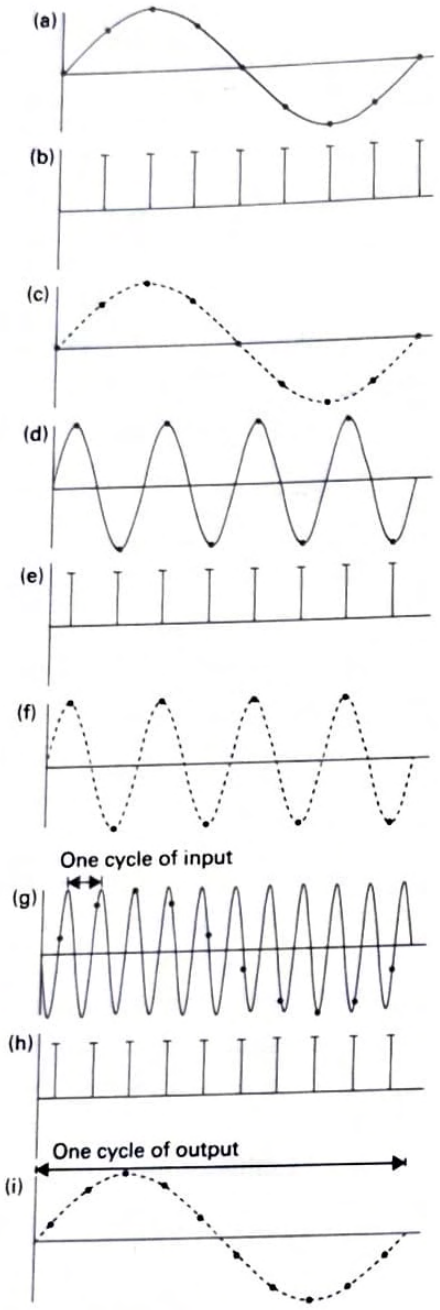

Figure 3.5 Aliasing effects. At the bottom of each set of three graphs, the thick black dots represent samples, and the dotted line shows the signal as reconstructed by the DAC. Every cycle of the sine waveform (a) is sampled eight times in (b). Using the same sampling frequency, each cycle of (d) is sampled only twice in (e). If the sampling pulses in (e) were moved to the right, the output waveform in (f) might be phase-shifted, although the frequency of the output would still be the same. In (h), there are ten samples for the eleven cycles in (g). When the DAC tries to reconstruct a signal, as shown by the dashed lines in (i), a sine waveform results, but the frequency has been completely changed due to the foldover effect. Notice the horizontal double arrow above (g), indicating one cycle of the input waveform, and the arrow above (i), indicating one cycle of the output waveform.

frequency of the original signal in figure 3.5g has been changed by the *sample rate conversion* process. This represents an unacceptable change to a musical signal that must be avoided if possible.

The Sampling Theorem

We can generalize from figure 3.5 to say that as long as there are at least two samples per period of the original waveform, we can assume that the resynthesized waveform will have the same frequency. But when there are fewer than two samples per period, the frequency of the original signal is lost. If the original frequency is higher than half the sampling frequency, then

new frequency = sampling frequency − original frequency

This formula is not mathematically complete, but it is sufficient for our discussion here. It means the following. Suppose that we have chosen a fixed sampling frequency. We start with a signal at a low frequency, sample it, and resynthesize the signal after sampling. As we raise the pitch of the input signal (but still keep the sampling frequency constant), the pitch of the resynthesized signal is the same as the pitch of the input signal until we reach a pitch that corresponds to one-half the sampling frequency. As we raise the pitch of the input signal even higher, the pitch of the output signal starts to descend to the lowest frequencies!

The process is depicted in figure 3.6, which shows why aliasing was sometimes called *foldover.*

To give a concrete example, suppose that we introduce an analog frequency at 30 kHz into an ADC operating at a 48 kHz sampling rate. When reconstructed by a DAC, it will produce a tone at 18 kHz, because 48 − 30 = 18.

The *sampling theorem* (or *Nyquist theorem*) describes the relationship between the sampling rate and the bandwidth of the signal being transmitted. It was expressed by Harold Nyquist (1928) as follows:

For any given deformation of the received signal, the transmitted frequency range must be increased in direct proportion to the signaling speed. . . . The conclusion is that the frequency band is directly proportional to the speed.

The essential point of the sampling theorem can be stated precisely as follows:

In order to be able to reconstruct a continuous signal from its samples, the frequency at which we sample must be at least twice the highest frequency in the signal.

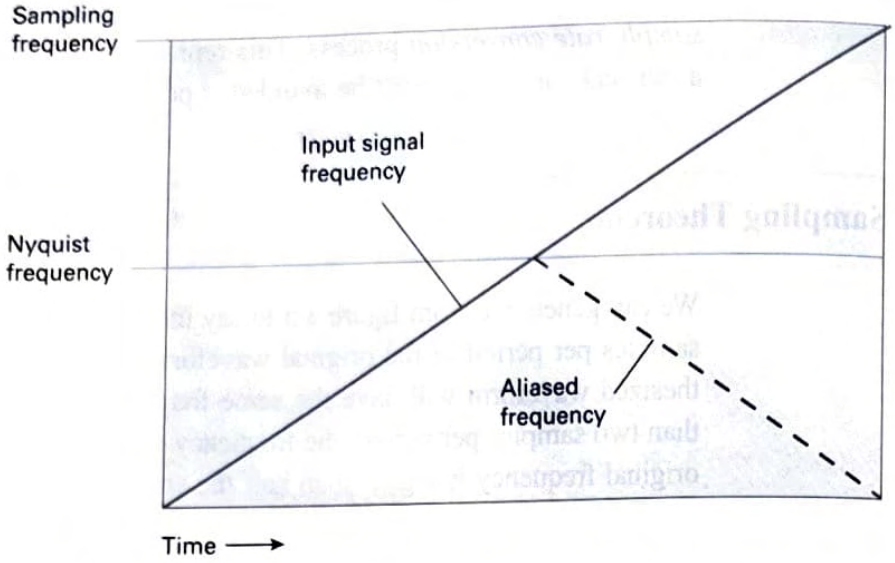

Figure 3.6 When the input frequency exceeds the Nyquist frequency, the recorded signal *folds over* and proceeds downward.

The highest frequency that can be reproduced in a digital audio system (i.e., half the sampling rate) is called the *Nyquist frequency.* In digital musical systems, the Nyquist frequency is usually above the upper range of human hearing, that is, above 20 kHz. Then the sampling frequency can be specified as being at least twice as much, or above 40 kHz. For example, in a recording system that samples at 44.1 kHz (the sampling frequency of compact discs), the Nyquist frequency is 22.05 kHz.

Antialiasing and Anti-imaging Filters

In order to make sure that a digital sound system works properly, two important filters are included. Recall figure 3.2. One filter is placed before the ADC, to make sure that nothing in the input signal occurs at a frequency higher than half the sampling frequency. As long as this filter does the proper work, aliasing should not occur during the recording process. Logically enough, such a filter is called an *antialiasing filter.*

The other filter is placed after the DAC. Its main function is to change the samples stored digitally into a smooth, continuous representation of the signal. In effect, this *lowpass anti-imaging* or *smoothing filter* creates the dotted line in figure 3.3c by connecting the solid black dots in the figure.

Audio Interfaces

The ADCs and DACs built into computers and mobile devices are inexpensive to manufacture. They perform adequately for daily use. However, the standard audio input and output of a computer, tablet, or phone is inadequate for high-quality audio recording and playback.

An *audio interface* provides a high-quality audio solution (figure 3.7). It connects to the computer or mobile device using a protocol that the device supports (such as USB, Thunderbolt, or Ethernet). It includes high-quality DACs and ADCs and microphone/line preamplifiers. Many have traditional MIDI input/output connectors.

The cost of audio interfaces ranges from inexpensive units (<$100) designed for home recording to professional units (>$10,000) that support multiple channels at high sampling rates.

Ideal Sampling Frequency

The Audio Engineering Society has recommended a set of standard sampling rates for audio, including 32, 44.1, 48, and 96 kHz (Audio Engineering Society 2008). It also recognizes that multiples of these basic rates are in use, such as 88.2. 176.4, 192, 352.8, 384, and even 768 kHz.

What sampling frequency is ideal for high-quality music recording and reproduction? Experts disagree.

We all want cameras with high resolution. Should we not also want high-resolution audio recordings? Higher sampling rates increase the bandwidth of recording, which means better audio quality. However, they also produce

Figure 3.7 Front and back panel of the RME Fireface UFX+ audio interface. The front panel shows four mic/line preamplifiers and MIDI jacks. Twelve analog inputs and outputs are supported. The rear panel bristles with connectors, including optical MADI multichannel jacks. The interface processes a total of ninety-six inputs and outputs and connects to a computer via USB 3 or Thunderbolt.

much larger file sizes. The more samples in a file, the greater the processing load. Transmitting a high sample rate file over a network takes more time. A file recorded at 192 kHz/24 bits is about nine times larger than one recorded at 44.1 kHz/16 bits.

One way to view resolution is to see the effect of the sampling process on an analog impulse. Figure 3.8 plots these effects. See how recording at 48 kHz blurs in time the original analog transient, as though it is out of focus.

One justification that has been given for higher sampling rates is that some people hear information (referred to as *air*) in the region around the 20 kHz limit of human hearing (Neve 1992). Many analog systems can reproduce ultrahigh frequencies. Microphones like the Sanken CO-100K can record sounds up to 100 kHz. Some microphone preamplifiers have a bandwidth that extends far beyond 200 kHz.

Scientific experiments confirm the effects of sounds above 22 kHz from both physiological and subjective viewpoints (Oohashi et al. 1991; Oohashi et al. 1993). Melchior (2019), however, points out how the improved quality of high-resolution recordings can be heard by people without extraordinary high-frequency hearing. She observes that the *temporal blur, pre-echo,* and *ringing* inherent in brick-wall filters associated with sampling rates below 50 kHz are contributing factors, and she lists several high-resolution schemes that alleviate these symptoms.

Another argument for high resolution is the more focused spatial imaging that accrues due to the presence of more high-frequency spatial information.

In sound synthesis applications, the lack of *frequency headroom* in standard sampling rates of 44.1 and 48 kHz is a source of serious prob-

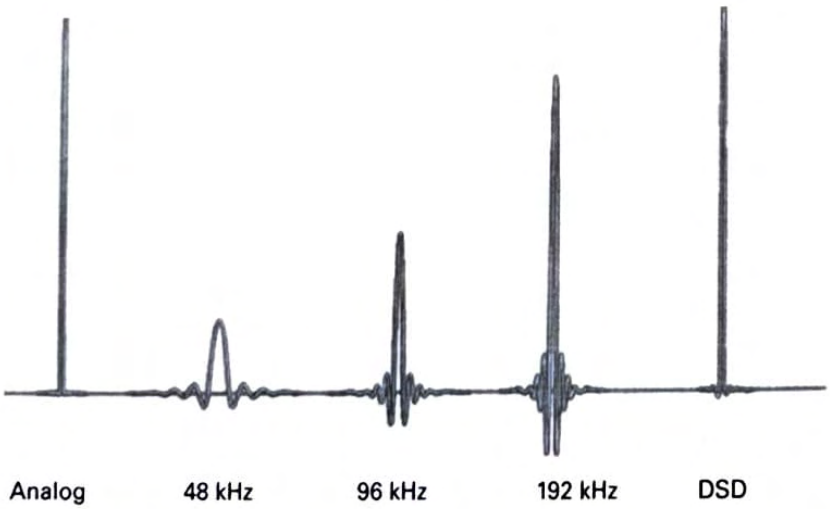

Figure 3.8 Impulse responses (*left to right*) of an analog impulse (*left*) as recorded by 48 kHz, 96 kHz, 192 kHz, and direct digital stream (DSD) recording systems. DSD is discussed in chapter 4.

lems. It requires that synthesis algorithms generate nothing other than sine waves above 11.025 kHz (44.1 kHz sampling rate) or 12 kHz (48 kHz sampling rate) in order to avoid aliasing. The reason is that any nonsinusoidal periodic waveform can have partials that exceed the Nyquist rate.

In sampling and *pitch-shifting* applications (see chapter 31), the lack of frequency headroom requires that sampled sounds be lowpass filtered before they are pitch-shifted upward to avoid aliasing. When tones recorded at 44.1 or 48 kHz are pitch-shifted downward, they become muffled and lose their high-frequency content because everything above the Nyquist frequency has already been eliminated by the antialiasing filter. In a recording made at a high sampling rate, pitch-shifting downward does not necessarily mean a loss of high frequency content, because ultrasonic energy is transposed into the high-frequency audio range.

High sampling rate recordings are preferable from an audio standpoint, but one needs a high-quality playback system to make the effort worthwhile. Many people listen to music on low-quality ear buds connected to mobile devices, where the luxury of high-resolution recordings is wasted.

Jitter

Jitter is time-based error in sampling. If the clock driving the ADC or DAC is not stable, then the conversions will not happen at the correct times. A typical audible effect of jitter is a high-frequency clicking noise added to the signal.

In a home studio, jitter is not likely to be an issue. The possibility of jitter increases when multiple physical devices are interconnected. Jitter can be a product of multiple factors: clock instability, poor cable/connector quality, impedance mismatches, or software issues. Thus jitter is a practical issue in complex digital audio systems. The general solution to jitter problems is to rely on a *master wordclock generator* device that is interconnected to all other devices via high-quality cables and connectors.

The sampling rate determines how often a digital system measures a continuous signal in time. The next chapter presents the topic of quantization, that is, how precisely a digital system measures a signal in amplitude. Taken together, these two processes, sampling and quantization, constitute the core of digital audio theory.

4 *Sample Quantization, Conversion, and Audio Formats*

This chapter examines the notion of sound magnitude as it pertains to digital audio recording. We then look at the technical issues surrounding dynamic range, quantization, oversampling, and digital audio media and formats.

More on Sound Magnitude

As pointed out in chapter 2, everyone has an intuitive notion of sound level or magnitude. Dozens of terms have been devised by scientists to describe the magnitude of a sound. Among many are the following:

- Peak-to-peak amplitude
- RMS amplitude
- Gain
- Sound energy
- Sound power
- Sound intensity
- Sound pressure level
- Loudness

From a scientific point of view, these terms are all different. From a common sense point of view, the terms are all correlated and proportional to one another: a significant boost in one corresponds to a boost in all. Our ears are sharply attuned to sound magnitude, so the concept is physical and directly perceivable.

From a musical point of view, the most useful scientific terms are peak-to-peak and RMS amplitude (as seen in a sound editor), gain (a standard term for boosting or attenuating a sound), sound pressure level (what a sound-level meter measures in the air), and loudness (perceived magnitude). Sound energy, power, and intensity are terms used by physicists to describe measures of sound magnitude relative to the amount of work done, that is, how much energy it takes to vibrate a medium.

Table 4.1 summarizes the formal definitions of these terms. Figure 4.1 illustrates three of the most important measures: peak, peak-to-peak, and RMS amplitude. The rest of this section explains the useful concept of decibels.

Decibels

The ear is an extremely sensitive organ. Suppose that we sit three meters in front of a loudspeaker that is generating a sine tone at 1,000 Hz that we

Table 4.1 Units for measuring sound magnitude

Peak-to-peak amplitude	A measure of the peak-to-peak difference in waveform values expressed as a percentage or as decibels (dB). Useful for describing the magnitude of periodic waveforms in particular.
RMS amplitude	For complex signals such as noise, root mean squared (RMS) amplitude describes the average power of the waveform. RMS amplitude is the square root of the mean over time of the square of the vertical distance of the waveform from the rest position.
Gain	A measure of the ratio of the input and the output amplitude (or power) of a process, usually measured in decibels. A gain of greater than 1 dB is a boost, and a gain of less than 1 dB corresponds to attenuation.
Sound energy	A measure of work, sound energy is the ability to vibrate a medium, expressed in joules. A joule is a unit of energy corresponding to the work done by a force of 1 newton traveling through a distance of 1 meter. A newton is equal to the amount of force required to give a mass of 1 kilogram an acceleration of 1 meter per second squared
Sound power	The rate at which work is done or energy is used. The standard unit of power is the watt, corresponding to 1 joule per second. 1 watt is the rate at which work is done when an object is moving at 1 meter per second against a force of 1 newton.
Sound intensity	Sound power per unit area, measured in watts per square meter.
Sound pressure level (SPL)	Air pressure at a particular point, given in decibels as a ratio of sound pressure to a reference sound pressure of 20 micropascals. A pascal is a unit of pressure equivalent to the force of 1 newton per square meter.
Loudness	A psychoacoustic measure based on queries of human subjects, measured in phons. 1 phon equals 1 dB SPL at 1 kHz.

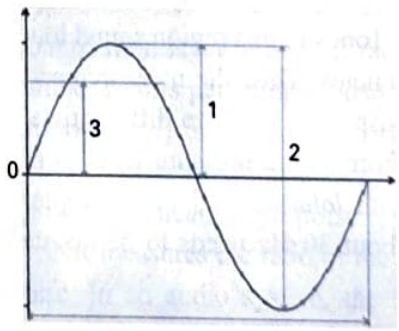

Figure 4.1 Measures of amplitude. (1) Peak amplitude. (2) Peak-to-peak amplitude. (3) RMS amplitude.

perceive as being very loud. Amazingly, one can reduce the power by a factor of one million and the tone is still audible. In an anechoic chamber where all external sounds are eliminated, the reduction extends to a factor of more than one billion (Backus 1977).

Sound transports energy generated by the vibration of a source. The range of sound energy encompasses everything from the subsonic flutterings of a butterfly to massive explosions. A whisper produces only a few billionths of a watt. By contrast, a large rocket launch generates about 10 million watts of power.

The decibel (dB) unit compresses these huge exponential variations into a smaller range by means of logarithms. The dB unit can be applied to myriad physical phenomena; however, the definition changes according to the phenomenon being measured. A standard unit in audio is dB SPL. This compares a given sound pressure level (SPL) with a standard reference level. The logarithm (base 10) of this ratio is the level in decibels, hence

$$\text{SPL in decibels} = 20 \log_{10} (W/W_0)$$

where W is the actual SPL of the signal being measured, and W_0 is a standard reference level of 20 micropascals of air pressure. This corresponds to the quietest sound that a human being can hear.

Describing sound levels in terms of dB enables a wide range. Table 4.2 shows how the decibel unit compresses large changes in percentage amplitude into relatively small changes in the number of dB.

As we move away from a sound source, its SPL diminishes according to the distance. Specifically, each doubling of distance decreases SPL by about 6 dB, which represents a 50% decrease in its amplitude. This is the famous *inverse square law*: intensity diminishes as the square of the distance.

So far we have been talking in terms of amplitude and SPL. Another pair of terms—*volume* and *loudness*—are intuitive. Technically, loudness refers to perceived subjective intensity measured through psychoacoustic tests on human beings and not to sound pressure level measured by laboratory instruments. For example, the ear is especially sensitive to frequencies between 1,000 Hz and 4,000 Hz. Tones in this region sound louder than tones of equal intensity in other frequencies. Thus the measurement of loudness falls into the realm of psychoacoustics. In order to differentiate loudness level (a perceptual characteristic) from sound pressure level (a physical characteristic), the unit *phon* (rhymes with *John*) is used. For example, to sound equally loud (60 phons), a tone at about 30 Hz needs to be boosted 40 dB more than a 1,000 Hz tone.

Table 4.2 Amplitude as a percentage versus as decibels

100%	0 dB
70%	−3 dB
50%	−6 dB
25%	−12 dB
12.5%	−18 dB
6.25%	−24 dB
3.125%	−30 dB
1.562%	−36 dB
0.781%	−42 dB
0.39%	−48 dB
0.195%	−54 dB
0.097%	−60 dB
0.048%	−66 dB
0.024%	−72 dB
0.012%	−78 dB
0.006%	−84 dB
0.003%	−90 dB

Quantization

Sampling at discrete time intervals constitutes one of the major differences between digital and analog signals. Another difference is *quantization*, which is sampling at discrete amplitude intervals. Digital numbers do not have infinite precision. They can be represented only within a certain range and with a certain accuracy, which varies with the hardware used.

Signal-to-Noise Ratio and Dynamic Range

Quantization is an important factor in digital audio quality. Specifically, the number of bits per sample (also called the *sample width*, *bit depth*, or *quantization level*) is important in calculating the *signal-to-noise ratio* (SNR). SNR is a measure of the noisiness of a system: a high SNR means low noise; a low SNR means high noise.

SNR measures the ratio of the strength of an audio signal to the strength of noise. In an audio system, the SNR is specified as the difference in level between the standard operating level (usually 0 on a VU meter) and the average

level of the noise floor, expressed in decibels. In general, each additional bit in the analog-to-digital converter will contribute about 6 dB to the SNR.

Another measurement is the *dynamic range* (DR) of a digital sound system. In simple terms, this is the difference in dB between the loudest and softest sounds that the system can produce. Clearly, SNR and DR are correlated; when the DR is high, so is the SNR. In theory, this is straightforward. It becomes more complicated when stipulating exactly how and in what units of measurement SNR and DR are calculated. As mentioned, the SNR is the difference between the standard operating level of 0 VU and the noise floor. The DR is the difference between the noise floor and the point of distortion. Thus the DR depends on a method of measuring distortion, which people define in different ways.

For our purposes, a simple formula for the dynamic range of a digital audio system is

$$\text{Dynamic range in decibels} = \text{Number of bits} \times 6.11$$

The number 6.11 is a close approximation to the theoretical maximum (van de Plaasche 1983; van de Plaasche and Dijkmans 1983; Hauser 1991); in practice, 6.0 is a more realistic figure. A derivation of this formula is given in Mathews (1969) and Blesser (1978).

Thus, if we record sound with an 8-bit system, then the upper limit on the DR is approximately 48 dB. This is correlated with a low SNR and is audibly noisy. But if we record 16 bits per sample, the dynamic range increases to a maximum of 96 dB—a major improvement. 16-bit audio became the standard of the compact disc. A 20-bit converter offers a potential DR of 120 dB, which corresponds roughly to the range of the human ear.

This discussion assumes that we are using a *linear PCM* scheme that stores each sample as an integer representing the value of each sample. Blesser (1978), Moorer (1979b), and Pohlmann (2010) have reviewed the implications of other encoding schemes. Some encoding schemes (e.g., MP3, discussed later) have the goal of reducing the total number of bits that the system stores or transmits.

Quantization Noise

Samples are often represented as integers. If the input signal has a voltage corresponding to a sample value between 53 and 54, for example, then the converter might round it off and assign a value of 53. In general, for each sample taken, the value of the sample usually differs slightly from the value of the original signal. This problem in digital signals is known as *quantization error* or *quantization noise* (Blesser 1978; Maher 1992; Lipshitz, Wannamaker, and Vanderkooy 1992; Pohlmann 2010).

Figure 4.2 shows the kinds of quantization errors that can occur. When the input signal is something complicated like a symphony, and we listen to just the errors, shown at the bottom of figure 4.2, it sounds like noise. If the errors are large, then one might notice something similar to analog tape hiss at the output of a system.

The quantization noise is dependent on two factors: the input signal itself, and the accuracy with which the signal is represented in digital form. We can explain the sensitivity to noise in the input signal by noting that on an analog tape recorder, tape imposes a soft halo of noise that continues even through periods of recorded silence. But in a digital system there is no quantization noise when nothing (or silence) is recorded. In other words, if the

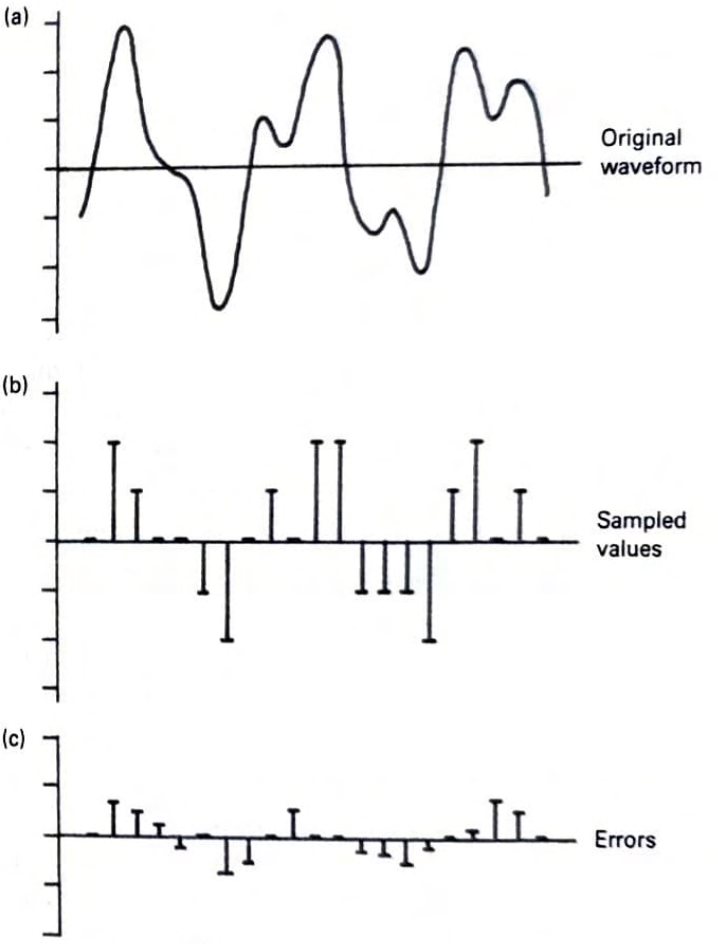

Figure 4.2 Effects of quantization. (a) Analog waveform. (b) Sampled version of the waveform in (a). Each sample can be assigned only certain values, which are indicated by the short horizontal dashes at the left. The difference between each sample and the original signal is shown in (c), where the height of each bar represents the quantization error.

input signal is silence, then the signal is represented by a series of samples, each of which is exactly zero. The small differences shown in figure 4.2c disappear for such a signal, which means that the quantization noise disappears. If the input signal is a pure sinusoid, however, then the quantization error is not a random function but a deterministic truncation effect that can be audibly gritty at low levels (Maher 1992; Stuart and Craven 2019). We discuss this further in the section on dithering.

The second factor in quantization noise is the accuracy of the digital representation. In a linear PCM system that represents each sample value by an integer, quantization noise is directly tied to the number of bits that are used to represent a sample. As previously noted, this is the *quantization level* of a system. Figure 4.3 illustrates the effects of different quantization levels, comparing the resolution of 1-bit versus 4-bit quantization. In a linear PCM system generally, the more bits used to represent a sample, the less the quantization noise.

Figure 4.4 shows the improvement in sine wave accuracy achieved by adding more bits of resolution, going from 2^4 (sixteen possible values) to 2^8 (256 possible values). Consider the improvement in precision accrued by using a 24-bit sample with 2^{24} or more than 16.7 million possible values.

The process of sampling can be viewed as fitting a waveform to a grid of time versus amplitude, as shown in figure 4.5. In general, the finer the grid, the better the approximation to the original waveform. More specifically, the finer the time grid (or sampling rate), the greater the bandwidth. The finer the amplitude grid (or quantization level), the greater the dynamic range and the smaller the amount of noise.

Low-Level Quantization Noise and Dither

Although a digital system exhibits no noise when there is no input signal, at very low (but nonzero) signal levels, quantization noise takes a pernicious form. This gritty sound, called *granulation noise* or *modulation noise*, can be heard when low-level tones decay to silence. A very low-level signal triggers variations only in the lowest bit. These 1-bit variations look like a square wave, which is rich in odd harmonics. Consider the decay of a piano tone, which smoothly attenuates with high partials rolling off—right until the lowest level when it changes character and becomes a harsh-sounding square wave. The harmonics of the square wave can extend even beyond the Nyquist frequency, causing aliasing and introducing new frequency components that were not in the original signal. These artifacts may be possible to ignore if the signal is kept at a low monitoring level, but if the signal is heard at a high level or if it is rescaled to a higher level (a common practice in

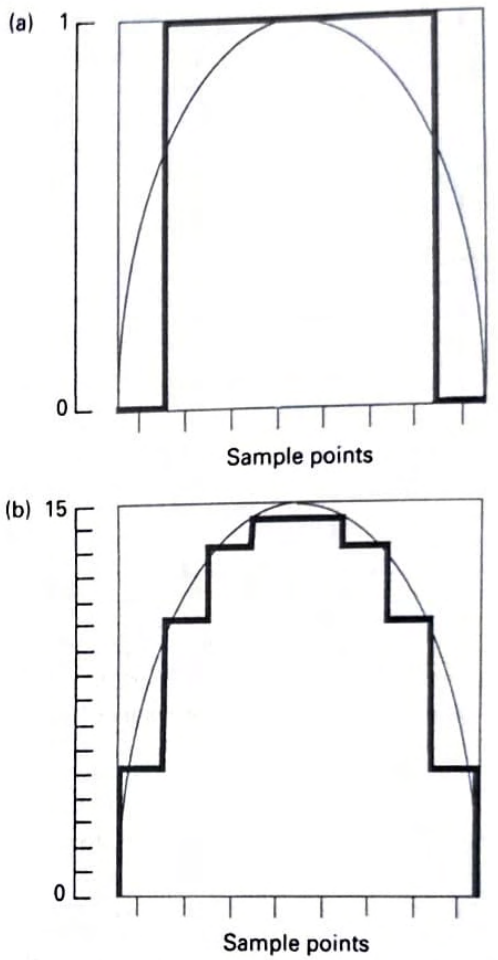

Figure 4.3 Comparing the accuracy of 4-bit quantization with that of 1-bit quantization. The thin rounded curve is the input waveform. (a) 1-bit quantization provides two levels of amplitude resolution. (b) 4-bit quantization provides sixteen different levels of amplitude resolution.

electronic music), it becomes more obvious. Hence it is important that the signal be quantized as accurately as possible at the input stage.

To confront low-level quantization problems, some digital recording systems take a seemingly strange action. They introduce a small amount of uncorrelated noise—called *dither*—to the signal prior to analog-to-digital conversion (Vanderkooy and Lipshitz 1984; Lipshitz et al. 1992; Stuart and Craven 2019). This causes the ADC to make random variations around the low-level signal, which smooths out the pernicious effects of square wave harmonics (figure 4.6). With dither, the quantization error, which is usually signal-dependent, is turned into a wideband noise that is uncorrelated with the signal. For decrescendos like the piano tone mentioned previously, the effect is that of a soft landing as the tone fades smoothly into a bed of

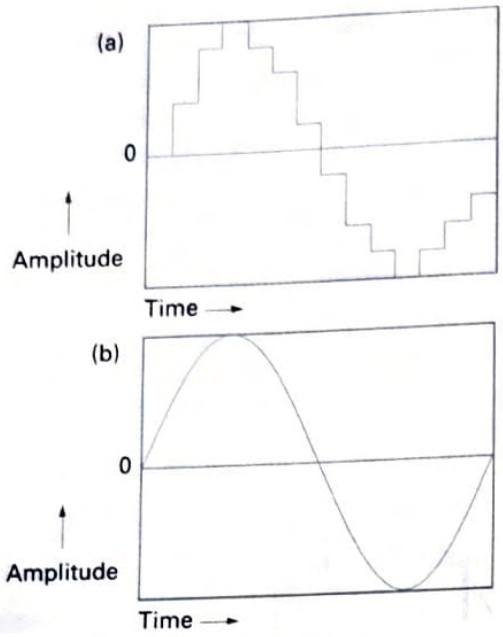

Figure 4.4 Effect of quantization on sine wave smoothness. (a) Sine wave with ten levels of quantization, corresponding to a moderately loud tone emitted by a 4-bit system. (b) Smoother sinusoid emitted by an 8-bit system.

low-level random noise. The amount of added noise is usually on the order of 3 dB, but the ear can reconstruct musical tones whose amplitudes fall below that of the dither signal.

Dithering is recommended in recording at a 16-bit quantization level. Adding dither may not be necessary in recording with a 24-bit converter because the low bit represents an extremely soft signal. But in requantizing signals from a 24-bit to a 16-bit format, for example, dithering is recommended to preserve signal fidelity (Stuart and Craven 2019). Many dithering algorithms exist, so sound editors and mastering plug-ins often provide a variety of options.

Converter Linearity

Audio converters can cause a variety of distortions (Blesser 1978; McGill 1985; Talambiras 1985; Pohlmann 2010). One such problem is that an n-bit converter is not necessarily accurate to the full dynamic range implied by its n-bit input or output. Although the *resolution* of an n-bit converter is one part in $2n$, a converter's *linearity* is the degree to which the analog and digital input and output signals match in terms of their magnitudes. That is, some converters use $2n$ steps, but these steps are not linearly spaced, which causes distortion. Hence it is not unusual to encounter a 24-bit converter, for example, that is actually 19 bits linear. Refer to Pohlmann (2010) for a discussion of these issues.

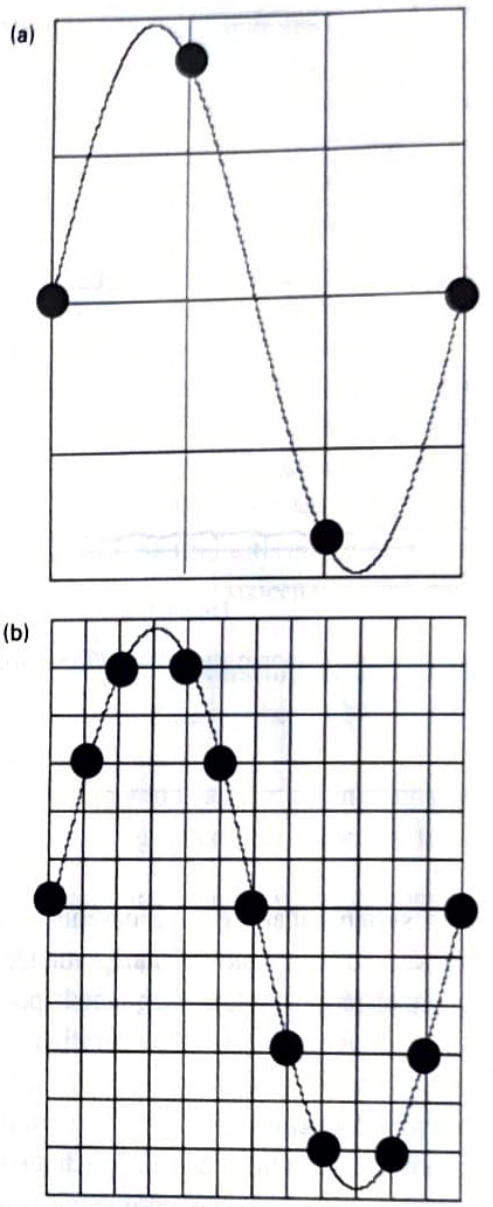

Figure 4.5 The sampling grid. The horizontal axis is time. The vertical axis is amplitude. (a) Crude approximation of a sine waveform caused by low sampling rate and low quantization. (b) Increasing grid resolution results in a better approximation to the waveform. Greater increases in grid resolution would closely approximate the original waveform.

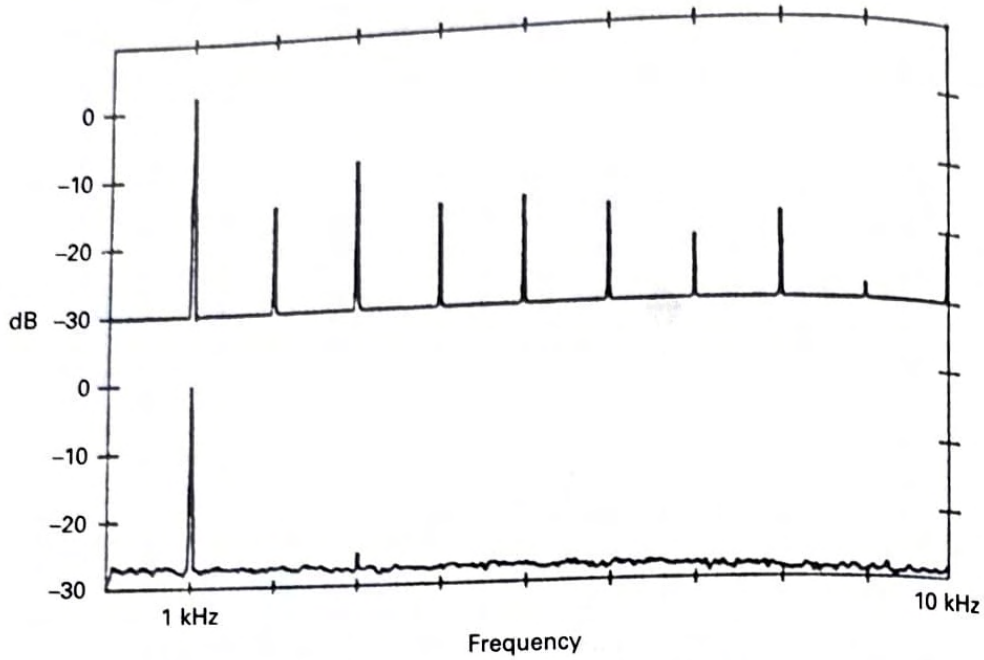

Figure 4.6 Dither reduces harmonic distortion. (*Top*) Original signal. (*Bottom*) Postdithered signal.

Oversampling Converters

In the digital audio systems that we have described, *multibit linear DACs* transform a binary sample value into an analog voltage in one step. That is, they convert a sample of 16 to 24 bits at each sample period. Correspondingly, multibit linear ADCs perform the inverse operation: converting an analog voltage into a multibit sample.

By contrast, *oversampling converters* use more samples in the conversion stage than are actually stored in the recording medium. For our purposes it is sufficient to present the basic ideas, leaving references for those who wish to investigate the topic further.

Oversampling is a family of methods for increasing the accuracy of converters. Most methods rely on 1-bit oversampling (Adams 1990; Hauser 1991; Reiss 2008). These methods convert just one bit at a time at a high sampling frequency.

1-Bit Converters

The theory of 1-bit oversampling converters goes back to the 1950s (Cutler 1960), but it took years for this technology to become incorporated into digital audio systems. The 1-bit oversampling converters constitute a family of dif-

ferent techniques that are variously called *sigma-delta*, *delta-sigma*, *noise shaping*, *bitstream*, *MASH*, or *direct stream digital (DSD)*, depending on the manufacturer. Class D amplifiers, popular in mobile phones and high-power sound reinforcement systems, incorporate this technology (Reiss 2008).

They have the common thread that they sample one bit at a time at high sampling frequencies. That is, rather than trying to represent the entire waveform in a single sample, these converters measure differences between successive samples. They need to measure only whether the waveform has made a positive or negative excursion since it was last sampled, but they do this so frequently that the waveform does not have time to make a large excursion, and so 1 bit of information per sample period is sufficient.

1-bit converters take advantage of a fundamental law of information theory (Shannon and Weaver 1949), which says that one can trade off sample width for sample rate and still convert at the same resolution. That is, a 1-bit converter that oversamples at sixteen times the stored sample rate is equivalent to a 16-bit converter with no oversampling. They both process the same number of bits. The benefits of oversampling accrue when the sampling rate is much higher, meaning that the number of bits being processed is greater than the number of input bits.

One way to describe a sampling system is to determine the total number of bits being processed, according to the relation

Oversampling factor × Width of converter

For example, a 128-times oversampling system that uses a 1-bit converter is processing 128×1 bits each sample period. This compares to a traditional 16-bit linear converter that handles 1×16 bits, or one-eighth the data. In theory, the oversampling 1-bit converter should be much cleaner sounding.

In any case, all the benefits of oversampling accrue to 1-bit converters, including increased resolution and phase linearity due to digital filtering. High sampling rates that are difficult to achieve with the technology of multibit converters are much easier to implement with 1-bit converters. Oversampling rates in the 2.8224 MHz and greater range permit highly accurate quantization per sample (Moorer 1996). The Swiss-based Pyramix digital audio workstation supports high-resolution DSD recording. DSD has evolved into a high-resolution download and streaming format for audiophiles, with some converters supporting oversampling rates in excess of 22 MHz (Melchior 2019).

Noise Shaping

Another technique commonly used in 1-bit oversampling converters is *noise shaping*, which can take many forms (Hauser 1991; Reiss 2008). The basic

idea is that the *requantization error* that occurs in the oversampling process is shifted into a high-frequency range, either to a less noticeable frequency band or out of the audio bandwidth, by a highpass filter in a feedback loop with the input sign/al. This *noise-shaping loop* sends only the requantization error through the highpass filter and not the audio signal.

The final stage of any oversampling converter is a decimator that reduces the sampling rate of the signal to that required for storage (for an ADC) or playback (for a DAC) and also lowpass filters the signal. In the case of a noise shaping converter this also removes the requantization noise, resulting in dramatic improvements in signal-to-noise ratio. With *second-order noise shaping* (so called because of the second-order highpass filter used in the feedback loop), the maximum signal-to-noise level of a 1-bit converter is approximately equivalent to 15 dB (2.5 bits) per octave of oversampling, minus a fixed 12.9 dB penalty (Hauser 1991). By running a 20-bit converter at 256 times the target sampling rate, one achieves 24-bit resolution, in theory. In practice, a host of issues in 1-bit oversampling are caused by feedback around a nonlinear quantizer. These include limits cycles, idle tones, distortion, dead zones, instability, and other scary phenomena (Reiss 2008).

Chapter 9 presents the basics on the related issue of *sample-rate conversion,* which is discussed in engineering texts in the context of *multirate signal processing* (Mitra 2006).

Digital Audio Media and Formats

Audio samples can be stored on any digital medium, typically optical or magnetic disks, solid-state drives, and memory chips. On a given storage medium, data files can be stored in a variety of *audio file formats.* The software that performs the encoding and decoding of the audio data is called a *codec* (coder-decoder).

An audio file format like AIFF or WAVE is a data structure that divides the file into subsections, each of which contains a specific type of data. For example, one section stipulates the sample rate, bit resolution, and how many channels of digital audio are stored in the file. Another section contains the raw sample data. Other sections can contain pointers to markers, loop points, and other kinds of information such as file name, author, copyright, and so on.

New media, codecs, and formats are constantly being developed. Scientific advances push these technologies, but just as often the introduction of a new medium, codec, or format is driven by a commercial strategy.

Lossless Versus Lossy File Formats

Three types of file formats are common:

1. Uncompressed
2. Compressed lossless
3. Compressed lossy

An *uncompressed* format (e.g., AIFF or WAV) stores full resolution PCM digital audio waveforms without any data reduction. This includes 16-bit and 24-bit formats at sample rates from 22.05 to 384 kHz. A special uncompressed format is 32-bit *floating-point* (FP), based on the IEEE 754 standard. This format encodes audio as a 24-bit mantissa that can be scaled by an 8-bit exponent, resulting in an effective dynamic range of about 1,500 dB. FP recording is provided in field recorders made by companies such as Zoom and Sound Devices. Audio editors such as Audacity, Adobe Audition, and Reaper also support the FP format.

An advantage of lossless formats is that one can transfer the bits from one medium to another in real time through *digital input/output connectors* (hardware jacks on the playback and recording systems) and *standard digital audio transmission formats* such as AES3 (two channels), S/PDIF (two channels), AES10 or MADI (56 channels), Ethernet 802.1 audio visual bridging (AVB) (200 channels), and Dante (1024 channels). Johns (2017) explains the proliferation of Ethernet audio formats.

A *compressed lossless* format, such as free lossless audio codec (FLAC) or Apple lossless (ALAC), takes advantage of redundancies in the data to pack it more efficiently without losing data. However, the file needs to be unpacked before it can be played. The original waveform can be perfectly reconstructed after it is unpacked. The ubiquitous ZIP and RAR file formats are examples of compressed lossless formats, but because they are not optimized for audio they typically achieve a 20 percent reduction in file size. By contrast, an audio-specific format like FLAC can achieve as much as a 50 percent reduction in file size.

A *compressed lossy* format, such as MP3, Vorbis, advanced audio codec (AAC), ATRAC, or Windows Media Audio), analyzes the input file to determine what information can be thrown away in order to meet a desired minimum bit rate. The goal is not only small file size but also minimum downloading time. The audio quality of these formats tends to be mediocre.

A 128 kbit/s MP3 file is about eleven times smaller than an uncompressed 16-bit 44.1 kHz compact disc track of the same song. The codecs associated with these formats usually let users choose the degree of data reduction or

loss when the file is encoded. The term *lossy* refers to the fact that as MP3 encodes audio signals, it discards parts of sound that are deemed less audible in order to compress the size of a sound file. Specifically, MP3 throws away frequency components that are *spectrally masked* by neighboring frequencies, as well as tones that are *temporally masked* by loud events that occur less than about 5 ms before them (Hacker 2000).

Basing audio compression on psychoacoustic factors is called *perceptual coding* (Gibson, et al. 1998). The amount of lossy compression is controlled by the *bit rate* parameter, which for MP3 typically varies from 32 to 320 kbits/s and the sampling rate (from 32 to 48 kHz). The bit rate has a direct impact on both file size and audio quality; a low bit rate means lower audio resolution. One encoding option is *variable bit rate* (VBR) that uses a low bit rate for simple passages (e.g., sustained tones) and a high bit rate for more complex passages, such as transient or noisy events.

In essence, an MP3 encoder subdivides the input signal into thirty-two spectral bands and measures the energy in each of these bands over time. Applying perceptual coding techniques, it then greatly reduces the analyzed data. For example, if the content in any given band falls below a threshold of audition (varying according to frequency), the encoder discards that band.

MP3 playback is performed by what is basically an additive synthesizer. Designed for mass distribution, MP3 files are often played back on cheap loudspeakers and earbuds where high audio quality is not a consideration. For an analysis of the myriad problems of sound quality in MP3 files, read Corbett (2012).

Formats like MP3 and AAC (used in YouTube audio) were conceived when data storage was expensive and network speeds were measured in hundreds of bits per second. Today storage is cheap and network speeds are measured in gigabits per second. Thus the need for music distribution via lossy formats is diminished, much to the relief of recording engineers (Faulkner 2011). The audiophile market has largely moved to high-resolution lossless download and streaming formats such as FLAC, ALAC, and DSD (Melchior 2019).