High-Performance Computing, Music Composition, and the Sonification of Scientific Data

Hans G. Kaper
Mathematics and Computer Science Division, Argonne National Laboratory
Sever Tipei
School of Music, University of Illinois
Elizabeth Wiebel
Mathematics and Computer Science Division, Argonne National Laboratory

Music composition and scientific computing are usually considered separate and unrelated fields of intellectual activity. The thought of a musical piece being the result of involved calculations is not a familiar one; nor is that of using sound objects to present the results of large-scale scientific computations. However, an unusual collaboration centered on the high-performance computing capabilities at Argonne National Laboratory pursues precisely these two ideas.

For the past few years, the authors have been engaged in a project to develop high-performance computing tools for the world of sound. Initially, the goal of the project was to see how a composer of serious music might benefit from access to the latest supercomputing technology. The idea of using computers for music composition was certainly not new; already in the 1950s, Lejaren Hiller performed many experiments at the University of Illinois [1], and the premiere of his Quartet No. 4 for strings "Illiac Suite" [2] in May 1957 is generally regarded as the birth of serious computer music. Computers have helped many composers since to algorithmically synthesize new sounds and produce new pieces for acoustical as well as digital instruments (see Box 1). Given the power of today's high-performance computing architectures, one could reasonably expect a significant increase in the quality of synthesized sounds and possibly realize a music composition "in real time." This expectation provided the original impetus for the project.

Currently, the centerpiece of the project is `DIASS`, a Digital Instrument for Additive Sound Synthesis. `DIASS` is part of a comprehensive *Environment for Music Composition* (Fig. 1). The *Environment* includes, besides `DIASS`, software for computer-assisted composition, automatic music notation, and visualization of sound objects in a multimedia environment. Not all components are as fully developed as `DIASS`, but even now the user can write a work with the help of the computer, have a score of the instrumental or vocal parts printed in traditional music notation, produce a tape

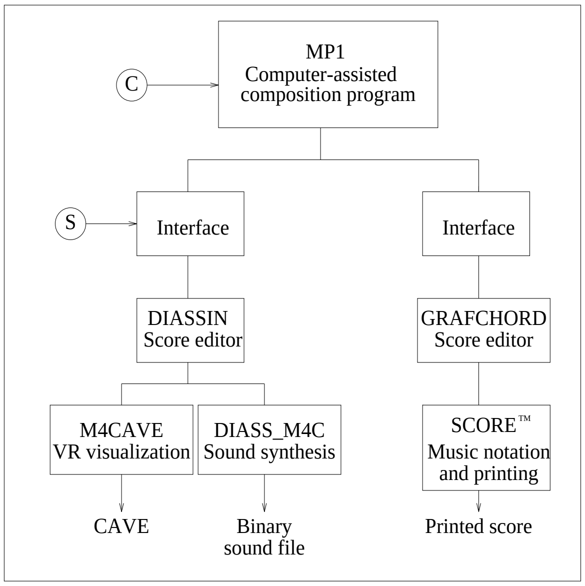

Figure 1: The *Environment for Music Composition*, with data entry points for composition (C) and sonification (S).

of digitally synthesized sounds, and have those sounds represented as graphic objects in a virtual-reality environment.

Early results obtained with the *Environment for Music Composition* include the *Argonne Chime* (see Box 2), a set of demonstration pieces, a work for voices and tape, and a manifold composition entitled *ANL-folds*. These compositions yielded intriguing evidence that DIASS, the sound synthesis software that was designed and implemented primarily with music in mind, was sufficiently general and flexible that it might be useful for other applications as well. In particular, attention focused on its potential for sonification—the rendition of complex scientific data in aural images. In this article we describe some of the capabilities of DIASS and the environment in which it operates and indicate how we intend to develop it further for scientific sonification.

DIASS

Conceived as a composer's tool giving the user minute control over every conceivable detail of a sound, DIASS is intended to run on machines that offer both high speed and large amounts of memory [3]. It consists essentially of two parts: the instrument proper, which computes the samples, and an editor, through which the user enters and modifies the instructions for the instrument. The DIASS instrument functions as part of the M4C synthesis language developed by Beauchamp and his associates at the University of Illinois [4]. As part of the project, the instrument and relevant parts of the M4C code were redesigned for a distributed-memory environment. The parallel implementation uses the standard MPI message-passing library [5].

The Instrument

Like all additive-synthesis instruments, DIASS creates complex sounds through a summation of simple sine waves (partials) (see Box 3). Unlike other similar instruments, however, DIASS can handle an arbitrary number of complex sounds, each sound can be made up of an arbitrary number of partials, and each partial can be controlled in many different ways (see Box 4). Because of these features, DIASS is currently the most flexible instrument for sound synthesis.

A unique feature of DIASS is its capability to control the loudness of a sound and synthesize sounds that are perceived by the listener as being "equally loud." The perception of loudness is a subjective experience; the loudness routines in DIASS implement relevant results of psychoacoustics research in software (see Box 5).

The results of the synthesis process are stored as 16-bit integers sampling the sound at a rate of at least 22,050 Hz. (The recording industry standard is 48,000 Hz.) The loudness routines in DIASS enable the user to produce an entire musical work in a single run, even when the sounds in the work cover a wide dynamic range. When the computation of a sample results in a number outside the interval $(-2^{15}, +2^{15})$, DIASS automatically scales the entire sound. This scaling eliminates "clipping" — the popping noise that occurs as the result of overflow. To appreciate the difficulty inherent in the scaling process, consider the case of a sound cluster consisting of numerous complex sounds, all very loud and resulting in clipping, followed by a barely audible sound with only two or three partials. If the cluster's amplitude is brought down to 2^{15} and that of the soft tiny sound following it is scaled proportionally, the latter disappears under system noise. On the other hand, if only the loud cluster is scaled, the relationship between the two sound events is completely distorted. Many times in the past, individual sounds or groups of sounds were generated separately and then merged with the help of analog equipment or an additional digital mixer. The loudness routines in DIASS deal with this problem by adjusting both loud and soft sounds, so their perceived loudness is equal to the one specified by the user, while preventing overflow.

The Editor

Features like the equal-loudness routines make `DIASS` a fine-tuned, flexible, and precise instrument. They are costly, though, and require not only intensive computations, but also significant amounts of input data. Specifying the details of a complex sound is laborious, error prone, and time consuming. The editor in `DIASS` is designed to facilitate this process. It comes in a "slow" and a "fast" version.

In the slow version, data are entered one at a time, either in response to questions from a menu or through a graphic user interface (GUI). The process gives the composer the opportunity to experiment and build sounds step by step. However, considering that one minute of music represents 1–200 sounds and that entering the data for each sound takes at least a few minutes, the slow editor does not offer a practical way to generate a piece. The alternative fast version uses the same code, but reads the responses to the menu questions from a script that is created by the computer-assisted composition program. Once the composer has decided what kind of sounds will be included in a work, the necessary information is passed along with the help of a composition program through the script.

In both the slow and the fast version, certain operations affecting not just one partial but the entire sound are handled through "macros." The macros relieve the user from the burden of specifying individual control parameters for each partial. For example, the loudness is specified for an entire sound, and all the component waves are automatically scaled according to a spectrum specified ahead of time. When a glissando is desired, macros ensure that all partials slide at the same pace. Similarly, "tuning" and "detuning" are accomplished by smooth changes of the frequency ratios of a sound's overtones. When applying amplitude or frequency modulation, all waves belonging to the same sound are affected at the same rate; and if a random element is present, all waves access the same random number sequence. The last feature is especially significant, because psychoacoustic tests have shown that it is an important factor in the perception of a complex wave as one sound, rather than a collection of independent sine waves.

Global operations as encapsulated in the macros are indeed time saving and, in certain situations, necessary. However, by digitally recreating functions of various pieces of equipment one used to find in the old analog studios—wave generators, filters, mixers, reverberating units, etc.—the macros tend to reinforce the old paradigm on which most traditional sound-synthesis software is based. Behind them is the even more entrenched habit of looking at music as being made up of sounds produced by an orchestra of instruments and read from a score. `DIASS` indeed accommodates this type of traditional thinking. But at the same time for the composer of serious music, it can stimulate new ways of thinking about sound. The composition *ANL-folds* exemplifies such a nontraditional approach to music composition (see Box 6).

Computing Requirements

The sound synthesis software embodied in `DIASS` is computationally intensive and requires significant amounts of memory. For example, a ten-minute sound file for which the 48 kHz sampling rate is used has a size of approximately 115 MB. Even if sampled at 22,050 Hz, the file will still have over 50 MB. Moreover, `DIASS` was conceived as a "Rolls Royce bulldozer," a combination of finesse and power where the complexity of the task is never sacrificed for the sake of expediency. The idea behind its design was to always use the maximum computing power available, under the assumption that in a not too distant future the same power would be accessible from the user's desktop.

The instrument proper, the engine that computes the samples, comes in a sequential and a parallel version [6]. The parallel version has been implemented on the IBM Scalable POWERparallel (SP) system. At least four nodes are required for each job—one node to distribute the tasks and supervise the entire run (the "master" node), a second to mix the results (the "mixer"), and at least two "slave" nodes to compute the samples one sound at a time. Parallelism is implemented at the sound level, and not at the wave (partial) or at the sample level, to minimize communication among the nodes and enable all partials of a complex wave to access the same sequence of random numbers, as required by some macros.

The performance depends greatly on the sophistication of the sounds—that is, on the number of partials per sound and the number of active controls for each partial. An *ANL-folds* variant of 2'30" comprising 4–500 sounds of medium to great complexity can be computed on 30 nodes in 20 minutes or less. The speed increases with the number of nodes, but load imbalances result in a sublinear speedup. Sounds are at present computed in their starting-time order, irrespective of their duration or complexity. A smart load-balancing algorithm is necessary, especially when there is a significant imbalance between the duration of various sounds or the number of their partials. We estimate that, in certain cases, it could reduce the run time to one-half the present value.

For files of only a few sounds, the editor runs almost in real time; for longer files, such as the *ANL-folds*, the computation time is at present two orders of magnitude greater. We expect that a restructuring of the I/O features now under way and eventually parallelizing this part of the code will alleviate this problem.

With the exception of the composition programs, which are written in FORTRAN, all the software is in the C language. M4CAVE, employed for the visualization of sound objects in Argonne's virtual-reality environment (CAVE), uses OpenGL.

Scientific Sonification

Computational scientists currently rely on powerful visualization systems, including real-time interactive display techniques and virtual-reality environments, to analyze and view the results of large-scale numerical simulations. Although sound offers another medium for the representation of complex data, its use in scientific computing has received little attention, mostly because of the lack of a suitable instrument. We claim that `DIASS`, which can probe multidimensional datasets with surgical precision, offers the first realistic possibility for scientitifc sonification—the faithful rendition of complex scientific data in aural images.

A number of efforts have been made in the past to combine visual images and sounds for data analysis; see [7, 8] and the references cited there. However, most attempts have been limited in scope. Some were based on the use of "earcons"—aurally recognizable identities like whistles, beeps, and sirens—to signal changes in the broad characteristics of the data; others used MIDI-controlled synthesizer sounds, which have drastic limitations in the number and range of their control parameters. The software developed by Bargar et al. [9] at the National Computational Science Alliance (NCSA) at the University of Illinois at Urbana-Champaign (UIUC), which is certainly the most complex in this category, includes the VSS sound server for the CAVE and has interactive capabilities. While these efforts recognized the need for sonification, they did not lead to an adequate tool. `DIASS`, on the other hand, can provide equivalences for the most intimate details of computed objects in a multidimensional space and appears eminently suitable as an instrument for scientific sonification.

What We Have Done So Far

Much of our work so far has been focused on the representation of complex sounds as graphical objects in the CAVE—the room-size virtual-reality environment at Argonne

National Laboratory [10]. Images are computed on the fly from precomputed sound files and are made to correspond exactly to the sounds through a one-to-one mapping between visual attributes and sound qualities. Partials are represented as spheres, whose size is proportional to loudness and whose height corresponds to frequency (Fig. 2). An optional grid in the background shows the octaves divided into twelve

Insert Figure 2 here

Figure 2: Visualization of nine sounds in the CAVE.

equal increments. A sound's position in the stereo field determines the placement of the sphere between far left and far right in the room. Visual objects rotate or pulse when tremolo or vibrato is applied, and their color varies when reverberation is present. In other renditions, we may choose a cloud of confetti-like objects or a collection of colored planes floating in space, which change size, color, and degree of internal activity according to the same musical parameters. These experiments, though limited in scope, have enabled us to explore different mappings from the space of data to the space of sounds.

What We Have Found So Far

The combination of visual images and sounds provides an extremly powerfool tool for uncovering complicated structures. Sometimes, the sounds reveal features that are hidden in the visual images; at other times, the visual images illuminate features that are not easily detectable in the sound. The two modes of perception reinforce each other, and both improve with practice.

Besides obvious variables such as time, frequency, and amplitude, it is possible to identify specific combinations of partials in a sound (a main ingredient in timbre recognition), along with modifiers such as amplitude and frequency modulation (tremolo and vibrato) and reverberation. Most of these modifiers lump two, three, or more degrees of freedom together: magnitude, rate, and phase of the modulation in the case of tremolo and vibrato; hall size, duration of the reverberated part of the sound, and ratio between direct and reverberated sound in the case of reverberation; and so on. It is easy to see how these modifiers can add up to a large number of recognizable and controllable parameters.

Like the eye, the ear has a very high power of discrimination. Even a coarse grid, such as the temperate tuning used in Western music, includes about 100 identifiable discrete steps over the frequency range encompassed by a piano keyboard. Contemporary music, as well as some non-Western traditional music, successfully uses smaller increments of a quarter tone or less for a total of some 200 or more identifiable steps in the audible range. Equally discriminating power is available in the realm of timbre.

Most auditory processes are based on the recognition of time patterns (periodic repetitions giving birth to pitch, amplitude or frequency modulation; spectral consistency creating stable timbres in a complex wave; formants or filtering patterns produced by the body of an instrument; etc.), and the ear is highly attuned to detect such regularities. On a higher level, music deals with rhythmic, metric, and pitch patterns, and with harmonic aggregates or textures that are restated or varied. Sound is an obvious means to identify regularities in the time domain, both at the microlevel and on a larger scale, and to bring out transitions between random states and periodic happenings.

Most conceptual problems in scientific sonification are related to finding suitable mappings between the space of data and the space of sounds. Common sense points toward letting the two domains share the coordinates of physical space-time if these are relevant and translating other degrees of freedom in the data into separate sound parameters. On the other hand, it may be advantageous to experiment with alternative mappings. Sonification software must therefore be sufficiently flexible that a user can pair different sets of parameters in the two domains.

Any mapping between data and sound parameters must also allow for redundancies to enable the exploration of data at different levels of complexity. Similar to visualization software, sonification software must have utilities for zooming, modifying the audio palette, switching between visual and aural representation of parameters, defining time loops, slowing down or speeding up, etc.

Through the proper manipulation of reverberation, loudness, and spectrum, one can create the illusion of sounds being produced at arbitrary locations in a room, even with only two speakers. Surround-sound systems featuring an array of speakers, already presented at the Philips Pavilion at the 1958 Brussels World Fair (see [11]) and currently promoted as essential for the contemporary home entertainment system, will be the dinosaurs of the digital sound world.

Where We Would Like to Go

The experiments done so far are a first step in the process of finding effective ways to represent scientific data in sound objects. They show convincingly that the sound domain greatly increases the bandwidth for the perception of complex structures and point to useful approaches to scientific sonification.

Our next goal is to use `DIASS` and the CAVE to explore complex data sets obtained in large-scale numerical simulations of physical phenomena. Several sources of data from materials science, structural biology, and astrophysics are available. Progress will require the enhancement of existing software and the development of new capabilities.

Larger Issues

This project is unusual in its interdisciplinary aspect. It also shows that the way in which both composers and scientists perform their tasks might be about to change in several respects.

For the composer, the use of high-performance computers means the possibility to experiment more and obtain results faster. But it also means that a lot of extra help is needed. No single person can possibly have the required expertise in so many and diverse realms: music, psychoacoustics, mathematics, and computer science. The romantic artist laboring in isolation to express his intimate feelings gives way to teamwork concerned with specific objectives. The concept of manifold composition challenges the very idea of an "art object" fixed in its form, available for repeated inspection or used as an investment. Interactive possibilities offered by virtual-reality systems and by the Internet may soon change even more the way in which music is composed, performed, and distributed.

Ours is a visually oriented culture *par excellence*, and as a society we watch rather than listen. Contemporary musical culture is often reduced to entertainment genres that use an outdated grammar and a simple-minded vocabulary. With an open mind and an awareness for unusual and unexpected sonorities, the scientists may yet discover the benefits of the world of sound.

Generally speaking, both scientists and musicians could gain by becoming more familiar with each other's work and ways of thinking. Such a rapprochement could be facilitated by offerings in the academic curriculum. A "Discovery" course entitled

Music, Science, and Technology, where some of the issues presented here will be discussed, is currently under development and will be offered for the first time at the UIUC in the fall of 1997.

Acknowledgments

This work was partially supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

References

[1] L. Hiller and L. Isaacson, *Experimental Music*, McGraw-Hill, 1959; reprinted by Greenwood Press, 1983.

[2] L. Hiller, *Computer Music Retrospective*, Compact disc WER 60128-50, WERGO Schallplatten GmbH, Mainz, Germany, 1989.

[3] C. Kriese and S. Tipei, "A compositional approach to additive synthesis on supercomputers," *Proc. 1992 Int'l. Computer Music Conference* (San Jose, California), pp. 394–395.

[4] J. Beauchamp, *Music 4C Introduction*, Computer Music Project, School of Music, University of Illinois at Urbana-Champaign, 1993. WWW: http://cmp-rs.music.uiuc.edu/cmp/software/m4c.html.

[5] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, MIT Press, 1994.

[6] H. G. Kaper, D. Ralley, J. M. Restrepo, and S. Tipei, "Additive synthesis with DIASS_M4C on Argonne National Laboratory's IBM POWERparallel System (SP)," *Proc. 1995 Int'l. Computer Music Conference* (Banff, Canada), pp. 351–352.

[7] R. M. Baecker, J. Grudin, W. Buxton, and S.'Greenberg, *Readings in Human-Computer Interaction: Toward the Year 2000*, second edition, Morgan Kaufmann Publ., Inc., San Francisco, 1995.

[8] G. Kramer (ed.), *Auditory Display: Sonification, Audification, and Auditory Interfaces*, Addison-Wesley Publ. Co., 1994.

[9] R. Bargar, I. Choi, S. Das, and C. Goudeseune, "Model-based interactive sound for an immersive virtual environment," *Proc. 1994 Int'l. Computer Music Conference* (Tokyo, Japan), pp. 471–477.

[10] WWW: http://www.mcs.anl.gov/FUTURES_LAB/index.html.

[11] I. Xenakis, Revue technique Philips, **20**(1), 1958.

[12] I. Xenakis, *Formalized Music: Thought and Mathematics in Musical Composition*, revised edition, Pendragon Press, 1992.

[13] S. Tipei, "The computer: A composer's collaborator," *Leonardo* **22**(2) 1989, 189–195.

[14] S. Tipei, "Manifold compositions — A (super)computer-assisted composition experiment in progress," *Proc. 1989 Int'l. Computer Music Conference* (Columbus, Ohio), pp. 324–327.

[15] J. G. Roederer, *The Physics and Psychophysics of Music*, 3rd edition. Springer-Verlag, 1995.

[16] H. Fletcher and W. A. Munson, "Loudness, its definition, measurement, and calculation," *J. Acoust. Soc. Am.* **5** (1933), 82.

[17] T. D. Rossing, *The Science of Sound*, Addison-Wesley Publ. Co., 1990.

[18] S. S. Stevens, "Neural events and the psychophysical law," *Science* **170** (1970), 1043.

[19] E. Zwicker and E. Terhardt, "Analytical expressions for critical-band rate and critical bandwidth as a function of frequency," *J. Acoust. Soc. Am.* **68** (1980), 5.

[20] E. Zwicker and B. Scharf, "A model of loudness summation," *Psych. Rev.* **72** (1965), 3.

[21] H. G. Kaper, D. Ralley, and S. Tipei, "Perceived equal loudness of complex tones: A software implementation for computer music composition," *Proc. 1996 Int'l. Conference in Music Perception and Cognition* (Montreal, Canada), pp. 127–132.

[22] S. Tipei, "ANL-Folds. mani 1943-0000; mani 1985r-2101; mani 1943r-0101; mani 1996m-1001; mani 1996t-2001" (1996). Preprint ANL/MCS-P679-0897, Mathematics and Computer Science Division, Argonne National Laboratory.

Hans G. Kaper is Sr. Mathematician at Argonne National Laboratory. After receiving his Ph.D. in mathematics from the University of Groningen (the Netherlands) in 1965, he held positions at the University of Groningen and Stanford University. In 1969 he joined the staff of Argonne. He was director of the Mathematics and Computer Science Division from 1987 to 1991. Kaper's professional interests are in applied mathematics, particularly mathematics of physical systems and scientific computing. He is a corresponding member of the Royal Netherlands Academy of Sciences. His main interest outside mathematics is classical music. He is chairman of "Arts at Argonne," concert impresario, and an accomplished pianist. He can be reached at kaper@mcs.anl.gov or http://www.mcs.anl.gov/people/kaper/index.html.

Sever Tipei is professor of composition and music theory at the University of Illinois at Urbana-Champaign (UIUC), where he also manages the Computer Music Project of the UIUC Experimental Music Studios. He has a Diploma in piano from the Bucharest Conservatory in Romania and a DMA in composition from the University of Michigan. Tipei has been involved in computer music since 1973 and regards the composition of music both as an experimental and as a speculative endeavor. He can be reached at s-tipei@uiuc.edu or http://cmp-rs.music.uiuc.edu/people/tipei/index.html.

Elizabeth Wiebel is a participant in the Student Research Participation Program, which is sponsored by the Division of Educational Programs of Argonne National Laboratory. She is an undergraduate student at St. Norbert College in De Pere, Wisconsin, where she is pursuing a degree in mathematics and computer science. In addition, Wiebel studies and teaches piano through the St. Norbert College music department. She can be reached at wiebep@sncac.snc.edu or http://members.tripod.com/ LibbyW.

Box 1. Computer-Assisted Music Composition

Why would a composer need computer assistance when composing? A quick answer is that, like in many other areas, routine operations can be relegated to the machine. A more sophisticated reason may be that the composer may rely on expert systems to write Bach-like chorales or imitate the mannerisms of Chopin or Rachmaninov. There are, however, more compelling reasons when composing is viewed as a speculative and experimental endeavor, rather than as an ability to manufacture pleasing sound objects, as demonstrated a long time ago by Hiller [1], Xenakis [12], and several other composers.

MP1, the computer-assisted composition part in the *Environment for Music Composition* (Fig. 1), illustrates the point. MP1 is based on the idea that the computer is a composer's collaborator [13]. While the composer controls the overall outlook of the music and its abstract structure, the computer provides the details of the piece and ensures a certain degree of randomness unencumbered by cultural conditionings. The composer sets in motion a well-defined process by supplying a set of rules together with the initial conditions and then does not interfere with the process or its end result. The output is accepted as long as it is consistent with the logic of the program and the input data.

In MP1, the particular details of a piece may depend on random occurrences. The same code and data may produce an unlimited number of compositions, which belong to the same "equivalence class" or *manifold composition* [14]. The members of a manifold composition are variants of the same piece; they share the same structure and are the result of the same process, but differ in the way specific events are arranged in time. The global effect depends on the coherence with which the control parameters are chosen. For example, the members of a manifold composition may share certain fixed sections, while other sections diverge in different degrees. A manifold composition is somewhat similar to a serigraph produced by a visual artist, except that its individual members may be more distinct.

A nontraditional way of composing, the manifolds show how high-performace computing provides a composer with the means to try out compositional strategies or materials and hear the results or see them notated on paper in a reasonable amount of time. While some existing commercial software products offers this type of feedback to certain degrees, they do not even approach the sophistication and flexibility required for experimental concert music. The combination of the *Environment for Music Composition* and an advanced computing architecture offers a novel medium for experimentation with and exploration of new paradigms in music composition.

Box 2. The “Argonne Chime”

Insert Figure 3 here

Figure 3: The “Argonne Chime” — the first serious composition produced on the IBM Scalable POWER (SP) System at Argonne National Laboratory.

The “Argonne Chime,” a 21-second sequence of seven sounds representing the letters in “Argonne,” was composed and produced by the authors and premiered at Argonne on October 15, 1995.

A (a), **R** ($re = d$), **G** (g), **O** ($sol = g$), **NN** (percussive), **E** (low e, gong).

Six sounds have six partials; one sound has sixteen partials. The sounds have different degrees of vibrato and tremolo and distinct frequency and amplitude transients. Individual envelopes vary from percussive (**NN**) to sustained (**E**). Various degrees of reverberation and channel placement suggest a realistic environment.

Box 3. Sound Synthesis

The goal in computer music is to produce a sampled-data signal that, when converted to an analog voltage and played through a high-quality sound system, produces the desired musical sound. A synthesis technique is a way to produce such a signal from a mathematical expression. The expression usually has many variables, such as frequency, duration, and loudness, which may change with time over the duration of the sound.

In general, the nth sample x_n, which represents the sound at the time t_n, is computed from an expression of the form $x_n = F(t_n, p_{1,n}, \ldots, p_{N,n})$. Here, $p_{1,n}, \ldots, p_{N,n}$ are the values of the control parameters $p_1, \ldots, p_N$ at the time t_n. The control parameters may have fixed values (static controls) or their values may change with time (dynamic controls). In additive synthesis, one forms the composite sound by summing several sine waves. Each sine wave (partial) is described by its amplitude, frequency, phase, starting time, duration, etc. The actual values of these parameters are controlled by the composer at the level of the partial, at the level of the sound (for example, by prescribing a glissando or a certain level of reverberation), and at the level of the entire piece (for example, through the specification of a global level of loudness).

The fact that controls can act at all levels significantly increases computational complexity and turnaround time.

Box 4. Sound Control in `DIASS`

In `DIASS`, each partial in a sound is controlled in 25 different ways. Some control parameters are static; they do not change for the duration of a sound. Obvious examples are the starting time, duration, and phase of a partial; less obvious parameters define reverberation features, including the size of the "hall" and the apparent strength of the reverberation. Other control parameters are dynamic; their evolution is governed by specified envelope functions, which may be piecewise linear or exponential. Panning (apparent shifting of sound location), tremolo (amplitude modulation), vibrato (frequency modulation), and the relative importance of transients can all affect the parameters of each partial in a sound.

Because each of the 25 control parameters can be applied independently to every partial, a sound that results from the summation of 65 sine waves can be molded in 1,625 diferent ways, about half of them time variant.

One can gain some idea of the input requirements for the synthesis of a typical sound by estimating the data required for a complete specification of the dynamic control parameters. Each dynamic control parameter requires three input items per time interval (starting time, starting value, and type: linear or exponential). Hence, a sound with 65 partials extending over an average of 10 time intervals and subject to 13 dynamic control parameters requires the specification of $65 \times 13 \times 10 \times 3$ data, or about 50 kB. Twenty such sounds would fill a megabyte.

Box 5. Loudness

Loudness is often described as the "strength" or "intensity" of a tone and associated with the energy flow or average pressure variation of the sound wave reaching the ear. However, this description is far too simplistic. The sensation of loudness of a tone of constant intensity varies with the frequency, and the loudness of a superposition of several tones of different frequencies is not related in a simple manner to the total acoustical energy flow. For digital sound synthesis, a more precise definition is needed. The definition requires several steps. (For a more complete discussion we recommend the excellent monograph [15].)

We begin by introducing the (dimensionless) *intensity* I of a pure tone with acoustical energy flux Φ (units of watt/m^2),

$$I = 10 \times \log_{10}(\Phi/\Phi_0).$$

Here, Φ_0 is a reference value, usually taken as the value of Φ at the threshold of hearing, $\Phi_0 = 10^{-12}\mathrm{W/m}^2$. The unit of I is the decibel (dB). Note that I is a relative measure. Because there is no energy flow in a standing wave, one may express I alternatively in terms of the average pressure variation Δp (measured in newton/m^2),

$$I = 20 \times \log_{10}(\Delta p/\Delta p_0),$$

where Δp_0 is the pressure variation for a traveling wave with acoustical energy flux Φ_0, $\Delta p_0 = 2 \times 10^{-5}$ newton/m^2.

Because of the way acoustical vibrations are processed in the cochlea (the internal ear), the sensation of loudness is strongly frequency dependent. For instance, while an intensity of 50 dB at 1,000 Hz is considered *piano*, the same intensity is barely audible at 60 Hz. In other words, to produce a given loudness sensation at low frequencies, a much higher intensity (energy flow) is needed than at 1,000 Hz. The intensity I is therefore not a good measure of loudness if different frequencies are involved.

In the 1930s, Fletcher and Munson [16] performed a series of loudness-matching experiments, from which they derived a set of *curves of equal loudness*. These are curves in the frequency (f) vs. intensity (I) plane; points on the same curve represent single continuously sounding pure tones that are perceived as being "equally loud." The curves show that, in order to be perceived as equally loud, very low and very high frequencies require much higher intensities (energy) than frequencies in the middle range of the spectrum of audible sounds.

The (physical) *loudness level* L_p of a Fletcher-Munson curve is identified with the value of I at the reference frequency of 1,000 Hz. The unit of L_p is the phon. The

Fletcher-Munson curves range from a loudness level of 0 phons to 120 phons over a frequency range from 25 to 16,000 Hz.

The loudness level L_p still does not measure loudness in an absolute manner: a tone whose L_p is twice as large does not sound twice as loud. Following Rossing [17], we define the (subjective) *loudness level* L_s in terms of L_p by the formula

$$L_s = 2^{(L_p - 40)/10}.$$

The unit of L_s is a sone. To be effective, loudness scaling must be done on the basis of sones.

Reasonable approximations in terms of the wave intensity I or pressure variation Δp have been given by Stevens [18]

$$L_s \approx C_s I^{1/3}, \quad L_p \approx C_p (\Delta p)^{2/3}.$$

The constants C_s and C_p depend on the frequency.

So far, we have limited the discussion to pure tones. When two or more tones are superimposed, many things can happen. If their frequencies are the same, the total intensity I is the sum of the individual intensities, and the total perceived loudness is approximately given by

$$L_s \approx C_s (I_1 + I_2 + \cdots)^{1/3}.$$

If their frequencies differ, however, the result depends on how well the frequencies are separated. If the frequencies fall within the *critical band* of the center frequency, the resulting loudness is still directly related to the total intensity, which is the sum of the individual intensities. The width Δf of a critical band centered at a frequency f is approximely given by the expression [19]

$$\Delta f \approx 25 + 75 \left(1 + 1.4 (f/10,000)^2\right)^{0.69}.$$

In the upper range, the critical band corresponds roughly to a pitch interval smaller than a minor third. If the frequency spread exceeds the critical band, the subjective loudness is greater than that obtained by simple intensity summation, increasing with increasing frequency difference and tending toward a value that is given by the sum of the individual loudness contributions from adjacent critical bands [20],

$$L_s = L_1 + L_2 + \cdots .$$

If the frequency difference is very large, we tend to focus on only one of the component tones and assign the sensation of total loudness to just that single component, so

$$L_s = \max\{L_1, L_2, \ldots\}.$$

Finally, a complex sound whose energy is spread across critical bands sounds much louder than one with the same amount of energy all in one critical band. A useful formula for its loudness is [17]

$$L_s = L_m + 0.3 \sum_i L_i,$$

where L_m is the loudness of the loudest band and the sum extends over the remaining bands.

When scaling either a single sound or a number of simultaneous sounds, one must first determine the loudness level (in sones) and then use the desired loudness level (in sones) to compute the required scaling factor. The "loudness routines" in `DIASS` [21] use the Fletcher-Munson curves to determine the loudness level of the constituent waves and then use the critical band information to determine the loudness of complex waves. The scaling involves adjusting the amplitude of every partial in the sound and checking how partials that belong to different sounds interfere with each other within critical bands. The complexity of the scaling operation is further increased by the fact that a new scaling is performed for every segment of the amplitude envelope of each active sound.

Box 6. ANL-folds

Composed with a reduced version of MP1 (Manifold.f), *ANL-folds* exemplifies the nontraditional way of composing made possible by high-performance computing.

ANL-folds, a 2'30" composition, starts with a rich, low (60 Hz) sound, which is followed by an irregular ascending arpeggio made out of the partials of the low sound. The low sound is always the same, but the exact order and duration of the notes in the arpeggio varies from version to version. The ending mirrors the beginning, with a descending arpeggio and a low 60 Hz tone. This time, too, the order and duration of the sounds are slightly different in each realization of the composition. In the middle of the piece, a seven-sound pattern—the *Argonne Chime* (Fig. 3)—is heard, always exactly the same and always at exactly the same moment. In all other sections, however, pitches, durations, attacks, and spectral changes follow stochastic distributions, and every variant of the composition has its own distinct profile. Even broad characteristics, such as density of texture, overall dynamics, and use of particular effects, may vary. These variations derive exclusively from the use of a new seed for the random number generator each time. The "score" of *ANL-folds* [22] consists, therefore, of no more than a collection of sound graphs identified by the seed that was used for the random number generator.

Manifold compositions, such as the *ANL-folds*, represent idiomatic ways of using computers in music composition by mass-producing slightly different and, at the same time, unique versions of the same archetype. The fact that a version is not performed in public more than once should stress the ephemeral quality of any musical activity.