INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

NOTE TO USERS

The CD is not included in this original manuscript. It is available for consultation at the author's graduate school library.

This reproduction is the best copy available.

Data Analysis through Auditory Display: Applications in Heart Rate Variability

Mark Ballora

Faculty of Music
McGill University, Montréal

May, 2000

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements of the degree of
Doctor of Philosophy in Music

Table of Contents

Abstract

This thesis draws from music technology to create novel sonifications of heart rate information that may be of clinical utility to physicians. Current visually-based methods of analysis involve filtering the data, so that by definition some aspects are illuminated at the expense of others, which are decimated. However, earlier research has demonstrated the suitability of the auditory system for following multiple streams of information. With this in mind, sonification may offer a means to display a potentially unlimited number of signal processing operations simultaneously, allowing correlations among various analytical techniques to be observed. This study proposes a flexible listening environment in which a cardiologist or researcher may adjust the rate of playback and relative levels of several parallel sonifications that represent different processing operations. Each sonification "track" is meant to remain perceptually segregated so that the listener may create an optimal audio mix. A distinction is made between parameters that are suited for illustrating information and parameters that carry less perceptual weight, which are employed as stream separators. The proposed sonification model is assessed with a perception test in which participants are asked to identify four different cardiological conditions by auditory and visual displays. The results show a higher degree of accuracy in the identification of obstructive sleep apnea by the auditory displays than by visual displays. The sonification model is then fine-tuned to reflect unambiguously the oscillatory characteristics of sleep apnea that may not be evident from a visual representation. Since the identification of sleep apnea through the heart rate is a current priority in cardiology, it is thus feasible that sonification could become a valuable component in apnea diagnosis.

Résumé

Cette thèse s'inspire de l'informatique musicale pour générer de nouvelles sonifications des informations tirées des battements cardiaques, ce qui pourrait s'avérer utile en milieu clinique pour les médecins. Les méthodes d'analyse visuelle actuelles procèdent par filtrage des données de façon à ce que, par définition, l'emphase soit mise sur certains aspects plutôt que d'autres, ces derniers étant ainsi écartés. Toutefois, des recherches antérieures ont démontrées la capacité du système auditif à décoder plusieurs séries simultanées de données. Grâce à cette aptitude, la sonification peut offrir des moyens de représenter un nombre potentiellement illimité de traitement effectué sur le signal, permettant ainsi l'observation de corrélation par le biais de diverses méthodes analytiques. Cette étude propose un environnement d'écoute versatile dans lequel cardiologistes et chercheurs peuvent ajuster la vitesse de lecture et les niveaux relatifs de plusieurs sonifications simultanées, chacune représentant différentes opérations effectuées sur le signal. Chaque piste de sonification est conçue pour être différencier perceptivement afin que l'utilisateur ait la liberté de réaliser un mixage audio optimal. Dans l'environnement d'écoute une distinction a été faite entre les paramètres aptes a représenter l'information directement pertinente et les paramètres de caractère perceptif sécondaire, ces derniers étant employés à la séparation des séries. Le modèle de sonification proposé a été validé par un test de perception pendant lequel les participants ont dû identifier quatre états cardiologiques differents à l'aide de représentation visuelles et auditives. Les résultats ont démontrés que la représentation auditive permet une plus grande précision de l'identification d'un des états cardiologiques appellé apnée obstructive du sommeil. Le modèle de sonification est ensuite finement réglé pour mettre en exergue de façon indubitable les caractéristiques oscillatoires de l'apnée du sommeil, caracteristique qui ne peuvent pas être mise en évidence par une représentation visuelle. Puisque l'identification de l'apnée du sommeil à partir des battements cardiaques est une importance capitale en cardiologie, la sonification est donc un candidat potentiel de premier choix pour le diagnostic de l'apnée.

Acknowledgements

The range of topics covered herein required consultation with numerous specialists in a variety of fields. I offer my heartfelt thanks to all those listed below. The thesis could not have come into being without them.

To my supervisors, Bruce Pennycook and Leon Glass, for their consistent support over the years that this thesis was in preparation; for the enthusiasm, stringent and meticulous attention to detail, and inter-departmental collegiality that was necessary to produce a work that draws equally from both music and science.

To James McCartney creator of SuperCollider, for providing prompt and to-the-point responses to user questions via internet mailing list; and for his willingness to discuss off-list specific matters pertaining to this thesis.

To Eugenia Costa-Giomi for her guidance and assistance in the preparation of the listening perception test and the analysis of its results.

To Tamara Levitz for her careful attention to the section on music history.

To Philippe Depalle for his careful attention to sections on signal processing and acoustics.

To Andrew Brouse, François Thibault and Philippe DePalle, for translating my abstract into French.

To Plamen Ivanov for sharing his work with me, and his eagerness for continued collaboration.

To the many others who spent time reviewing the work at various stages, offering helpful comments and bringing up new possibilities:
Linda Arsenault
Albert Bregman
Jason "Bucko" Corey
Marc Courtemanche
Poppy Crumb
Ary Goldberger
Michael Guevara
Jeff Hausdorff
Beatriz Ilari
Joseph Mietus
Chung-Kang Peng
Carsten Schaefer
Zack Settel
Geoff "Wonder Boy" Martin

To the administrative assistants who provided access to and requisite paperwork from their busy bosses:
Deborah Diamond
Kathey Wong

All illustrations of RR interval plots are provided courtesy of Joseph Mietus, Margret and H.A. Rey Laboratory for Nonlinear Dynamics in Medicine at Boston's Beth Israel Deaconess Medical Center.

Financial support was contributed by the Natural Sciences Engineering and Research Council, the Margret and H.A. Rey Laboratory for Nonlinear Dynamics in Medicine at Boston's Beth Israel Deaconess Medical Center and from the National Institutes of Health/National Center for Research Resources (Research Resource for Complex Physiologic Signals), NIH Grant no. 1P41RR13622-01A1.

1. Introduction

1.1 Purpose of the Study

This study explores the use of sound as a means of representing and examining data sets. Specific focus is given to applications in cardiology with examples intended to generate novel methods for displaying heart rate information that may be of potential clinical utility to physicians. Methods from music technology and computer music will be used to examine the representation of heart rate variability data with sound. The question explored will be whether clinically valuable information (which may not be evident with a conventional graphic representation) might become apparent through a sonic representation. The use of non-speech sound for purposes of conveying information is termed *auditory display*.

Auditory display represents a recent development in the intersection of multi-media technologies and scientific research. Just as the eyes and the ears play complementary roles in interactions with our environment, the complementary strengths of the two senses can play essential roles in data analysis. To date, graphical displays serve as the primary medium for presenting data. The 1980s brought tremendous increases in computing power, among them advanced visualization capabilities. Researchers building upon established graphing methods have employed these technologies. Over time, the various techniques have been combined, resulting in a vocabulary of commonly used images that are quickly understood (Kramer, *et. al.*, 1997). An example is the pie chart, which is a well-known illustration of proportional subdivisions. Pie charts are common vocabulary, appearing in specialized literature as well as in junior high school-level math textbooks.

In the 1990s, new and inexpensive computer technologies were developed that could generate and process digital audio content. Consumer-level personal computers are now capable of advanced sound signal processing in real time, leading a growing number of researchers to take up the question of utilizing sound to illustrate and distinguish relative elements of large data sets. Auditory display, however, lacks the recognized vocabulary of graphical displays. There is no auditory equivalent of the pie chart.

The development of auditory display technologies is an inherently multi-disciplinary activity. A successful auditory display must combine elements from perceptual psychology, music, acoustics and engineering (Kramer *et. al.*, 1997). Auditory displays, then, are best realized in an inter-disciplinary environment, with sound specialists who possess a working knowledge of the research area working alongside researchers who have a working knowledge of sound realization. A university music technology program is an environment that encourages such multi-disciplinary exchanges.

The work described herein explores various sound parameters and their suitability for conveying information in a way that permits meaningful discrimination. Through a succession of auditory models, a set of data operations is matched with a set of sonic parameters. As a result, new insights into the dynamics of the data sets are obtained, and general principles are discussed pertaining to the components of an optimal auditory display. It is hoped that the models presented here will reinforce the value of sound as an illustration medium and that the techniques will form a constructive step toward a standardized auditory display methodology.

1.2 Auditory Display

The idea of sound containing information is not new. Levarie and Levy (1980) point out that the trained ear can gain information through sound that is just as valid as information gained visually. For example, if asked to cut a string in half, most people would probably reach for a ruler. They point out that an alternative approach of measurement would be to find the dampening point of the string at which, when plucked, it sounds a perfect octave above its original frequency. Along the same lines, they report a humorous story published in the September 3, 1955 issue of *The New Yorker* about two violists who took an extended road trip in an automobile with a broken speedometer. When asked how they were able to maintain proper speed limits, one of them replied, "This DeSoto hums in B-flat at fifty. That's all we need to know." They point out that what is actually peculiar is that this story should be considered humorous. To the trained ear of a string player, such a measurement is as explicit as a distinguishing color on a road sign or a number read from a speedometer.

Current efforts towards advancing the use of sound to convey information have been largely due to the efforts of the International Community on Auditory Display (ICAD). This study will draw extensively from the precedents set by their work. Their publication *Auditory Display: Sonification, Audification and Auditory Interfaces,* a collection of papers taken from the first conference in 1992, defines the field and its objectives.

The distinction between the terms *sonification* and *audification* is defined in Gregory Kramer's introductory survey. He suggests that the term *audification* be used in reference to "direct playback of data samples," while the definition of *sonification* is taken from Carla Scaletti's paper to refer to "a mapping of numerically represented relations." This distinction, presented in 1994, is still in use in the ICAD literature, and will be employed in this study. The term *mapping* will appear throughout this study to refer to the translation of information to illustrative elements. While mapping of information to visual elements has an established canon of techniques in the field of visualization, auditory mapping is still in its formative stages.

1.3 Types of Auditory Display

As defined by Gregory Kramer, the objective of ICAD is to explore the uses and potential for conveying information through sound in technology. This broad definition encompasses a number of functions. One is the addition of sound elements to graphical user interfaces such as the Macintosh or Windows operating systems to enhance their functionality or ease of use. Another is implementations to make such user interfaces accessible to visually impaired users.

A number of real-time auditory monitoring implementations are in common use, such as the sonar and the Geiger counter. In medical settings personnel are well used to monitoring vital signs with sound-producing equipment. Relieved of having to keep their eyes on a visual monitor, medical workers can engage in other activities while still remaining aware of the conditions summarized by the auditory signals.

While the value of monitoring may be evident enough, the possibility of data analysis brings up new problems. The object of monitoring is to highlight known

conditions. All that is required is for steady states to be easily distinguishable from a set of known conditions that trigger some sort of alarm signal. It also, by definition, describes events as they are occurring, in real time. An analytical system, however, must have an added level of flexibility so that unknown conditions may be brought out. This flexibility is due to the fact that an analytical system does not exist in real time, but rather is something that is studied after the fact. The non-real-time nature of an analytical system introduces great flexibility in time resolution. Great volumes of data can be compressed to whatever playback time is desired. Varying levels of abstraction may emerge, depending on the degree of compression employed.

This study proposes an analytical model as a means of analyzing a complex data set. The specific data set explored represents heart rate variability.

1.4 Heart Rate Variability

The causes of fatal arrhythmias are central to cardiology. Heart rate fluctuations can be readily measured from an electrocardiogram and are thought to provide important insights into cardiac function. While clinicians may refer to healthy activity as "normal sinus rhythm," this term is merely a convenience (Peng, *et. al.*, 1993), since in reality healthy subjects often display more erratic patterns than unhealthy subjects do. For example, following a heart attack, patients whose heart rates are overly steady are prone to sudden, often fatal arrhythmia.

These heart rate fluctuations are referred to as *heart rate variability* (HRV), and are the result of three principal components. The heart's contractions are the result of electrochemical waves produced by the *sinus node*. The sinus node is the pacemaker of the heart and produces excitation waves spontaneously and very regularly, at roughly 70 cpm. The sinus frequency is modulated by the presence of chemicals secreted by the autonomic nervous system. The autonomic nervous system's components are twofold: *sympathetic* nerves secrete norepinephrine, which increases the heart rate, while the *parasympathetic* (or *vagal*) nerves secrete acetylcholine, which decreases the heart rate. Experiments to isolate the effects of each of these components have brought out interbeat intervals of 0.6s when the sympathetic and parasympathetic nervous impulses have been suppressed. Suppressing input from the sympathetic nerves can produce interbeat intervals up

to 1.5s. Suppressing input from the parasympathetic nerves produces interbeat intervals as low as 0.3s (Ivanov, *et. al.*, 1998). It is thought that nonlinear interactions between these two competing components are responsible for the heart rate's continual fluctuations, as well as external factors such as stress, or periods of exercise or rest. (Nonlinear interactions will be discussed in more detail in the next chapter).

To obtain HRV data, a medical technician attaches a series of electrical sensors to a patient's skin. A Holter monitor, a walkman-sized device that the patient can keep in a pocket or attach to a belt while engaging in normal activities, measures the voltage differences. The voltage differences recorded by the Holter monitor reflect cardiac activity. The voltage is sampled periodically just as an audio signal sampled for a CD recording. The result is a signal called an electrocardiogram.

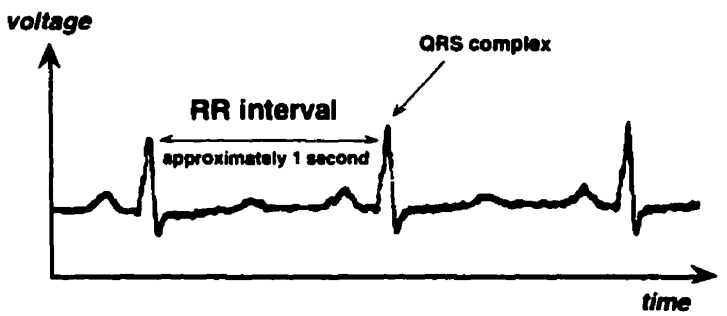


Figure 1_1: Electrocardiogram recording of heart activity

Of interest are the voltage spikes known as the *QRS complexes*. These electrical bursts are associated with the muscular contraction that is the heartbeat. The time interval between these bursts is known as the RR or NN (for normal-to-normal) interval. Following the recording, the samples of the continuous voltage signal are put through a beat recognition algorithm that timestamps each QRS complex. (These algorithms are generally proprietary, depending on the company manufacturing a given brand of Holter monitor). From these timestamps the NN intervals are saved as a one-dimensional data vector. The NN intervals are the data set used in heart rate variability analysis.

Such a series of discrete data points, measurements taken in time, is known as a *time series*. Operations on the time series are called *signal processing* (Kaplan and Glass, 1995). There are many types of time series. Computer musicians are

familiar with audio signals, such as those stored on compact discs, and operations performed with digital filters on the audio signal. An audio signal is an example of a *continuous* time series, in which measurements are taken periodically and the signal is reconstructed from these measurements. The degree of accuracy in the reconstruction is dependent on the sampling rate and the bit resolution.

A heart rate variability series represents an abstraction of the continuous data series. The heart rate variability time series consists solely of the NN intervals. Thus, the use of such a series restricts analysis to what can be determined by the time intervals between successive heartbeats, the heart rhythm. Such a series, which does not represent the complete contents of a continuous time series, but rather a select subset of points from it, is called a *point process* series. The NN interval series can be described as originating from a point process series.

There have been a large number of different statistical measures proposed to evaluate heart rate variability and there is not general agreement as to which are the most useful in explaining the erratic changes in heart rates, even those of subjects at rest. Many composers have explored applications of chaos theories to music composition and synthesis. Heart rhythms are also not new to musical contexts (Davids, 1995; Lombreglia, 1993). This project, however, takes a different focus. Rather than setting out to create musically interesting sounds, the approach is to explore whether these chaotic patterns can be a source of medically useful sounds. The question pursued here is: can cardiological diagnoses be aided by information taken from an auditory display?

1.5 Design of the Thesis

This introductory chapter has outlined the context of the work, providing essential concepts and terminology. Chapter 2 explores relevant background. Its categories include examples of musical compositions with data sets as their basis, work done to date in the field of auditory display, and current research in the field of heart rate variability. Chapter 3 introduces features of software sound synthesis and SuperCollider, the software programming language used to create the auditory display models. Chapter 4 reviews the steps that lead to a model auditory display program for heart rate variability. Once a general model is proposed, a listening perception test is carried out that compares auditory and visual displays of heart

rate variability data. Based on the results of the test, refinements are made to the general model to identify a particular heart condition. Chapter 5 provides a summary and conclusions. The appendices include a variety of background materials. Appendix 1 is a review of the physics of sound, pitch, timbre, volume, localization and phase. These topics have been the subject of exhaustive research; readers wishing for a more thorough study are directed to the references (Blauert, 1997; Bregman, 1990; Handel, 1989; Levarie and Levy, 1980; B.C.J. Moore, 1989; Pierce, 1983; Rossing, 1990; Helmholtz, 1885). Appendix 2 is an introduction to nonlinear dynamics, including the output of iterative equations, components of deterministic chaos, fractals and scaled noise. Appendix 3 is a brief summary of the Poisson Distribution to supplement the musical issues discussed in Chapter 2. Subsequent appendices include code examples of the sonification models and materials used in the listening perception tests. An accompanying CD contains audio tracks and a Macintosh-format CD-ROM portion that contains examples of the SuperCollider sonification models.

2. Survey of Related Literature

2.1 Data in Music

2.1.1 Data in Music—Making Art from Information

Interesting illustrations that bear on the topic of information and sound date back to some of the earliest written examples of classical Western science and philosophy. The most direct predecessors to the subject at hand can be found in the Twentieth Century, called by many "the scientific age," with the emergence of a scientific current among certain important composers.

We are indebted to the ancient Greeks for originating the idea of representing information through structured sound. The concept of a seven-tone diatonic scale derives from Greek cosmology, and until the Sixteenth Century cardiological diagnoses were conducted according to metrical patterns used by the Greeks in music and poetry.

In 6th Century BC Greece, Pythagoras used sound as the basis for illustrating cosmologically significant numbers. He derived a tuning system by experimenting with a monochord, a single-stringed instrument with a movable damper that allowed the string to be divided into two parts. While the ancient Greeks were not able to observe numbers of oscillations per second in a vibrating string, Pythagoras was able to codify aurally relationships between string length and pitch. His theory was based on two significant intervals: one with string lengths at a ratio of 2:1, which he called the *diapason*, and the other with lengths at a ratio of 3:2, which he called the *diapente*. Recognizing first the concept of tonal equivalence when a string length is either doubled or halved, Pythagoras derived successive diapentes. All intervals were normalized to fall within one diapason by multiplying ratios greater than 2 by 1/2, and ratios less than 1 by 2. The result was a diapason divided into seven steps, derived as follows:

```
1 x 2/3 = 2/3; 2/3 x 2 = 4/3
1 x 2 = 2
1 x 3/2 = 3/2
3/2 x 3/2 = 9/4; 9/4 x 1/2 = 9/8
9/8 x 3/2 = 27/16
27/16 x 3/2 = 81/32; 81/32 x 1/2 = 81/64
81/64 x 3/2 = 243/128

In ascending order:  1  9/8  81/64  4/3  3/2  27/16  243/128  2/1
```

Pythagorean tuning is thought by many historians (Wilkinson, 1988) to be based on the Greek perception of the number 3 as representing divine perfection. The scale, as shown above, is derived from the numbers 1, 2, and 3. Furthermore, all ratios of the scale are based on numbers that contain no prime factors greater than three. For the Greeks, music was part of an integrated cosmology that encompassed arithmetic, harmony, poetry and astronomy. This series of numbers was thought to represent physical and spiritual perfection (Grout and Palisca, 1988). Certain tones, as well as elements of Greek music theory, were thought to correspond to the motions of heavenly bodies. Thus, the Pythagorean tuning system was part of Plato's description in *The Republic* of the "music of the spheres."

Mastery of Greek music theory was considered an essential component of a physician's training (Cosman, 1978). The importance of music in perceiving patterns in the human pulse was an important element in the writings of Galen of Pergamum, the Third Century Greek physician whose voluminous output was the keystone of medical training until the Sixteenth Century. Galen's writings identify twenty-seven metric pulse varieties. The pulse of infants was described as having a trochaic meter, while a pulse of iambotrochaic meter described the pulse of elderly patients. Specialized pulses were thought to correlate with a variety of medical conditions. By the medieval times, pulse was just one concept of time that had far-reaching implications for physicians, whose diagnoses were based on the time of the patient's birth, time of injury and time of treatment, all of which were correlated with the motions of the stars and moon.

The early 1900s brought a number of scientific breakthroughs such as relativity and quantum physics. These concepts became an important inspirational focus in the music of Edgard Varèse. Anderson (1984) argues that scientific principles are essential to any serious analysis of Varèse's work. Rather than relying on classical ideas of harmony, his music seems to consist of juxtapositions of sonic events and their interactions. Varèse described his music as being composed of "unrelated sound masses," distinguished by timbre. Anderson speculates that his inspiration came from quantum theory, the discovery of x-rays and radiation.

In his interviews and lectures, Varèse frequently equated the act of composition with that of scientific research, as indicated by his titles (*Ionisation*, *Density 21.5*,

Intégrales, etc.). However, his claims of music as science are subjective at best since his pieces are not meant to reveal quantitative data about the natural world. Rather, scientific analogies for Varèse are perhaps comparable to the exoticism practiced by some of the previous generation of composers. The Oriental elements incorporated in the compositions of Rimsky-Korsakov and Ravel were not a serious exploration into ethnomusicology. Similarly, Varèse's references to science contain no more information about physics than Vivaldi's *The Four Seasons* contains information about climatology.

Iannis Xenakis, an admirer of Varèse, made his mark as a composer by using calculations as primary musical material. This innovation was the result of two factors. One was his background. His formal education was in engineering, coupled with a more than passing interest in the Greek classics. Though born in Romania, Xenakis was raised in Greece, which he considered to be his country. He was highly influenced its philosophical heritage of attempting to find order in the universe, an interest that seems to have helped him come to terms with his violent experiences as a political activist in Greece during World War II. The second factor was the musical context of the time. Xenakis attempted to fuse his range of experiences into musical expression at a time when the European musical community was increasingly preoccupied with serial composition.

Serialism can be traced to a set of compositional strategies conceived by Arnold Schöenberg beginning in 1908. As a reaction to the increasing chromaticism in musical works of the late Romantic era, Schöenberg began writing pieces that were not based on a tonal center, and were thus termed *atonal*. By 1923, Schöenberg had developed a system of *twelve-tone* or *dodecaphonic* principles that treated all twelve notes of the octave with equal rank. His system relied on a *row*, a sequential ordering of the twelve pitch classes. No note was to be repeated until all the others had sounded, although this stipulation has been treated with more stringency by subsequent theorists than it ever was by Schöenberg himself. A piece was based on operations performed on the row, chiefly *transposition* (maintaining all intervals between pitches, but starting with a different pitch class), *inversion* (reversing the direction of all intervals within the row), *retrograde* (reversing the order of pitches in the row), and *retrograde inversion* (an inversion played in reverse order).

The *Ferienkurse fuer Neue Musik*, which began at Darmstadt in 1946, featured a group of young composers who looked to first to Schöenberg, and later to his student Anton Webern, as the originator of music's next evolutionary step. Pierre Boulez observed in Webern's music extensions of row operations to sequential ordering of other musical elements, such as note duration. *Total serialism* was characterized to a large extent by a high degree of determinism in a composition, achieved through extending twelve-tone pitch techniques to other musical parameters such as rhythm, dynamics, articulation or instrumentation. Precise control over musical elements was achieved by writing material that occurred in a predetermined sequence according to its place in a row. The preoccupation with total serialism was exemplified by Boulez's statement "I, in turn, assert that any musician who has not experienced—I do not say understood, but in all exactness, experienced—the necessity for serialism is *useless*."

In a 1956 article in *Gravesaner Blätter* (Xenakis, 1956), Xenakis asserted that, so to speak, the emperor was wearing no clothes. As a case for the ultimate futility of serial music, he wrote that its coherence was based on permutations of 12-tone matrices that no one could actually hear. The result was not the supreme order and clarity claimed by serialism's proponents, but rather an incoherent mass of sound. The linearity of the rows was lost with the intersecting lines of activity. The structure was evident only when the work was perceived as a whole, an impossibility since music exists in time and only a fraction of a whole work is audible at any given instant. He reasoned that since listeners were presented with a mass of sound based on these rather trivial arithmetic operations, it would be in composers' interests to acknowledge the nature of a sound mass, and manipulate it with more sophisticated mathematical equations found in nature.

Xenakis' first three works are of particular interest as they are partially sonifications of non-musical information. The first contains formal divisions according to classical proportion and musical representations of an architectural design. The second contains a sonification of Brownian motion. The third contains an implementation of probability.

In the 1950s, as a structural engineer and architect at the firm of Le Corbusier in Paris, Xenakis took interest in Le Corbusier's implementations of the Golden Mean, a proportion found throughout nature and classical Greek architecture. The

Golden Mean involves forming elements according to the ratio B:A = A+B:B, as shown below:

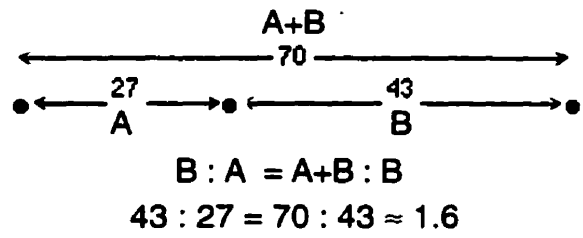


Figure 2_1: Golden Mean proportions

The Fibonacci series is a related number series:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233 . . .

Leonardo Fibonacci was one of the first great mathematicians in European culture. He derived the sequence above in the early Thirteenth Century as an analysis of optimal reproduction rates among rabbits (Gillipsie, 1970-90). It was subsequently demonstrated that the asymptotic ratio between successive numbers in the series was equivalent to the Golden Ratio shown in Figure 2_1, an irrational number represented by:

$$\frac{1+\sqrt{5}}{2} \approx 1.618\ldots$$

Many composers have employed the Golden Mean and Fibonacci series. Webster (1950) cites formal divisions that approach Golden Mean proportions in composers from the Classical period to the Twentieth Century, including Haydn, Mozart, Beethoven, Schumann, Chopin, Debussy, Schöenberg and Bartok. It is not clear, however, whether this proportion was applied consciously, or whether it was employed intuitively, as a division point lying between one half and two-thirds of a given length. Kramer (1973) cites several Twentieth Century composers who employed Fibonacci numbers in their work, including Bartok, Stockhausen and Nono. Bartok, in particular, extended their use beyond formal divisions to derive scales with interval contents taken from the Fibonacci series and in the lengths of repetitions of certain themes.

In the early 1950s Xenakis also began to study composition in his off-hours. While Xenakis did not possess a great deal of formal musical training, his teacher, Olivier Messiaen (who had also taught Boulez and Stockhausen), fostered his interest in applying architectural principles into a compositional methodology. Xenakis began attending the workshops at Darmstadt, established himself as a maverick with his article in *Gravesaner Blätter*, and began his own explorations.

Xenakis' approach was to treat music as a field of sound in which material could be plotted as a series of vectors over multi-dimensional axes of dynamic, frequency, intensity, duration, etc. (Matossian, 1986). His first major work, *Metastaseis* (1953-54), was written entirely *divisi* for 61 players—46 strings, 7 brass, 6 winds and 2 percussion. With its sixty-one independent parts, the piece was his first experiment in what he termed the *sound cloud*. He used the Golden Mean for formal sub-divisions, pitch, articulation, duration and dynamics. He also adopted an architectural fad of the time, the hyperbolic paraboloid, which he later used in the design of the Philips Pavilion for the World's Fair of 1956. Attracted to the creation of curved shapes created by component straight lines, Xenakis wrote the climax of *Metastaseis* based on this shape. Each straight line represented a glissando trajectory of one string instrument. The starting and ending height were represented by pitch, the horizontal point of origin by time of entry.

Xenakis applied this system of proportion to a number of architectural projects during this time, which culminated in a chapter included in Le Corbusier's *Modulor II* (1958). In this chapter, Xenakis recalled Goëthe's description of "architecture [as] music become stone," and inverted it to "music is architecture in movement."

After *Metastaseis*, Xenakis began to incorporate other types of sonification into his work that were not based in architecture. His second piece, *Pithoprakta* (1957), took the sound cloud/string glissandi concept a step further. The glissandi are not homogeneous, but in various directions and speeds. As with the "architectural" section of *Metastaseis*, it is most helpful to view Xenakis' graph of the relevant section. Matossian (1986) provides many illustrations meant to illuminate the underlying principles of Xenakis' work. In *Pithoprakta*, the

trajectories were a mapping of a Lévy distribution, simulating the Brownian ricochets of gas molecules, as described in Appendix 2.

Xenakis' third work, *Achorripsis* (1958) was a musical examination of probability. A matrix of activity determines interaction among timbral elements over time. The number of events from each instrument group per time unit is distributed according to the Poisson distribution. Due largely to his use of it, the Poisson distribution is now a common probability formula employed in algorithmic composition. While algorithms for the Poisson distribution appear in many sources, its history and what exactly it illustrates are not as commonly described. A brief summary of the Poisson Distribution is provided in Appendix 3. The wide range of its applications makes it clear why it would be attractive for a composer such as Xenakis, who was seeking ways to reflect universal laws in music.

In a series of articles, which eventually culminated in the publication of his book *Formalized Music* in 1971, Xenakis articulated his theories of what he termed *stochastic music*. The term derived from the Greek *stochos*, which he defines as an equilibrium state that is eventually reached after a very large number of particles are taken through a very large number of interactions that contain some element of randomness. Examples from nature might include the sound of rainfall or a swarm of insects. In each of these cases, listeners do not perceive the activity of any one individual particle, but instead perceive a macroscopic sound mass, or gestalt, that is the sum of all the micro-level interactions. He describes music as an organization of operations of logic and relations on sound (and, by implication, time).

2.1.2 Biofeedback Music—Medical Monitoring as Performance Art

Among the aesthetic explorations of the 1960s were inquiries into the nature of a performance event. New elements of spontaneity were sought in events termed "happenings," in which an artist assembled an environment of some kind, and the audience's interactions with it became the performance. This spirit of "anything goes, everything is art" caused some performers to look literally inward, offering sonic monitors of their physical vital signs as performance material. Alvin Lucier's 1965 performance piece *Music for Solo Performer* involves a performer sitting silently on-stage, wired to a set of electrodes and an EEG machine. The

low frequency alpha brain waves are amplified, and their vibrations cause percussion instruments placed near the speakers to resonate. This piece was performed in the Fall of 1999 at McGill University, made possible by the donation of an older, out of service EEG machine from the Montreal Neurological Institute. A grounding electrode was placed on one of the performer's ears, and four others were placed on his forehead, temple, top and rear of the skull. The EEG measured and amplified the potentials between pairs of these electrodes. The alternating current was at frequencies in the range of 5-15 Hz. The four frequency channels were fed to four channels of a mixing console, from which they were distributed to a four-channel amplifier. Each channel fed a speaker near a percussion instrument. The performer, Andrew Brouse, reported that the goal was to reach a "meditative, non-visual state" in which the alpha brain frequencies became active. The trick was not to focus the attention, but to reach a semi-conscious state. The piece ends when the performer opens his eyes, dropping the alpha waves to low levels.

Benjamin Knapp's Biomuse (1990) is a MIDI adaptation of this idea. Bands are placed around a performer's wrist, knees and head. The bands track neuroelectric (brain and eye potentials) and myoelectric (muscle potential) signals and send them to a DSP processor that converts their values to MIDI information.

Since the 1960s, David Rosenboom has produced a number of biofeedback pieces in which signals from the performer's brain waves controlled structural events in the music. In his piece *On Being Invisible* (1977, 1995), the computer's role is threefold. The piece begins with the computer generating musical material based on pre-programmed algorithms. As it generates the material, a listening process analyzes the output, searching for events that would likely be perceived as structurally significant. At the same time, it is monitoring the EEG output of a performer with the aim of extracting Event-Related-Potentials (ERPs) from the ongoing brain wave activity. ERPs are more embedded transient waves that are related to recognition of beginnings of events. If the computer listener finds that new musical events correspond with the performer's ERPs, it generates a new type of pattern that is based on its analysis of previously generated patterns and is meant as a logical continuation of them.

2.1.3 Nonlinear Dynamics in Music

With the publication of Mandelbrot's *The Fractal Geometry of Nature* in 1983, and the popularization of terms such as "self-similarity," "chaotic dynamics" and "strange attractors," visual art based on the output of iterative functions became a standard item in poster shops. Besides the abstract beauty held in these images, chaos theory's appeal to the public imagination was due in part to hyperbolic claims such as "a butterfly flapping its wings in Beijing can cause a rainstorm in Montreal five days later" (Kaplan and Glass, 1995).

The interest of the computer music community was similarly sparked, as musicians adapted the instigations of Xenakis to create music by mapping the output of fractal and chaotic equations. The following survey, while not meant to be exhaustive, details many of the ways that nonlinear dynamics have been applied to musical composition. An introduction to fundamentals of nonlinear dynamics is provided in Appendix 2.

2.1.3.1 Fractal Music

Statistical self-similarity seems to have supplemented the Poisson Distribution as a ubiquitous principle that has been found to underlie the nature of many phenomena. Just as Xenakis was drawn to the Poisson Distribution to reflect naturalistic distributions of musical events, fractal dynamics have been the basis of a number of musical investigations.

Voss and Clarke, studying extended radio broadcasts, found that loudness fluctuations in music displayed a $1/f$ distribution below 1 Hz (Voss and Clarke, 1975; Voss and Clarke, 1978). Voss and Clark then expanded their study to the creation of music by self-similar principles. Gardner (1978) describes the algorithms created by Voss and Clark for generating a series of numbers that follows the statistical properties of scaled noise. These algorithms are summarized in Appendix 2. Once generated, the numbers can be mapped to pitch, duration, or any musical parameter. Voss and Clarke conducted numerous experiments in which melodies with pitch and duration generated by each of these methods were played for listeners who were asked to evaluate them. It is perhaps not surprising that listeners found "white" melodies to sound consistently random and "Brown" melodies to sound consistently monotonous. "Pink" melodies, on

the other hand, sounded "about right" in terms of consistency and change. The conclusiveness of these studies is limited in that only short segments were played for the listeners. The results say much more about the nature of these scaled noises than they do about music itself. It would be a mistake to assume the converse, that great music can be generated by a simple $1/f$ algorithm. Still, it is intriguing that a series of $1/f$ distributed numbers can produce melodies that appear to have some planned intention behind them.

Bolognesi (1983) extends the $1/f$ number generation algorithm described by Gardner in two ways. One way is to add a random element to the number of die cast with each iteration, by use of a probability distribution that maintains the average scope of any given die's value. This variation in the number of die cast at each step serves to disrupt the strict binary hierarchy of the running total that results from Voss and Clark's algorithm. Bolognesi then goes a step further by "weighting" the dice, with the result that there are tendencies towards certain pitches. The melodies produced by these modifications have a clustered character. The changing pitch centers are determined by the values generated from the dice corresponding to the most significant bits of the incremented binary number.

Bolognesi then describes the generation of self-similar material via Lévy walks (or "random walks," as described in Appendix 2). The size of each step is determined by a probability function introduced by Mandelbrot (1983) as a model to describe the clustering of galaxies. The result is a "melodic clustering" of changing pitch centers, but with a more continuous scale than the discrete scale that resulted from the dice algorithm. Using the Lévy walk over multiple axes allows each axis to represent different musical elements, as the steps and the multiple axes then become similar in function to the vector system employed by Xenakis. Generating more than one musical line produces similarities in the rate of change in pitch centers among the multiple melodies.

Dodge (1988) takes a different approach to fractal methodology, describing the creation of self-similar values via list processing operations. A list of pitch classes is created. A member of the list is selected at random and copied into a melodic line. Random numbers are generated, and serve as indices to the list of pitches. Pitches are added to the melody until all the pitches from the list have

been selected. Thus, the melody will likely contain a high number of repeated notes. For the second line, a second list of accompanying pitches is created, with shorter durations, for each note of the first line. In the same fashion, a third line is created. To derive durations, Dodge then worked backwards. Using the same generation technique, a duration value was found for each pitch in the third line. The durations of each pitch in the second line were then determined by simply summing the durations of the notes in the third line that corresponded to each of the second line's pitches. The durations of the first line were then taken as the sum of corresponding notes from the second line.

Gary Lee Nelson uses a fractal image as the basis for generating microtonal pitches in "Fractal Mountains" (Rowe, 1996; Nelson also describes this piece on his web page: *http://www.timara.oberlin.edu/people/~gnelson/gnelson.htm*). An interactive piece for Nelson's MIDIhorn instrument, his fractal algorithm tracks the onset time of successive notes and their interval difference. Treating each time and interval as an (x,y) pair, the program then interpolates pitches in 96-tone equal temperament that subdivide the resulting line. (Appendix 1 contains a description of equal temperament).

The work of Bolognesi, Dodge and Nelson bears conceptual parallels with the music of Varèse. They are not nonlinear dynamics specialists, yet they take a keen interest in adapting scientific elements for their compositions. While they do not use fractal principles to explore data in a quantitative manner, the self-similar algorithms they employ provide new means for generating material. Thus, these algorithms might be seen as providing an element of exoticism to their work similar to the adaptations of physics created by Varèse.

Mapping Chaotic (and other) Data

Other musical investigations have focused on the output of iterative equations. Pressing (1987) describes sonifications of the logistic difference equation described in Appendix 2:

$$x_{t+1} = Rx_t(1 - x_t)$$

Figure A2_4 in Appendix 2 shows the bifurcation diagram that describes the asymptotic output of the equation depending on the value chosen for R. Pressing

used the Csound synthesis language, constructing a Karplus-Strong plucked string algorithm (Karplus and Strong, 1983; Jaffe and Smith, 1983) to map the iterated values to pitch. His mapping formula was 2^{cx+d}, where 2^d was the frequency of the lowest note, c was the octave range, and x was the value of the data point. Setting d to 6 and c to 3, he established a three-octave range from C at 64 Hz (two octaves below middle C) to C 512 Hz (an octave above middle C).

Choosing an initial value of x at 0.5, he worked with values of R above 3.6, which fall in the grey areas of the diagram, just before the onset of a new cycle. He described these regions as "quasi-periodic" (although this is not a correct use of the term) according to his observations. For example, setting R to 3.828, a cycle of 3 emerged following a transient period of 150 iterations. After continued iterations, Pressing noticed that the cycle length would shift to different lengths, in the range of 2-7. He identified the start of each cycle when a frequency over 400 Hz was produced. It was an easy delineation, as subsequent pitches fell well below this value. He found that cycles of n pitches all had similar contours. This was a feature not found in any mathematical descriptions, yet clearly audible with his mapping of values to frequencies.

Bidlack (1992) describes four chaotic equations. Two are iterated maps that are notated by difference equations, as described in Appendix 2. His third and fourth equations are continuous maps in three dimensions, which are notated with differential equations. These last two require integration, which Bidlack employs with the Euler method (Kaplan and Glass, 1995). More a tutorial than a musical analysis, Bidlack's article is a straightforward introduction to nonlinear dynamics complete with accompanying C code to demonstrate the translation of each of the equations into computer algorithms. Bidlack suggests pitch as a mapping of one variable, leaving it to the reader to imagine musical parameters that might be modulated by mappings from other variables of the equations.

Harley (1994a) provides a general discussion on the question of creating effective data mappings of the output of iterative equations. Two issues of concern are resolution and listener comprehension. The first is an issue shared by scientists. The output of a chaotic function is highly dependent on how many decimal places their values are rounded to. For a visual artist, the screen resolution can, in the same way, change the appearance of the function's visual plot. For a musician,

this problem translates into working with an appropriate averaging of the data values. The second question is more overriding, addressing the translatability of functions that produce effective spatial (visual) representations into the time-based (aural) medium of music. Composers interested in implementing iterative equations face the same problem observed by Xenakis pertaining to serial music. It is far from certain that music generated by iterative equations has the same power as visual representations produced by these equations. The totality of the function cannot be perceived in music, only a moment-to-moment iteration of data points. Unlike the viewer of the visual output, the listener's appreciation of the aural output is limited by the amount that can be remembered effectively.

The question of resolution was the creative basis of another work by Charles Dodge, described in (Dodge and Jerse, 1995). In *Earth's Magnetic Field* (1970), Dodge sonified measurements of the sun's radiation onto the magnetic field that surrounds the earth. He took averages of the radiation over three-hour periods taken from twelve measuring stations placed throughout the world, resulting in a total of 2,920 readings for a year's worth of data (he worked with the year 1961). Twenty-eight possible values were mapped to diatonic Meantone pitches[1]. The piece's section breaks were taken from the 21 "sudden commencement" points of sudden increase in value. These section breaks were grouped into five movements. In three of them, the sudden commencements were mapped to tempo change. The length of each commencement section was plotted on a horizontal axis, with the highest value in each section plotted on the vertical axis. The resulting function described continuous tempo changes within these movements. In the other two sections, the tempo was constant, with one note sounding for a one second duration whenever there were two identical readings in succession, with the next second containing all pitches corresponding to readings between the next two identical successive readings.

The question of resolution is termed *binning* in Ary Goldberger's description of Zach Davids' piano album *Heartsongs: Musical Mappings of the Heartbeat* (Goldberger, *et. al.*, 1995; Goldberger, 1995). Binning breaks a data set into

---

[1]Meantone temperament was an attempt to resolve the disparities between Pythagorean and Just tunings. It involved flattening the primary fifths of the scale, so that some degree of transposition was possible. It was used in some Baroque pieces prior to the universal adoption of equal temperament (Wilkinson, 1988).

coarser subsets, with a bin containing all data points within a given range of possible values. Davids' recording is a mapping of heart rate variability data. A data set of approximately 100,000 points is averaged over every 300 beats, leaving approximately 330 values. The range covered by these points (the highest value minus the lowest value) is then divided into 18 equally spaced bins. Having thus collapsed the data set into 18 values, each value is then assigned to a musical pitch, creating a melody of 330 notes. Davids then composed harmonies and rhythms to underlie this melody.

The examples of Davids and Dodge raise the issue of freely composed material vs. generated material. Since the melodies in each of these pieces were a matter of the composer's taste, the reliance of the music on the data is qualitative rather than informative. The experience of hearing these pieces may be equally effective if the same melodies are heard over freely composed harmonies.

The problem of effectively translating imagery to music was undertaken by Gary Kendall in *Five Leaf Rose* (1981). His solution was to base his composition on a simple and periodic image, the polar plot $r = |\sin 2.5\theta|$. The shape of this plot is a five-leaf curve, with each leaf moving down from multiples of seventy-two degrees: 72° to 0°, 144° to 72°, 216° to 144°, etc.

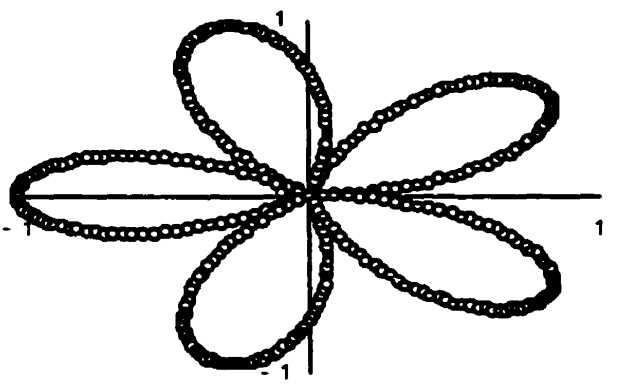


Figure 2_2 Five-leaf rose plot of $r = |\sin(2.5\theta)|$

The progression of the piece takes the listener through 360°, with a changing accompaniment pattern throughout. The plot is divided into points every 2°, with each point mapped to a particular pitch and timbre. The changes occur gradually over 360°. Given the "jumps" that occur at the starting point of each leaf, the

changes divide the piece into five sections. The traversal of the curve is at a constant velocity of radians per time unit. This means that more space is covered per time unit in areas of the curve farthest from the origin, as can be seen from examination of the above figure. The result is that there is a denser series of notes in the farther than in the nearer regions. There are always six notes that sound, some ahead of the present position, some behind. The timbre changes over the course of the plot via a three-operator cascade FM algorithm (FM synthesis is discussed in the description of the HRV Sonification). While the frequency of one modulator is fixed, the other goes through a series from one to twelve. The result is a series of harmonics that correspond to five pitch classes. Each of these five pitches is used as the fundamental of another equivalent harmonic series. Each leaf has two such harmonic series sounding simultaneously. Thus, larger-scale elements of the piece are based on the curve, as are the moment-to-moment elements. Creative elements are added to the strict mapping of curve elements by modulating entry times of the accompanying notes, and mapping other elements such as detune and low-pass filtering to the positional angle as well, with all such changes out of synchronization with each other for maximum variety. Over this accompaniment the melody was freely composed.

A less literal form of mapping is described by Harley (1994b) with the aid of his CHAOTICS software. Performing iterations of the logistic difference equation, the output can be re-scaled to whatever range is desired and the values mapped to pitch or any other parameter. Such a direct function, however, is not the primary purpose of the software. A statistical module creates a histogram that keeps track of how many times each value has been generated. Musical parameters can then be created from these values based on the number of occurrences of the output value. For example, pitches could be assigned according to diatonic function. When a value is generated that matches the most frequently generated value, the tonic tone may be output; when the value generated matches the second most frequently generated value, the dominant tone may be output. Thus, the output of the equation is mapped to musical functions that are chosen by the composer. Harley stresses that the software was not created to represent sonic mathematical analyses, such as those described earlier, created by Pressing or Bidlack. Rather, the CHAOTICS modules are meant to provide a level of musical cohesion by that composers may choose structural elements that maintain a balance according to values output from the chaotic equation.

2.1.4 Concluding Thoughts on Data as Music

The examples discussed in this section provide orientation into the kinds of sound-making methodologies that must be at the heart of an effective auditory display. The use of nonlinear functions as a musical basis is a compelling and potentially fruitful compositional tool representing a blending of scientific and music theories. In order for it to succeed, however, those who choose to engage in it must appreciate its multi-disciplinary nature. Articles on chaos theory and music are often written by authors who specialize in one field but who have limited understanding of the other. As cited above, Pressing misuses the term "quasi-periodic" in his description of the output of the logistic difference equation. By the same token, Harley, while his software modules provide interesting musical possibilities, misuses terms such as "aurocorrelational," "deterministic" and "chaotic" to the extent that discussions with nonlinear dynamics specialists would be limited at best.

Similarly, an article by mathematician Diana Dabby (1996) associates the output of a chaotic equation to successive pitches of a Bach Prelude. Changing the initial conditions of the equation produces different output. Using the same output-to-pitch mapping, the new equation's output is associated with melodic sequences that are similar to the original melodies composed by Bach, but which contain substitute pitches taken from elsewhere in the composition. Different forms of the equation produce different versions of the Prelude, some of which are similar to the original material and some of which are very different due to greater degrees of pitch substitution. Besides the misuse of terms such as "appoggiatura" and "contrapuntal," Dabby terms her substitute pitches as "variations" on Bach's work without acknowledging that musical variations based on good compositional craft involve more than pitch rearrangement. As a result, it is not clear from her article whether there is any inherent relationship between the Prelude and the Lorenz equation that she uses beyond that of an artificial superimposition.

Similarly, Gogins (1991) describes a system of iterative functions that is meant to produce fractal computer graphics. Each successive value is fed into a different function, chosen at random or in sequence. The focus of this article is primarily on graphic output, although MIDI files are created from these visualizations. However, it is not clear how well MIDI's resolution of 0-127 represents the gradations of these fractal images. Gogins goes on to describe briefly Julia sets as

variants of the Mandelbrot set, derived by changing the constant in the Mandelbrot equation in order to highlight one area of the Mandelbrot set. Thus, the Mandelbrot set is termed a "one volume dictionary" of all Julia sets. It seems a stretch, however, when Gogins follows this line of thought with the concept of a Mandelbrot set for music that would be a "one volume dictionary of all possible musical scores."

It might be argued that the above citings of misused terminology amount to little more than semantic nit picking. After all, if these authors are able to produce material with their respective algorithms, why does it matter if their terminology lacks precision? The answer lies in the fact that composition of this nature is still a recent development. The question of effective representation of chaotic dynamics in music remains largely unsolved. A straightforward mapping of output from a chaotic equation may be too fundamental to create compelling musical representations. The approach of Kendall, in which the time progression follows a specific trajectory through the visualized data set, is an interesting possibility. However, the choice of trajectory would be a difficult matter with a more complex image. Returning to the question raised by Harley as to the manner in which chaos and music may relate to each other most effectively, it seems that a general methodology would have to take into account the abilities of both the eye and the ear in perceiving chaotic dynamics. The eye is able to perceive all iterations simultaneously on a visual graph, while the ear's perception is time-based and subject to the constraints of short-term memory. Therefore, trying to map the complexity of these visual images to an aural representation may be taking the wrong approach.

The next section will take up the subject of what the particular strengths of the auditory system are. A feature of the auditory system that will be expanded in the next section has to do with the ear's strength in following simultaneous streams of information. An effective musical representation of chaos is likely to be one that seeks to extract as many dimensions as possible from the generated data set. An intertwining of these parameters may be an effective musical substitute for the visual nature of all iterations being present simultaneously.

The solutions explored in auditory displays with sound parameters assigned to multi-dimensional data sets may provide practical solutions for researchers as well

as composers. However, for such displays to be created it is important that sound and data specialists be able to work together with a solid grasp of the concepts central to each field. It is hypothesized that the sonification models presented in this thesis will be useful for successful future work in research as well as for music creation.

2.2 Auditory Display

As stated in the Introduction, a primary source for auditory display work is Kramer (1994). In comparing functional elements of auditory and visual displays, another essential primary source is Bregman (1990), who explores perceptual principles of audition. The grouping of auditory elements and their perceptual assignment to an object or event creates an *auditory stream.*

2.2.1 Elements of Auditory and Visual Displays

Many elements of visual displays have intuitive correlates in the sonic domain. Height often means "more," a greater magnitude of some kind. A natural sonic correlate is pitch, such that a higher pitch can signify greater magnitude. The use of pitch involves *relative* changes. Only the rare individual who possesses perfect pitch would be able to identify the numerical value of a sounding frequency. However, fluctuations in pitch are adequate to indicate relative changes in value. The human ear is highly sensitive to changes in frequency, such that even small changes are perceivable as differences in pitch. Another possible magnitude correlate is volume, although this parameter is problematic due to the difficulty in assigning definite loudness scales, described in Appendix 1. Given the ambiguity of loudness as a percept, this is a mapping that is likely to be most effective in measuring changes on a large scale, perhaps in tandem with other parameters. It is not likely to be effective in conveying small-scale changes in magnitude. Frequency, then, would be the preferred method to convey magnitude, although due consideration must be given to the size of the changes involved. Due to the logarithmic nature of the auditory system's pitch perception, also described in Appendix 1, changes in the lower frequency ranges of only a few cycles per second can produce differences on the order of a number of musical scale steps, while much larger changes in frequency are required to produce the same relative pitch change in higher ranges. Hence, it is often preferable to map changes of frequency on a logarithmic, rather than linear, scale.

Color, or brightness, is often an important component in a visual display to differentiate between different types of elements. A literal mapping of color to the auditory domain might involve pitch, since both color and pitch are related to frequency. But if pitch is best employed to represent changes in magnitude, then perhaps a more suitable correlate for color is timbre. Musicians often informally refer to timbral characteristics as color, with comments such as "This piece brings interesting colors out of the piano." With the advent of computer music synthesis, many studies (Moorer and Grey, 1977; Gordon and Grey, 1978; Wessel, 1979) have added quantitative classifications of timbre based on overtone content and attack time. It is far from certain, however, that small changes in these parameters could be an effective basis for an auditory measurement. Like loudness, timbre is probably best employed to reflect large-scale changes, or as an enhancement in combination with other parameters. The cardiology model presented later will provide an example of a suggested use of timbre.

Another possible component is that of location. As described in Appendix 1, Blauert (1997) concludes that the eye displays greater precision in discerning changes in location than does the ear. Localization, however, is not a simple cue. Bregman (1990) observes that localization alone is not sufficient as a means to discriminate independent auditory streams. In life it is rare that we hear only a direct sound source; enclosed spaces, surfaces and obstacles all create a multitude of reflections. Thus, all identification of objects through hearing would break down if each reflection were indicative of a new auditory event. However, we get a great deal of information from the timbral changes introduced by these multiple reflections. The superimposition of the sound wave with copies of itself creates reinforcements or cancellations of certain frequency regions, an effect known as *comb filtering*. The pinnae (outer ear) also carry out comb filtering to assist in identifying vertical placement of sound objects. As is the case with small frequency differences, the auditory system is highly sensitive to small differences of inter-onset time. This sensitivity is used to assess acoustic environments. Reverberation filters sound, depending on the size, shape and material contents of a room. It would appear that the evolutionary process has been carefully selective in how our perception of location has developed. As discussed in Appendix 1, differences in phases of a complex tone do not change the tone's primary characteristics: if the tone is steady, introducing phase differences will have at best a minimally audible effect. However, phase differences experienced as inter-

onset times of sound events, either among overtone components during the attack portion of a sound event, or as reflections of a sound as a component of reverberation, give information about the listening environment.

Localization can also be highly effective when used in conjunction with other parameters. Two tones close to each other in frequency may be indistinguishable if heard over headphones, balanced equally in each channel of a stereo playback system. Simulating spatial separation via interaural intensity difference, however, can cause the two tones to segregate and be perceived as two separate pitches (Cutting, 1976). Early papers on multi-channel recording noted that the effect of adding channels was not so much the ability of the listener to perceive precise apparent locations of instruments, but rather a more qualitative impression of *spaciousness* (Steinberg and Snow, 1934). While listening to music through one speaker, the impression was that of hearing through a window the size of the speaker; listening to music through two speakers gave the impression of an elongated window that filled the space between the two speakers. Bregman confirms this experience anecdotally by reporting his experiments of switching a sound system back and forth from stereo to mono, or covering one ear in a concert hall. He noted an increased level of segregation among the various instruments, a factor that audio engineers call *transparency*.

This auditory distinctiveness among sound sources suggests a tenet that will recur throughout this work: *the auditory system is particularly well suited for following simultaneous streams of information*. This strength related to the attentional filtering the auditory system is able to carry out, commonly known as the "cocktail party effect" (Handel, 1989). With multiple sound sources, we have the ability to selectively prioritize a single source. By the same token, unchanging sounds tend to recede into the attentional background. In clinical environments such as hospitals, patients may be monitored by a variety of sound-producing devices. The consistent output of these devices keeps any one of them from being prominent until a particular vital sign crosses a critical threshold, causing its associated monitor to emit an alarm noise and "pop out" of the sound field.

2.2.2 Background Work in Auditory Display

In his introduction to *Auditory Display: Sonification, Audification and Auditory Interfaces*, Gregory Kramer provides a historical survey that ascribes the first

consolidated exploration of sonification to Sara Bly's 1982 dissertation from University of California Davis (unpublished). Her most definitive conclusions involved multi-variate data parameters. Participants were asked to identify different flower species based on four measurements per plant, represented with sound parameters such as pitch, loudness and attack time. She found a high degree of accuracy in participants' identifications.

Bly performed further experiments comparing illustrations involving sound only, graphics only, and both. Asking participants to identify test samples as belonging to one of two differentiated sets, she found that both the auditory and the visual displays to be equally effective, with the bimodal display yielding the highest degree of accuracy. Steven Frysinger also took up the issue of graphic, auditory and bimodal perception, reported in *Journal of the American Statistical Association* (Mezrich, *et. al.*, 1984), his 1988 master's thesis from the Stevens Institute of Technology in Hoboken (unpublished), and in the 1990 SPIE Proceedings (out of print). Participants in his tests were asked to identify patterns in single-dimensioned data sets after a period of training. His results showed the same degree of accuracy with a bimodal format as with an auditory-only format. Kramer's conclusion is that auditory displays are thus valuable on their own merit, not only as adjuncts to visual displays.

Kramer cites a 1982 study for the London Civil Aviation Authority by R.D. Patterson as an important step in developing a standardized sonic vocabulary. It corroborates many factors found independently by others. Patterson's study reported on the effectiveness of warning systems on commercial aircraft. He defined three priority levels: *emergency*, *abnormal* and *advisory*. Recommendations were made for alarms in these categories in terms of sound level, temporal characteristics, spectral characteristics, and ergonomics. Signal levels at 10-15 dB above the cockpit noise threshold were found to be optimal, being loud enough to ensure notice, but not so loud as to interfere with pilots' verbal communications. Attack times of 20-30 ms were found optimal, with shorter attack times tending to be overly startling in their abruptness. Alarms consisted of on-off temporal patterns. Patterson found that sounds with similar changes in volume over time tended to be confused, even if their spectral content differed greatly. This observation correlates with that of Chowning (1974), noted in Appendix 1 in the discussion of a synthesized tone's envelope shape. Faster

pulses resulted in a greater sense of urgency. The optimal frequency range was found to be 143-1000 Hz, with harmonics in the range of 500-5000 Hz. Spectral contents outside of this range were found to be either to low to be perceptible, or too shrill.

Listeners were found to learn 4-6 warning signal types quickly, after which learning slowed to a maximum of ten signals. Speech warnings were found to be problematic. On the one hand, they were highly versatile in that they could convey any meaning expressible in language. However, they also tended to interfere with other cockpit communication, and often did not to contrast enough with pilot's communication to signal a warning effectively. This qualification on the use of speech cues is consistent with Kramer's definition of auditory display as being composed of non-speech cues in order to rely on reactions acquired through evolution, and not on cognitive processing. This view of speech is also consistent with Bregman's observations. Many perceptual researchers have noted that the processing of spoken cues appears to rely on a specialized area of the brain that humans have developed. It is reasonable to speculate that speech perception, not being a primitive percept, relies on a higher level of functioning that requires more time and learning for proper decoding.

2.2.3 Monitoring Implementations

Kramer and Fitch (1994) describe simulation of an operating room environment with auditory cues. Students played the role of anesthesiologists, with eight vital signs of a virtual patient represented. Taken through a series of simulated emergencies, students responded more quickly and accurately to the auditory cues than they did to a similar simulation involving visual cues. This is another example of the ear's effectiveness in tracking multiple streams of information.

Wenzel (1994) cites three areas explored by NASA Ames laboratories. The first, in telerobotics, was a virtual reality environment that allowed remote maintenance of machinery in distant locations, via goggles and gloves through which operators could perform actions such as inserting circuits into a circuit board. Auditory cues served as reinforcement to the actions being performed. Physical contact with objects was registered by a beep sound. Similar cues signified correct or incorrect insertion of the parts. Often the virtual hand's proximity to a target was uncertain, so a range finder produced two simultaneous tones, one of which

changed frequency as the glove veered from the target. Successful approach was indicated when the two tones were at the same frequency.

The two other areas explored by NASA involved spatialization. In an air traffic controller simulation, subjects wore headphones in which the location of close aircraft was simulated by interaural intensity differences. Responses were found to be faster with these spatial cues than were responses to visual cues or non-localized audio cues. In a similar experiment, shuttle launch communications were transmitted via headphones, with each voice separated by different simulated location. Subjects found the multiple voices to be much more intelligible when localized than when all voices appeared at equal volume through one speaker. This result corroborates with the spaciousness observation of early audio engineers to stereo broadcasts, as well as with Bregman's discussion of localization as an enhancer of other streaming cues.

Jameson (1994) describes a software debugger that is enhanced by auditory cues. He points out that fixing bugs is often quite simple; it is finding them that can waste hours or days of time. His system has enabled him to detect bugs quickly through the use of sound to register events such as beginning, incrementing and ending loops. Jameson gives two examples in which a tone sounded the initiation of function calls. The tone corresponding to one function would change in volume with each iterated loop, the tone associated with the other function would change in timbre. Not hearing the expected changes, either steady increase in volume or brightness, enabled quick identification of where the program's bug was likely to be found. This sonification, like Kramer and Fitch's operating room simulation, makes use of the auditory system's ability to track multiple streams of information.

2.2.4 Analysis Implementations

Kramer's 1994 assessment of auditory analysis is that while it is a provocative prospect, there has not yet been a successful enough demonstration of it to result in any universal implementation. His point is borne out by the fact that most of his references are either to unpublished theses or out-of-print volumes.

Further evidence of the difficulty is shown by Bly (1994). In preparation for the 1992 ICAD, participants were given two data sets and asked to perform the

exercise of creating sonifications of them. The first challenge was a multivariate identification, the second was an analysis problem. The multivariate data set involved the mapping of six soil characteristics in an attempt to identify which soil types were likely to contain gold. With this static data exercise, a number of interesting sonifications were created. The analysis exercise contained a set of time-varying data in which six atmospheric measurements were supposed to determine the likelihood of thunderstorms. Three sets of measurements were given, each representing 100 days of data. For the first two analysis sets, stormy days were given; the participants' task was to try to identify which days in the third analysis set were likely to have storms. The difficulty of this second challenge was great enough that no one submitted a sonification solution. Thus, the pattern of analysis involving pattern recognition is not a trivial problem. Its difficulty, however, makes the successes to date that much more compelling.

2.2.4.1 Rings of Saturn

Kramer (1994) cites an example from NASA's space exploration history. In 1979 the Voyager craft reached the rings of Saturn. The eight-channel plasma wave data was translated into sound. Each electrical field's frequency was in the audible range, so it was a straightforward mapping of the frequency values to a synthesizer program contained in an Apple II computer. Various wave types were easily distinguishable. The audification was received as a pleasant novelty, but appeared to contain no particular scientific contribution. In 1981, however, the Voyager 2 craft transmitted some peculiarities that could not be traced to any information contained on the visual printouts. When the audification was employed, the cause of the irregularities became clear. The plasma was giving many of the dust particles in the rings a negative charge. The irregularities were the result of these particles striking the craft, and creating an electromagnetic "splash" across the frequency bands that came across as a distinct "machine gunning" sound in the audification.

2.2.4.2 Seismology

Chris Hayward (1994) describes the use of sound for seismology. Seismology involves the study of waves in the range of 40 Hz that travel through the surface of the earth. Hayward transposes their frequencies into auditory regions for analysis through listening. At the same time, he takes advantage of the fact that

audified data can be played back at any desired speed, and thus a new form of data compression is achieved.

Hayward describes two branches of seismology, exploratory and planetary. Exploratory seismology involves placement of geophones in concentric circles and a controlled impulse from a hammer or small explosion is sent into the ground. The reflections from the impulse are captured by the geophones. People interested in a local geology, such as civil engineers use the information. By audifying the data, Hayward speculates that the training time involved in learning to recognize significant patterns can be reduced. Also, audifying the patterns as they are recorded can be of help on-site, since supervisors of exploratory seismology are kept constantly busy scheduling successive impulses, giving directions and following up on general troubleshooting. If they were able to listen to cues rather than watch them on a visual monitor, their eyes would be free to attend to the multitude of other tasks before them.

Planetary seismology deals with large-scale sources of disruptions that keep the earth's surface in constant motion. Sources may be volcanic activity, earthquakes or nuclear explosions. Data is gathered at numerous observatories situated in quiet spots around the globe, and the results are compared and correlated to trace the time and place of various events. Again, Hayward cites the "eyes free" heuristic as being of particular value. The work at these observatories involves constantly monitoring information from many different sites, and decisions must be made about which sources to examine more closely. Given the auditory system's ability to track simultaneous streams, the efficiency of these decisions could be increased by audifying, rather than visualizing, spectrographic data from different sites.

Hayward also cites certain "ringing" patterns that show up in his audifications that do not correspond to anything in conventional visual displays. This observation suggests that there may be information present in the seismological signals that is better represented in an auditory display than a visual display.

Hayward is not the only seismologist who has audified earthquake data. CNN reported on February 5, 2000[2] on Andrew Michael, a U.S. Geological Survey researcher who also audifies sped up versions of seismograms. Michael's aims are apparently more pedagogical than analytic. In lectures on the physics of earthquakes and the processes of seismograms, musical instruments are employed as analogs to the physics of earthquakes. A trombone slide, for example, is used to show the effect of the wave propagations in the earth. A musical performance features an audified seismogram that is looped to form a rhythm track. A trio of trombone, vocal and cello plays melodies that are meant to represent the stresses within the earth's surface. Apparently, the audifications contained an unexpected "windy" noise, the source of which neither he nor Chris Hayward was able to identify.

2.2.4.3 Financial Analysis

Kramer (1994B) presents a multi-variate representation of financial data. Two pulsing tones sonify 265 data points, representing American financial indices from September 1987 through March 1992. The display contains five dimensions. Closing figures of the Dow Jones Industrial Average are mapped to the tones' pitch. Bond prices, taken from the Lehman Brothers T-Bond Index, are mapped to the pulsing speed. The value of the U.S. dollar, taken from the J.P. Morgan Index vs. 15 Currencies, is mapped to brightness (strength of higher harmonics). Interest rates, taken from the Federal Funds Rate, New York Federal Reserve, are mapped to detuning (slight difference in frequency between the two tones). Commodities, from the CRB Futures Index, are mapped to attack time. The dimensions are flexible. Some examples feature more than one parameter applied to an index for clarity, with stereo location an added parameter. Examples are contained on the CD that accompanies *Auditory Display: Sonification, Audification, and Auditory Interfaces.*

Some of the dimensional parameters are more apparent than others. Some training would be required to appreciate subtle differences in detuning and attack time. Kramer introduces a reference "sound bite" that he terms a *beacon*. Using beacons to represent a smaller number of points focused at market extremes, the

---

[2]*http://www.cnn.com/2000/SHOWBIZ/Music/02/05/earthquake.music/index.html.* Chris Hayward reported his impressions of one of Michael's lectures in private correspondence.

combined effect of the five dimensions can be learned more readily as a combined gestalt.

This paper was written some years before the advent of "day trading," in which investors (or would-be investors) buy and sell stocks quickly via special software packages. Day traders make decisions on a minute-by-minute basis, tracking various indices to time a decisive mouse-click to buy or sell. It is easy to imagine a flexible investment auditory display package with which traders could set sonic parameters to track chosen indices or even individual stocks. Such a display could be used in both monitoring and analysis applications. For monitoring, real-time changes could be tracked. For research, sets of dates could be sonified to compare trends at different times.

2.2.4.4 Quantum Mechanics

Researchers at UC Berkeley used sonification to detect quantum interactions (Pereverzev, *et al.*, 1997). Quantum mechanics equations have long predicted particle current oscillations between two weakly coupled macroscopic quantum systems, although these oscillations had never been observed. These researchers used two reservoirs of a helium superfluid. Membranes in the reservoirs traced voltage changes. Oscilloscopes revealed nothing useful in terms of the oscillations between the two reservoirs, but when the voltage was audified, a clear tone revealed the expected oscillations. Further observations were then carried out through the study of sound recordings of these tones.

2.2.4.5 Fluid Dynamics

McCabe and Rangwalla (1994) look to auditory displays to improve the representation of computational fluid dynamics data. The data describes fluid flow, analyzed within a grid of three-dimensional volumes. Visualization programs are able to represent the three dimensions effectively, but the illustration is static, as these programs (Plot3D, FAST) are not well suited to reflect changes in time. An auditory representation is their solution for the presentation of data of higher than three dimensions. They present two examples.

The first example is a model of an artificial heart pump. Fluid dynamics give solutions for location of blood cells, pressure and vorticity at various points within

the heart chamber; critical points for their exploration of the heart cycle were the moments when the valves opened and closed, and the times at which blood cells reached unsafe levels of vorticity. Their model added auditory elements to a visual animation by sending data to a MIDI synthesizer. The auditory elements sonified three of the model's components. The pressure plate's changes were mapped to a continuous tone, with pitch-bend changes corresponding to changes in pressure. A note-on message to a wood block timbre corresponded to particles reaching threshold vorticity. A note-on message to a bass drum timbre corresponded to valves opening or closing. Informal responses to the combined audio-visual display were favorable. The auditory cues made it easier to correlate the activities of the various components, allowing researchers to focus visually on one area of the visual display and listen for a cue from another area.

Their second example is an audification of pressure changes inside a jet turbine. The compressor produces rotating air pressure patterns at a potentially infinite number of harmonics to the blade's rotating frequency. Such engines are modeled in computer programs, with wave equations simulating the pressure changes. McCabe and Rangwalla divided the area into a grid and "sampled" the pressure changes within each grid. Listening to the resultant audio signal and observing the changes in timbre gave them insights into the changes over time among the harmonics of the rotating pressure patterns.

2.3 Heart Rate Variability

Having completed a survey of work to date in auditory display, the discussion will now shift to background work in the field of cardiology, the focus of the sonification models to be presented in upcoming chapters. As the focus of this thesis, discussion of this work will necessarily contain more detail than did the previous sections.

As stated in the Introduction, heart rate variability is the measure of changes in interbeat interval times. The analysis of HRV can be broadly classified into two methodologies. One considers absolute time, with the heartbeat indexed by a continuous clock. This approach introduces problems since it requires interpolation of a curve to estimate a function that would include the data points on a best-fit basis. This work will focus on the second approach, which focuses

on the actual interbeat interval recorded, and indexed by *beat number* rather than absolute time.

2.3.1 Spectral Analyses

While digital signal processing operations can be performed on any discrete series of measurements, what they represent depends on the contents of the series. The fluctuations reflected in a discrete Fourier transform, for example, need to be seen in the context of what the data set represents. An NN interval series, as discussed in the Introduction, is derived from a point process series rather than a time series. Its Fourier transform does not reflect the frequency content of a continuous signal, but rather the changes present within the continuous signal; as such, the Fourier transform of an NN interval series is analogous to its first derivative, reflecting the frequencies of the signal's fluctuations[3].

Spectral interpretations of NN interval sets fall into four frequency ranges (Roach, 1996):

High Frequencies (HF)	.15 - .4 Hz.	Related to respiration.
Low Frequencies (LF)	.04 - .15 Hz	A ≈.1 Hz cycle (10 seconds) likely related to blood pressure
Very Low Frequencies (VLF)	.003 - .04 Hz	
Ultra Low Frequencies (ULF)	≤ .003 Hz	

The VLF and ULF regions are of particular interest, since they seem irregular and not associated with any physiological cause. Changes in these regions are likely due in part to changes in external activity, such as sleeping or exercising, or in emotional condition. Exploration of these regions plays a major role in the analytical methods to be described subsequently. A variety of methods will be described that are performed to extract externally based factors from the intrinsic behavior of the heart. The intrinsic spectra of frequencies lower than 0.1 Hz tends to display $1/f$-like characteristics (Peng, *et. al.*, 1993). ($1/f$ characteristics are discussed in Appendix 2.)

---

[3]Private consultation with Carsten Schaefer of McGill's Center for Nonlinear Dynamics.

2.3.2 Statistical Analyses

Statistical analyses form another class of HRV operations (Task Force of the European Society of Cardiology and NASPE, 1996). Statistical measurements include the mean of a measured timespan and the *SDNN*, the standard deviation of a timespan. The *RMSSD*, root-mean-squared standard deviation, is the standard deviation of interbeat interval differences. Changes due to cycles of less than five minutes are represented by the *SDNN index*, which is the mean of a series of 288 standard deviations over five minute periods, spanning twenty-four hours. Changes in cycles greater than five minutes in length are represented by the *SDANN* (standard deviation of average normal to normal intervals), which is the standard deviation of a series of mean values over five minute periods.

The SDANN is a form of lowpass filtering. It can be represented by the lowpass filter difference equation that is familiar to audio filter designers. For N samples over a span of 5 minutes, we have:

```
output = (1/N)*x[n] + (1/N)*x[n-1] + (1/N)*x[n-2] . . .
```

A more static view is given by the NN50, which is the total number of successive interval differences exceeding 50 ms (that is, beats representing a sudden change in the heart rate). As a statistical measurement, the NN50 count is a single number, and gives no indication of the heart rate activity as a function of time.

A geometric view is given by the NN interval histogram, in which the intervals are categorized into bins that span .0078125s (1/128). The number of intervals within the set that falls into each bin is plotted vertically. Cardiologists then may analyze the area under the resultant curve, or attempt to create a function to describe its shape.

2.3.3 Nonlinear Dynamics

2.3.3.1 Nonlinear Dynamics and Biological Systems

Conventional statistics are often not successful in differentiating between significantly different cardiological conditions. Heart rate data from a healthy subject and a patient who has just suffered a heart attack may contain identical means and standard deviations, while even a naive observer can differentiate

between the two data sets when they are plotted (Goldberger, 1999). With complex data sets of this nature, conventional statistics offer just one lens through which to view them. Important information is often obtained via methods taken from nonlinear dynamics. Nonlinear dynamics have revealed patterns concealed by conventional statistics in a number of aspects of human physiology, including respiration, gait and white blood cell counts (Goldberger, 1996). A summary of nonlinear dynamics fundamentals is provided in Appendix 2.

Many cardiologists suspect that heart dysfunctions are the result of overly regular cycles. Cardiac tissue is an example of an *excitable medium* (Kaplan and Glass, 1995). An excitable medium is one that propagates waves, yet can only support waves with a suitable length of time between them. The oscillations produced by the sinus node travel in a circuitous path, with excitations moving in opposite directions from the sinus node. When these oscillations meet at a point on this pathway opposite the sinus node, they cancel each other out under normal circumstances. Under some conditions, however, the wave fronts do not cancel each other, and the result is a *re-entrant wave* that cycles continuously throughout its path. The excitations from the sinus node are then overridden by the re-entrant wave, so that the sinus node no longer functions as the pacemaker. The synchronization present within a healthy heart then breaks down, a condition known as *atrial fibrillation*. At the opposite extreme is *congestive heart failure* (CHF), a condition in which a ventricle is not pumping properly. CHF data sets may contain little or no variability, appearing as a flat line. They also may display low-amplitude oscillations within a frequency range of 0.01-0.02 Hz (50-100 seconds per cycle), corresponding to a cyclical respiratory condition that originates from the central nervous system known as *Cheyne-Stokes* respiration[4]. People who suffer from congestive heart failure are at high risk for sudden cardiac death (Peng, *et. al.*, 1999).

Many nonlinear dynamics approaches require that the behavior of the system under observation be similar throughout its duration. Such a system is termed *stationary* (Kaplan and Glass, 1995). One definition of stationary behavior is that

---

[4]A *central* condition refers to a problem originating in the central nervous system. With a condition such as Cheyne-Stokes respiration, the brain is not sending the signals that initiate normal respiration (NIH, 1995).

the mean and standard deviation remain unchanged throughout the series (Kaplan and Glass, 1995; Ivanov, *et. al.*, 1996). Biological systems, however, are typically non-stationary, as local means and standard deviations can vary for different time intervals of a time series. Many of these drifts result from low frequency fluctuations that, as stated above, are due to external factors (Roach, 1996; Peng, *et. al.*, 1995; Viswanathan, *et. al.*, 1997). To analyze non-stationary data sets such as NN intervals, which may span a period of hours, signal processing may be applied to the time series so that it exhibits *stationarity*. The purpose of the signal processing is to extract the nonstationarities due to external factors. The processed data set presumably reflects the internal heart dynamics.

Dynamical systems may be *correlated*, meaning present values are related to past values, even those that occurred many hours earlier (Pilgram and Kaplan, 1997). A correlated system "has a memory" in that its values are not random, as in white noise, but *deterministic*, in that present values determine future values. In biological systems, correlations that extend over multiple scales of space or time are sometimes termed *fractal ordering* (Goldberger, 1999). An HRV time series may be analyzed as an example of a correlated time series. Nonlinearities due to external factors have a shorter "memory," indicated by correlations that exist over shorter time scales. On the other hand, nonlinearities due to inherent dynamics show longer-term correlations.

2.3.3.2 Magnitude Fluctuation Analysis

A magnitude fluctuation analysis, notated $F(n)$, is one method for illustrating correlations over different timescales (Peng, *et. al.*, 1993). An interval set is lowpass filtered to remove fluctuations over time periods greater than three minutes or so, and notated $B_L(n)$. The analysis is performed by choosing a difference in beat index, n, than beginning with the first beat and moving through the time series sequentially, $n' = 1, 2, ...$, taking the difference between each beat and the one n beats ahead of it over the entire interval set, and then taking the average of these differences. The process is represented by the following equation, in which the bar indicates an average over all difference values over the course of the set:

$$F(n) \equiv \overline{|B_L(n' + n) - B_L(n')|}$$

F(n), representing the magnitude of fluctuations over beat difference *n*, is then taken for many values of *n*. The values *F(n)* are then plotted as a function of *n* on a log-log plot. The slope of the resulting line shows the degree of correlation within the series. The slope is termed the *scaling exponent*, α. A healthy subject will have a scaling exponent near zero, which corresponds to the "memory" inherent in the pink noise generating algorithm described by Voss and Clark, described in the discussion of Scaled Noises in Appendix 2. A diseased subject will have a slope near 0.5, which corresponds to a random walk or Brown noise, showing that for such a diseased state, the beat intervals are uncorrelated on a scale greater than three minutes. Figure 2_3 shows lowpass filtered interbeat interval plots for a healthy subject and a diseased subject (dilated cardiomyopathy). The bottom graph shows the magnitude fluctuation for each, with reference lines to show a slope of $\alpha = 0$ for $1/f$ noise and $\alpha = 0.5$ for Brown noise.

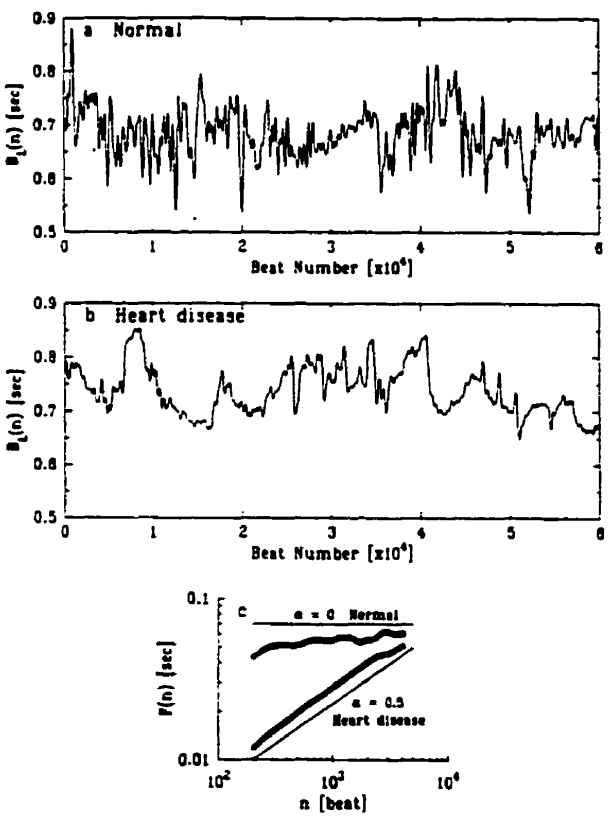


Figure 2_3: Magnitude fluctuations for healthy and diseased subjects (Source: C.K. Peng, et. al., Physical Review Letters 70, p. 1344, 1993. Copyright 1993 by the American Physical Society)

For purposes of reducing external factors, it is useful to use the first derivative, or *first-difference series*, of the set $B(n)$. The first difference series is obtained by taking the inter-interval differences, $I(n) \equiv B(n+1) - B(n)$. This process removes linear trends in the series and often generates stationarity (Kaplan and Glass, 1995; Viswanathan, *et. al.*, 1997). In heart rate analysis, linear trends are likely due to external factors. The first difference series is meant to remove external factors.

The differences between diseased and healthy states can become blurred when their interval differences are reduced to a static interval histogram. Figure 2_4 compares the interval histogram of $I(n)$ for both for both healthy and diseased time series. Both histograms are seen to be identical. The histogram similarity signifies that it is the *sequence* of inter-beat interval differences, and not the set of intervals themselves, that distinguishes healthy from diseased patients (Peng, *et. al.*, 1993).

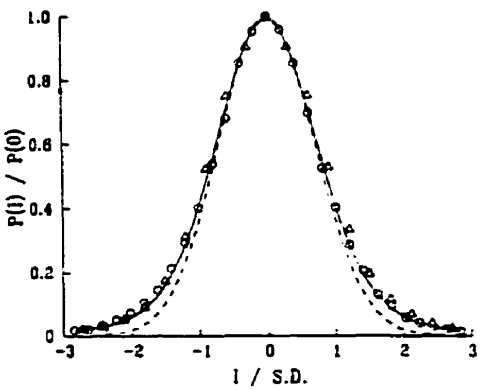


Figure 2_4:
HRV interval histogram for healthy (circles) and diseased subjects (triangles) compared with Lévy stable distribution (dashed)
(Source: C.K. Peng, et. al., Physical Review Letters 70, p. 1344, 1993. Copyright 1993 by the American Physical Society)

2.3.3.3 Spectrum of First Difference Series

The value of $I(n)$ is in its power spectrum, which is created by plotting the series as a function of beat number, taking the FFT of the function, and squaring the amplitudes. The power spectrum is only meaningful for stationary signals, since linear trends can mask the underlying frequency content (Peng, *et. al.*, 1993). Thus, the first-difference series allows the power spectrum to be implemented in a useful way. The slope of the power spectrum, when plotted on a log-log plot,

determines the degree of correlation among NN intervals. Notated β, it is related to α by $\beta = 1 - 2\alpha$. If $\beta = 0$, then there is no correlation in the time series, making it analogous to white noise. If $-1 < \beta < 0$, the correlation is such that positive values in *I(n)* are likely to be close to each other, as are negative values. If $0 < \beta < 1$, then positive values are more likely to be followed by negative values, and vice versa, an *anticorrelation*.

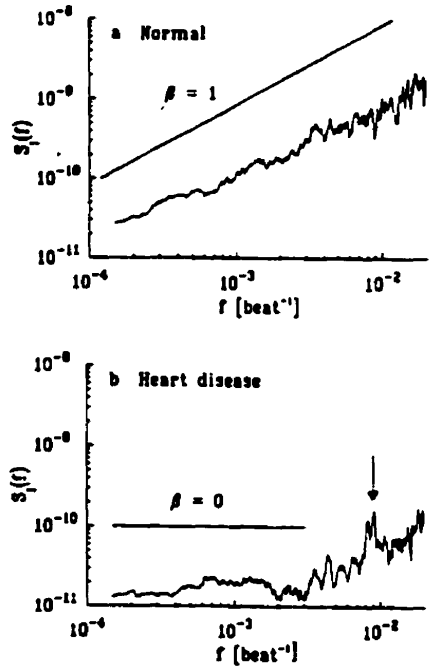


Figure 2_5: Power spectra for healthy and diseased subjects
(Source: C.K. Peng, et. al., Physical Review Letters 70, p. 1345, 1993. Copyright 1993 by the American Physical Society)

Figure 2_5 shows the power spectrum for a healthy subject and a subject with heart disease. For the diseased subject, the slope of the power curve is nearly flat for the very low frequencies (longer time scales), which suggests that this subject does not display correlation (deterministic patterns) over longer time scales. The slope of the power curve for the healthy subject has a value of β close to 1.0 for all frequencies (time scales), indicating an anticorrelation over longer time scales.

The principle of *homeostasis*, introduced by Walter Cannon, is defined as a constant internal environment within the prescribed limits for cellular life (Cannon, 1929). Many researchers have assumed (Peng, *et. al.*, 1995) that regulating mechanisms within an organism will work to keep it at a uniform state.

Figure 2_6
Detrended Fluctuation Analysis

Take an NN interval series, *B(i)*. Each value of *i* is an NN interval value. The sequence has *k* values.

Take the mean of *B(i)*, B_{avg}.

Integrate the series *B(i)*, into a new series, *y(k)* :

$$y(k) = \sum_{i=1}^{k} [B(i) - B_{avg}]$$

y(*k*) represents an integrated version of *B(i)*.
Subdivide y(*k*) into windows of equal length, *n* points each.

Create a least-squares line in each window, which represents the trend within the window.
This line of *n* points is $y_n(k)$.

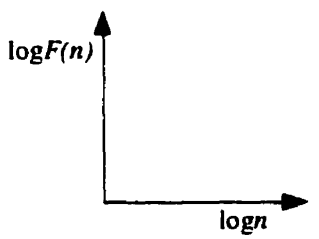
Within each window, take the difference between each corresponding point of y(*k*) and $y_n(k)$. Use each difference measurement to get the RMS of the fluctuations present within each window, *F(n)*:

$$F(n) = \sqrt{\frac{1}{N} \sum_{k=1}^{N} [y(k) - y_n(k)]^2}$$

Get a single value *F(n)* for each window. Take the average of all *F(n)* values.

Repeat for all sizes of *n*.

Make a log-log plot, with the average *F(n)* value as a function of each corresponding value of *n*.



The slope of the line is also called the scaling exponent, α. It gives information about the degree of correlation within the series *B(i)*.

$\alpha = 0$:	random, white noise
$0 < \alpha < 0.5$:	power law correlation; large and small values alternate
$\alpha = 0.5$:	random walk, uncorrelated
$0.5 < \alpha < 1.0$:	anticorrelation; large values tend to be followed by large values, small values by small values
$\alpha = 1.0$:	1/*f* noise
$\alpha > 1.0$:	non-power law correlations
$\alpha = 1.5$:	Brown noise

Sources: Viswanathan, *et. al.*, 1997; Goldberger, 1999.

This assumption has been modified in light of the erratic nature of healthy systems, and the correlative properties of their fluctuations. Many current models are based on *stochastic feedback systems*, that is, regulatory systems that maintain fluctuations within safe limits, keeping the system from reaching extreme values. The anticorrelations illustrated in the magnitude fluctuation analysis may suggest such a regulating mechanism (Peng, et. al., 1993). Absence of this regulation may underlie certain diseased states that are characterized by *mode locking*, an inflexible periodic state observed in some types of malignancies such as Cheyne-Stokes respiration (described above), sudden cardiac death, epilepsy and fetal distress syndromes (Peng, *et. al.*, 1993). An HRV series that displays either random walk, white noise or high periodicity is likely to be indicative of a diseased diagnosis. This idea is termed *complexity loss in disease* (Lipsitz and Goldberger, 1992; Goldberger, 1999). The regulating mechanism of a healthy set keeps it from reaching any of these steady states.

2.3.3.4 Detrended Fluctuation Analysis

Another method of removing nonstationarities in a signal is the *detrended fluctuation analysis* (DFA). Once this process has been performed the fractal dimension of the time series may be estimated by a method similar to the "box counting" technique described in Appendix 2. The steps for the DFA are shown in Figure 2_6.

2.3.3.5 Cumulative Variation Amplitude Analysis (CVAA)

The spectral results described above contain an inherent shortcoming found in all Fourier transforms, which is that there is always a conflict between the resolution of time and frequency. The *cumulative variation amplitude analysis* (Ivanov, *et. al.*, 1996; Ivanov, 1999) offers more refinement through a series of *convolution* operations.

The convolution h of two discrete signals x and y is described by the equation

$$h(n) = (x * y)(n) = \sum_{m=0}^{n} x(m)y(n-m) \qquad n = 0, 1, 2, \ldots$$

Graphically, the process can be visualized as sliding one signal over another, one entry at a time, multiplying overlapping members at each time increment, and taking the sum of these products.

```
  Output
Time     Value              [ ... y4  y3  y2  y1  y0 ]
                                                  [ x0  x1  x2  x3 ... ]

0    x0*y0                  [ ... y4  y3  y2  y1  y0 ]
                                                [ x0  x1  x2  x3 ... ]

1    x1*y0 + x0*y1             [ ... y4  y3  y2  y1  y0 ]
                                                [ x0  x1  x2  x3 ... ]

2    x2*y0 + x1*y1 + x0*y2        [ ... y4  y3  y2  y1  y0 ]
                                                [ x0  x1  x2  x3 ... ]

3    x3*y0 + x2*y1 + x1*y2 + x0*y3    [ ... y4  y3  y2  y1  y0 ]
                                                [ x0  x1  x2  x3 ... ]
  ... etc.
```

Convolution is a crucial signal processing operation due to the tenet that the convolution of two signals produces a multiplication of their spectra. Filtering any signal can be described as a convolution of that signal with the filter's impulse response, with the result that the spectrum of the signal is multiplied by the frequency response of the filter.

A discrete Fourier series is theoretical in that a finite set of signals is assumed to represent one period of a signal with an infinite length. This transform is accomplished by convolving the signal with a series of signals that represent the signal's harmonics. Through the resultant spectral multiplications, the contribution of each harmonic to the signal may be quantified. Thus, a Fourier transform is analogous to the output of a set of bandpass filters at a fixed bandwidth.

These harmonics may be termed a *basis set,* which describes a set of linked (dependent) basis vectors in a system, such that any point in the system may be described as a linear combination of these basis vectors; conversely, these basis vectors are able to describe any point in the system. The Fourier transform of a continuous time signal has a basis set consisting of an infinite number of vectors that are able to describe any possible frequency component of the analog signal. A discrete Fourier series of N data points has a basis set of N vectors, which represent N harmonics of the signal.

The infinite theoretical length of the signals convolved by the Fourier transform make it an effective process to describe stationary signals, but it has shortcomings in the description of non-stationary signals. Any transient behavior in the signal will be interpreted as resulting from frequency components whose contributions of constructive and destructive interference over the total length of the signal produce the transient. The non-stationary nature of the transient is translated as a coincidence brought about by the amplitudes and phases of a number of stationary components. The appearance of the transient will be lost in the Fourier transform. Thus, it is said that a Fourier transform loses *time localization* of events in a signal, resulting in the tension between time and frequency resolution mentioned above. A longer signal will contain more points, and thus a Fourier transform will reveal the presence of more harmonics. Any non-stationary activity will be "smeared" over the length of the signal. A shorter signal will be better able to describe the timing of an event, but due to its fewer data points, will be translated as containing fewer frequency components. There is an inverse relationship between time and frequency resolution. This tradeoff is analogous to the *uncertainty principle* in physics, which states that the better a particle's position can be observed, the less accurate its velocity can be estimated, while a more accurate measurement of its velocity will bring about more uncertainty as to its precise position. Since biological systems, including the heart, are typically non-stationary, some information will necessarily be lost in a Fourier transform of its behavior as a function of time.

One solution to this tradeoff is to divide a signal into shorter pieces, called *windows*, and to perform a *Short Time Fourier Transform* (STFT) of each window. A suitable window length must be found to provide a workable compromise, since the shorter window will divide the spectrum into fewer components. Each window, representing one period of a waveform, will thus have a higher fundamental frequency than the entire signal would have if it were to be transformed. This solution is not suitable to a heart rate variability set since the spectral components of interest are those in the lower frequency ranges.

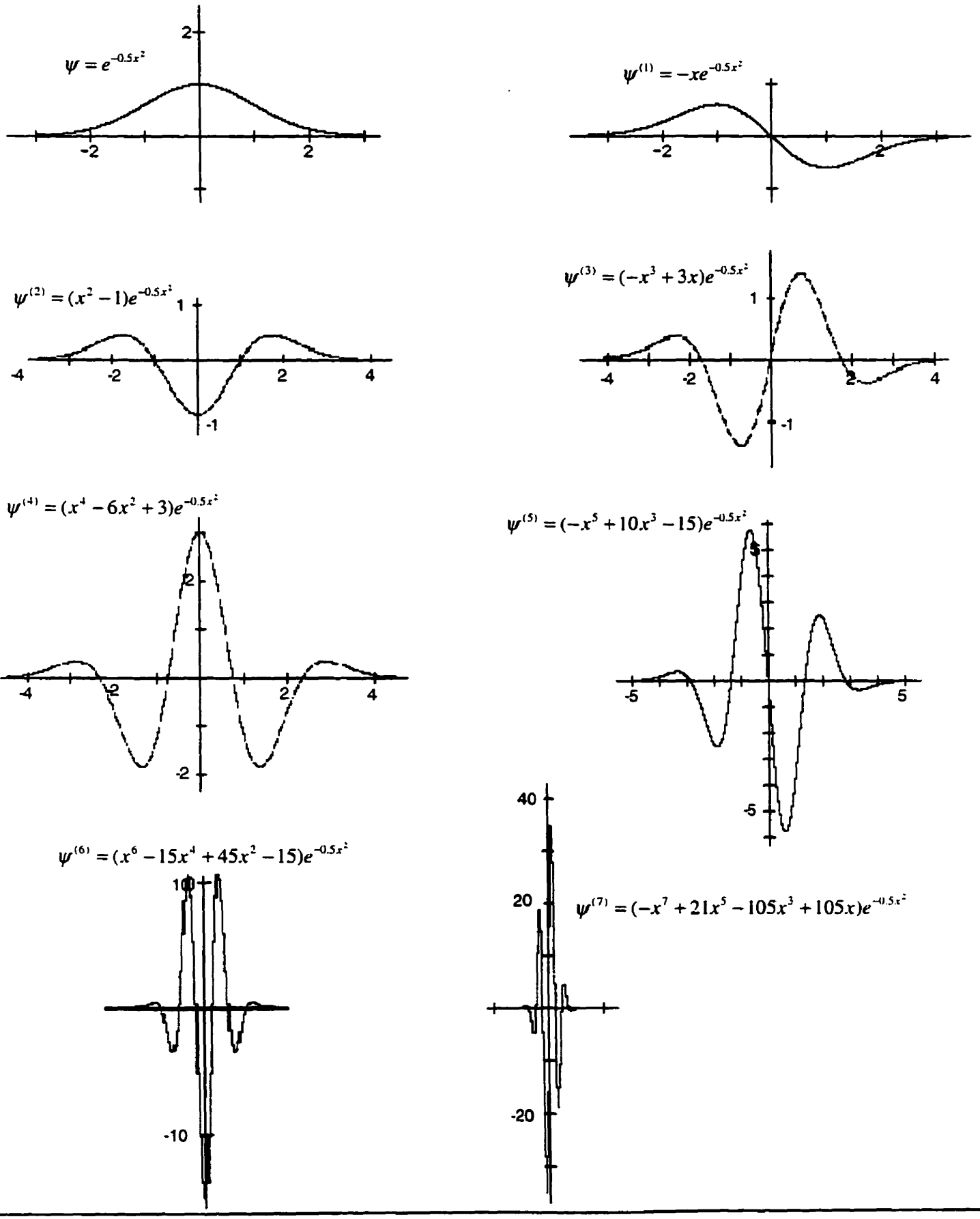
The first step of the CVAA is to use an alternative transformation method, the *wavelet transform*. This wavelet transform is a variant of the Fourier transform that has been widely reported in a number of fields, computer music among them (Kronland-Martinet, 1988; Strang, 1989; Graps, 1995; von Baeyer, 1995). The

wavelet transform is an attempt to resolve the time-frequency resolution mismatch through an alternative basis set. The alternative basis set relies on the principle that the amount of time required to understand the behavior of a frequency component is dependent on that frequency. Lower frequencies evolve slowly, and thus require a longer observation time to be understood, while the opposite is true of high frequencies. The basis set for a wavelet transform, in contrast to that of a Fourier transform, consists of a series of signals that have a finite effective duration characterized by their *scale*. The length of the wavelet signal is the inverse of its scale. All scales contain the same number of cycles of the wavelet signal. Higher scales, then, are higher frequency representations of the wavelet.

The spectral result of the wavelet convolution is dependent on the wavelet shape and its scale. There are many wavelet types, which are intended for specific analysis applications. The bandwidth of a wavelet is dependent on its scale. Signals of shorter length are composed of a greater number of frequency components than are longer signals. The extreme example is an impulse signal, consisting of a single value of 1 followed by zeroes. The spectrum of an impulse is all frequencies at equal amplitudes, as can be demonstrated mathematically by a Fourier transform. Roads (1996) offers an intuitive explanation. Just as transients in a Fourier transform are interpreted as being the result of the interaction of many components, so is the nature of a finite signal, which starts and ends due to the presence of many components that combine in such a way that amplitude values of zero exist outside of the observation window. Thus, the length of a signal is inversely proportional to its spectral bandwidth. Therefore the result of a wavelet transform is analogous to the output of a series of bandpass filters at a fixed Q (center frequency/bandwidth). At the same time, the shorter length of higher scaled wavelets allows for better time localization of high frequency behaviors, so that transients may be better represented than in a Fourier transform.

For heart rate variability analysis, a wavelet transform offers a greater degree of refinement in extracting the heart's intrinsic dynamics than does the integrated time series used in the magnitude fluctuation analysis described earlier. Different scales of the wavelet can extract features over different time scales. Wavelet filtering, as used in the cumulative variation amplitude analysis, is a first step towards revealing distinctively different characteristics between healthy cardiac dynamics and unhealthy cardiac dynamics that display cyclic behavior. With a

Figure 2_7
Derivatives of Gaussian Function



wavelet scale corresponding to the length of the cycles, the filtering is highly responsive to cycles of this length and represents them clearly (Ivanov, 1999). A Gaussian wavelet and its derivatives (examples are shown in Figure 2_7) are used in the cumulative variation amplitude analysis since these wavelet types are orthogonal to linear trends in the data that result from external factors. The wavelet scale is an integrating procedure, since transients that occur within the wavelet scale are smoothed over the length of the wavelet by the convolution process, while frequencies with a longer period than the wavelet length are not represented. A single wavelet scale that spans thirty-two beats is used by Ivanov to study characteristics of *obstructive sleep apnea*, that is characterized by heart rate cycles at 0.17-0.35 Hz, spanning 30-40 beats.

Obstructive sleep apnea is caused by excessive relaxation of muscles in the back of the throat during sleep. The airway becomes closed and breathing can stop for time periods on the order of a minute or so. Breathing is suddenly resumed by a loud snorting. These episodes may occur twenty to thirty times in an hour, hundreds of times in a night, without the sufferer even being aware of them. The symptom most noticeable for those in close proximity is a loud snoring. The daytime result is a loss of alertness due to lack of sleep, even to the point of suddenly nodding off. In the long term, apnea sufferers are at increased risk for high blood pressure, heart attack and stroke. There are an estimated 20 million apnea sufferers in the United States (McMillan, 1999). Most are not aware of their condition, a hazardous reality if these people work in professions requiring alertness, such as truck drivers, airline pilots or air traffic controllers. Ideally such jobs would require regular screening for sleep apnea, just as they require periodic vision tests. Sleep apnea can often be treated easily and non-invasively by devices such as machines that provide patients with continuous air pressure while they sleep. While doctors are required to report patients who suffer from apnea-related blackouts to the Department of Motor Vehicles, many apnea sufferers remain undiagnosed. The reason is that apnea diagnosis involves a patient spending a night in a sleep clinic, monitored by a variety of respiratory equipment. It is an expensive procedure and therefore not currently an economically justifiable element of routine job screening.

Sleep apnea is currently a compelling issue in cardiology, as these respiratory starts and stops have a distinct effect on the heart rate. Figure 2_8 is a segment of

an RR interval plot from a patient diagnosed with obstructive sleep apnea. The RR intervals, which are the inverse of the heart rate, are plotted as a function of time. The black triangles that appear along the time axis correspond to apneic episodes as identified by a respiratory analysis. Apneic episodes can also be recognized in the heart rate, which increases while breathing stops and quickly normalizes when breathing resumes. Since heart rate information is much easier to obtain than respiratory information, effective diagnosis of sleep apnea via the heart rate could make easy apnea diagnosis economical.

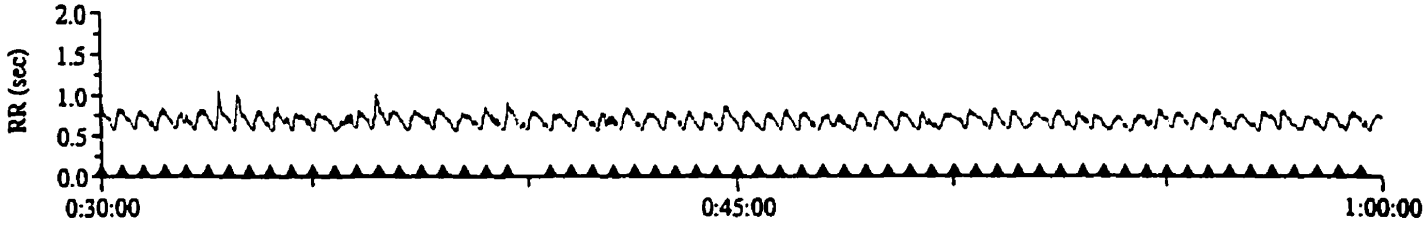


Figure 2_8: Segment of an HRV data set displaying obstructive sleep apnea

As described above, the first step of the cumulative variation amplitude analysis is to filter the NN interval series with a wavelet signal of a length that corresponds to the apnea cycles. The wavelet transform, like an integrated data set, provides a representation that is akin to the set's first derivative. The data set is converted to a series of positive and negative values that oscillate about a value of zero.

Following the wavelet filtering, the filtered signal is put through a *Hilbert transform*. The Hilbert transform is used to make a link between data sets used by physicists, who are accustomed to working with complex signals, and signal processing researchers, who are accustomed to working with real signals. The Hilbert transform produces a signal with an imaginary part such that when this signal is added to the original signal, the result is an *analytic signal* in which all negative frequency components have been removed. A signal x added to its Hilbert transform $h(x)$ produces an analytic signal z:

$$z = x + h(x),$$

and expressed as spectral components:

$$Z(f) = X(f) + H(f) ,$$

where $H(f)$ is the Fourier transform of $h(x)$.

The Hilbert transform is a complex operation that produces real and imaginary spectral components. The real components are the same as the positive spectral components of the original signal. As a result, when the spectrum of a signal is added to the spectrum of its Hilbert transform, the positive spectral components are doubled. The imaginary components of the Hilbert transform correspond to the negative spectral components of the original, but with the opposite sign. As a result, the negative spectral components are eliminated when the two spectra are summed. Thus, adding a signal with its Hilbert transform may be thought of as a spectral multiplication involving the *Heaviside function*, $U(f)$:

$$U(f) = \begin{cases} 1; & f \geq 0 \\ 0; & f < 0 \end{cases}$$

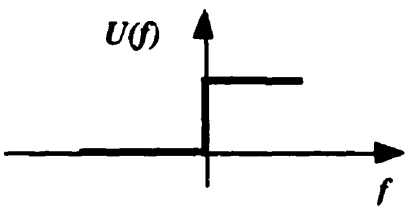


Figure 2_9: the Heaviside function

A Heaviside variant used to derive the Hilbert transform can be described in terms of a value and its sign S, such that $S(x) = \pm 1$. Any value x may be described as its absolute value $|x|$ times its sign $S(x)$; conversely, the sign $S(x)$ of any value x is the value x divided by its absolute value $|x|$. The spectrum of a signal's Hilbert transform, which reinforces the positive components of a signal and cancels out its negative spectral components, may thus be described as a multiplication of the signal's spectrum by twice the Heaviside function:

$$2U(f) = 1 + S(f)$$

Thus, the analytic signal may be re-written as:

$$\begin{aligned} Z(f) &= X(f) \times 2U(f) \\ &= X(f) \times [1 + S(f)] \\ &= X(f) + X(f)S(f) \end{aligned}$$

This equation, when combined with the definition of the analytic signal given above, produces:

$$\begin{aligned} Z(f) &= X(f) + H(f) = X(f) + X(f)S(f) \\ &\Rightarrow H(f) = X(f)S(f) \end{aligned}$$

Thus, the Hilbert transform of a signal is accomplished via a spectral multiplication. Spectral multiplication, as described earlier, is accomplished by time domain convolution, that is, a convolution of the inverse Fourier transforms of $X(f)$ and $S(f)$. According to a principal valued signal processing tenet, the inverse Fourier transform of $S(f)$ is $\frac{1}{\pi n}$ (Papoulis, 1977) . Thus the Hilbert transform is accomplished by the convolution:

$$h(n) = x * \frac{1}{\pi n}$$

The Hilbert transform of a signal produces a phase-shifted version of the signal, delayed by $-\pi/2$ radians, or 90°. With Euler's identity, $e^{i\phi} = \cos\phi + i\sin\phi$, as the basis of the Fourier derivation, it can be demonstrated, as shown in Figure 2_10, that the spectrum of the base case function $x(\phi) = \cos(\phi)$, when combined with the spectrum of its phase-shifted signal, $x(\phi) = \sin(\phi)$, produces an analytic signal consisting of only positive spectral components:

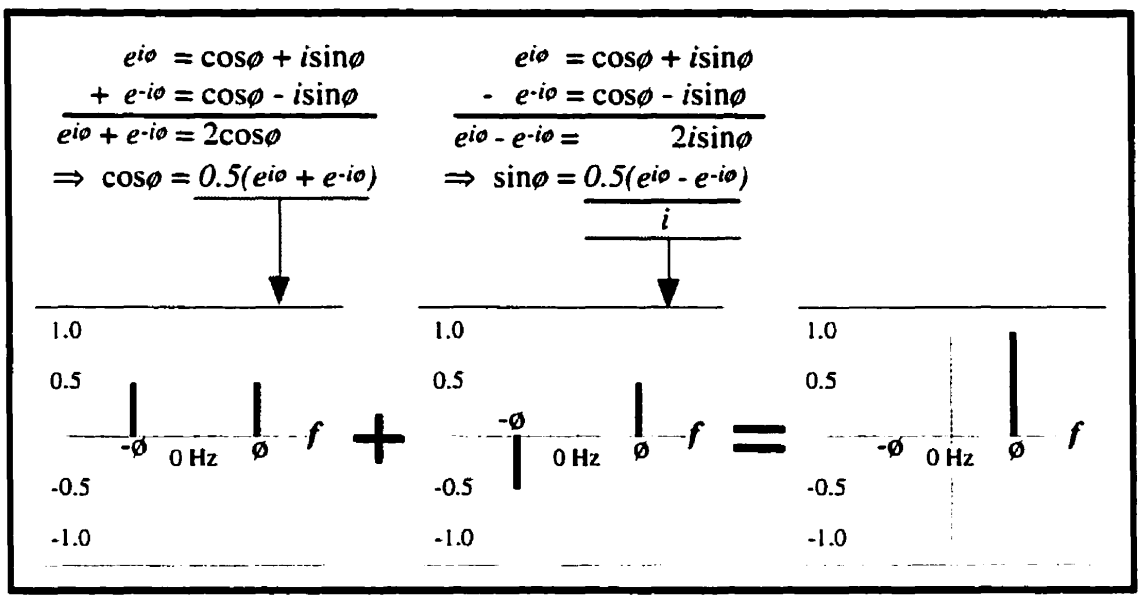


Figure 2_10: Analytic signal derived from a signal plus its Hilbert transform

The Hilbert transform has many applications in physics. A simple use is to derive a constant amplitude envelope for a fluctuating signal by taking the square root of the sum of the squares of the signal and of its Hilbert transform:

$$\text{amplitude} = \sqrt{x^2 + h(x)^2}$$

Using the simple example above, $\cos(\phi)$ combined thus with its Hilbert transform, the amplitude of an oscillating signal $A\cos\phi$ would be a constant value of A, due to the trigonometric identity $\sin^2 x + \cos^2 x = 1$.

All amplitudes taken from the Hilbert transform are thus positive. A line interpolated from amplitude point to amplitude point forms an envelope that provides both time and spectral domain information. In the CVAA, the oscillating signal produced by the wavelet filtering of a signal is combined with its Hilbert transform, after which an amplitude envelope is computed as described above. The values of this amplitude envelope are then put into a histogram. In this instance, a histogram of Hilbert amplitudes produces a uniform function curve that fits a group of healthy subjects. The shape of this function is a uniform probability curve, even when data sets of different lengths are used, indicating that it is scale-invariant. Sleep apnea subjects, however, do not have histograms that fall onto this uniform probability curve. The CVAA, then, represents a new level of information than that produced by the first-difference series, since the results of the first-difference series produced identical histograms from healthy and diseased subjects.

Additional information is derived from the generation of a *surrogate* data set (Ivanov *et. al.*, 1996). A surrogate data set is a set created artificially from a mathematical formula that is thought to underlie a real data set. Comparing the surrogate data set to an actual data set is a means of comparing the accuracy of the mathematical description. In this case, a Fourier transform is performed on the actual HRV time series. The phases of the spectral components are then randomized, and a surrogate set is generated that has the same spectral amplitudes as the original set, but with randomized phase values. When the CVAA is performed on the surrogate signal, the result yields a different probability curve from that produced by the original signal. Assuming that the difference is not an artifact of the transform process, the different probability curves suggest that the phases of the low frequency Fourier components play a critical role in differentiating healthy from non-healthy heart rate dynamics.

Further work by Ivanov, as yet in preliminary (and unpublished) stages, suggests that apneic episodes may be identifiable through a third step to the CVAA. Following the wavelet and Hilbert transforms, a median filter is applied to the

data set. The values are normalized to fall within the range ≤ 1.0 and a histogram is kept of values within different subdivisions of the total range of values.

A median filter is distinctly different from a mean filter, such as the lowpass filter described in the section describing statistical analyses. A mean filter takes the mean of all values within a window of data points. Abrupt changes are smoothed and widened. A median filter sorts all values within a window and outputs the mid-point of the sorted set of values. Thus, a median filter preserves abrupt changes in the data, giving a better representation of the range of interval sizes. Figure 2_11 gives an approximation of the difference between mean and median filtering.

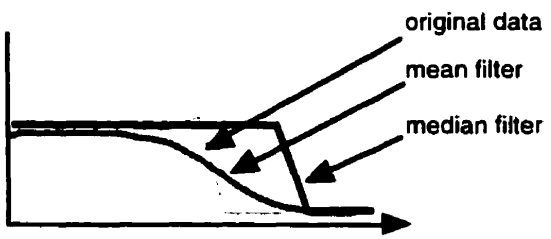


Figure 2_11: Comparison of mean and median filtering

Median filtering, applied to a set that has already been smoothed by the wavelet and Hilbert transforms, produces a jagged set of abrupt changes and plateaus. A median filtered apneic set, *slp37*, is shown in Figure 2_12.

Dividing the range of values into a series of steps approximates an identification of apneic episodes. A count is kept of the number of discrete sets of intervals that fall above each step. Since apneic episodes are characterized by a series of oscillations spanning at least five minutes, the interval sets must contain at least 150 beats to be counted as a discrete set. As shown in Figure 2_12, all intervals will fall above a boundary at 0.1, so that the count of sets for this step would be one. For a boundary at 0.2, all of the beats until ~11,650 would count as one set. Another set would be from ~15,150—~16,300, a third set would extend from ~16,350—~24,000, and a fourth set would run from ~24,400 to the end of the set. The changes above the 0.2 boundary that fall within the range ~11,641—~15,200 may not count as discrete sets if they do not contain enough beats. For a boundary of 0.3, more discrete sets would appear in the range of ~5,900—~11,640, while the smaller peaks in the range of ~11,640—~15,200 would be lost altogether.

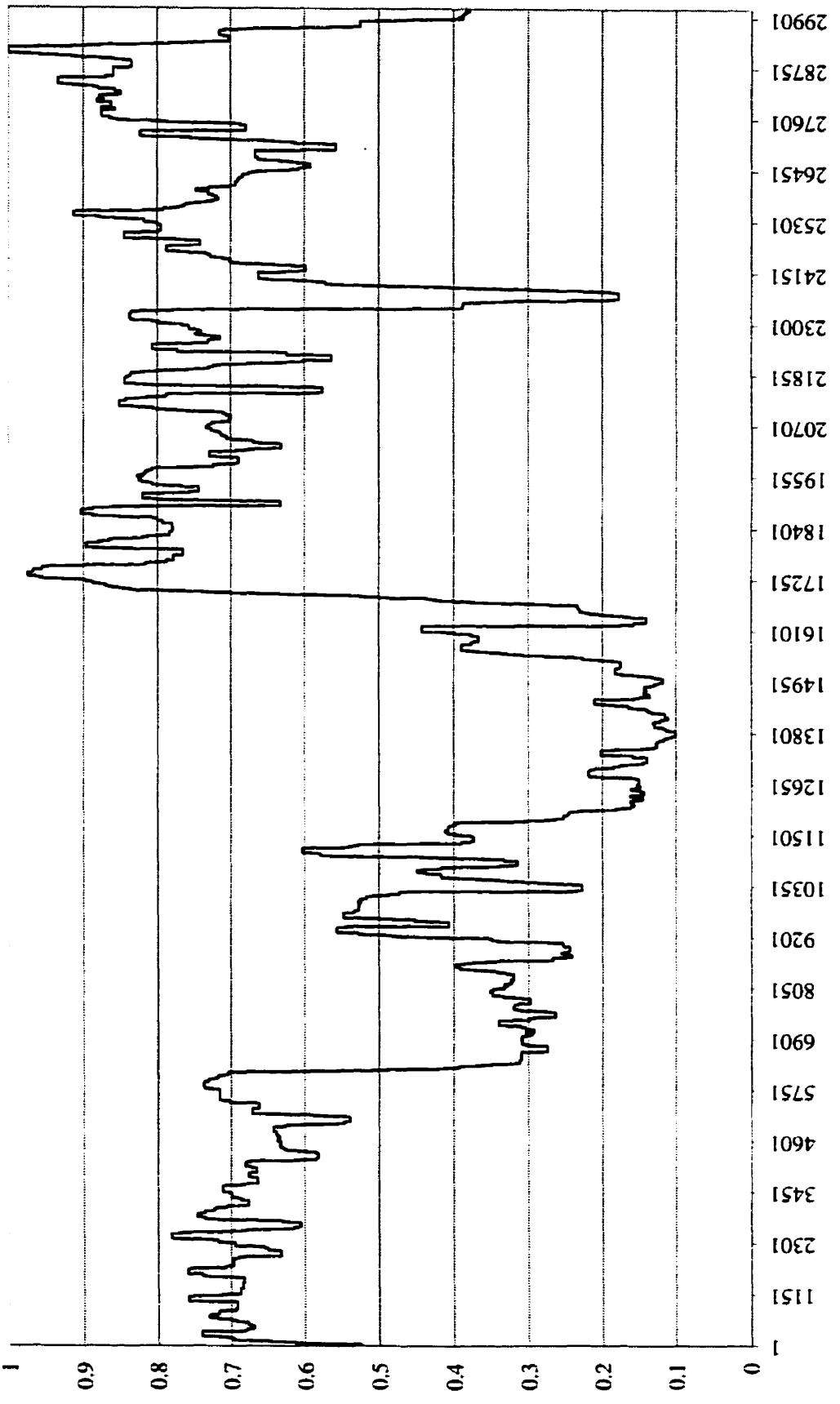


Figure 2_12
Median filtering of CVAA
Data set: slp37

Thus, as the boundary value increases, the large sets of values that exceed it break into smaller peaks, while lower-level values that do not exceed the boundary are lost. Lower-valued boundaries will have a small number of discrete sets, as will higher values. In the figure, a boundary of ~0.7 produces the greatest number of discrete sets.

A systematic analysis of this nature for a given data set produces a threshold value above which fall the greatest number of discrete sets of data points. The values above this threshold correspond with the oscillations produced by apneic episodes, and show significant overlap with the regions identified as apneic by the respiratory analysis. Figure 2_13 shows a segment of an NN interval set with the apneic episodes indicated by black triangles, and the crossings of the median filter threshold indicated by a heavy dark line along the top of the graph.

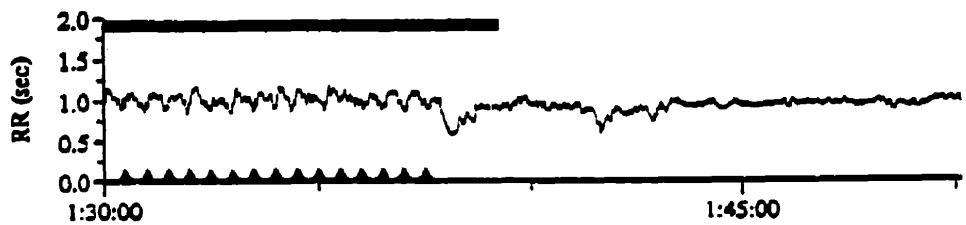


Figure 2_13:
Plot of NN intervals, annotations of apneic episodes and median filter threshold crossings

Factors in this work include the choice of step boundaries, the minimum number of beats above each boundary to count as a discrete set, the size of the filter window, and the number of times to apply the filtering process. The median filtered sets used in the sonification examples described in Chapter 4 are produced with a window size of 201 beats, and the set is filtered two times. The more times the set is filtered, the steeper the slopes in it become. Similarly, shorter window sizes also result in steeper slopes.

This analysis method remains tentative and has yet to be verified by comparison with tests done with healthy data sets to compare their differences. However, since the wavelet-Hilbert transformed data sets of healthy subjects differ markedly from apneic subjects, there is reason to believe that there will be differences reflected in median filtered versions of these sets. Further work in this area is likely to proceed in tandem with auditory display realizations.

3. Choice of Software

3.1 Software Synthesis

A software sound synthesis program (SWSS) realizes the HRV sonification presented in the next chapter. These programs represent the air pressure changes of musical events as a discrete series of numbers or *samples*. While the set of samples for a piece of music may be very large and complex, software synthesis is viable due to the redundancy of musical signals. Composed of periodic waveforms, a soundwave lasting for a span of several minutes need not be specified for this length of time. Rather, a template can be created (a *wavetable*) and its samples referred to repeatedly for the amount of time that the sound source is needed.

Samples are audified by being passed into a digital to audio converter (DAC), which converts the numbers into voltage changes that are used to vibrate the cone of a loudspeaker, thus producing the desired sound. For example, an audio compact disc (CD) contains a set of discrete samples. The CD player contains a DAC that feeds the numbers to an amplifier, hence driving a loudspeaker proportionally to the discrete sample values. A commercial synthesizer also contains wavetables and a DAC to produce its unique set of sounds. Software synthesis enables a composer to create a set of samples so that a composition may be realized and stored digitally.

SWSS systems originated with the work of Max Mathews at Bell Laboratories in the 1950s. His book *The Technology of Computer Music* (Mathews, 1969) is the seminal volume of computer music systems. It is a description of his software *Music V*, the fifth incarnation of a software series commonly referred to as *Music N*. The *Music* series established the conceptual building blocks that remain in place in most music software systems. All synthesis algorithms found on commercial synthesizers were first realized on computers running SWSS systems. Commercial synthesizers simply burn these algorithms onto a microchip. In the early 1980s, these microchip implementations made digital synthesis affordable to large numbers of musicians. With the computing advances of the 1990s, SWSS systems have become implementable on home personal computers, and have thus become popular to a wider population of musicians/programmers.

3.2. Method of Illustration: Unit Generators and Signal Flow Charts

Fundamental to the Music N series was the *unit generator*. A unit generator (or "ugen") is an algorithm that either generates or modifies an audio signal. For example, a primary unit generator is an *oscillator* that produces a periodic waveform. A synthesis instrument consists of a number of interconnected ("patched," a term borrowed from telephony) unit generators. Software synthesis instruments, also called *patches*, are commonly illustrated with flowchart diagrams, as described in numerous sources (Dodge and Jerse, 1995; Moore, 1990; Roads, 1996). Each unit generator has parameters to describe specific characteristics of its operation. An oscillator, for example, is described by its waveform table lookup method, as well as the wave's frequency, phase and amplitude. A sine wave oscillator with a frequency at 440 Hz, phase of 0 and amplitude of 0.5 would be illustrated as in Figure 3_1.

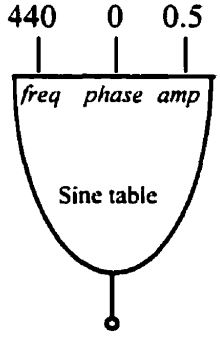


Figure 3_1: Sine oscillator unit generator

The unit generator is conventionally represented as half an ellipse, with a description of its function printed within it. Its parameters are printed along the top, and function as inputs. A line extending from the bottom of the ellipse figure represents the unit generator's output. A unit generator may have one or more outputs.

Figure 3_1 is a simple example in that all of its parameters are fixed. Complete patches are rarely so static. Connecting them to each other can modify unit generator output signals, so that the output of one can be directed to an input parameter of one or more other unit generators. For example, a patch consisting of two sine oscillators at frequencies of 440 and 880, each with periodic volume

oscillations, can be created by patching another sine oscillator with a very low frequency to their amplitude inputs, as in Figure 3_2.

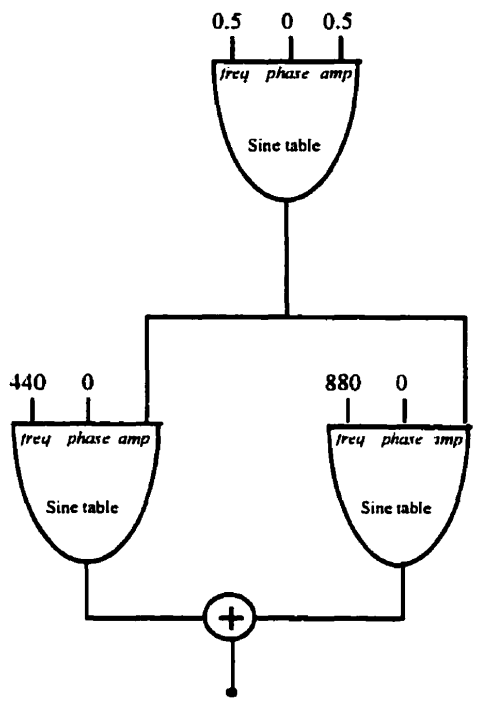


Figure 3_2: Two sine oscillators with a tremolo

Unit generators' output may have arithmetic operations applied to it. In the figure above, the output of both sine oscillators at audible frequencies is combined with an adder into the sound output. Unit generators may be patched in any configuration, with the restriction that no output may be patched to another output.

A sine oscillator is a unit generator type common to virtually all software synthesis programs. In addition to such standard unit generators, software synthesis programs are likely to have specialized unit generators developed for the product.

3.3 Software Synthesis and Real Time Systems

The popularity of SWSS systems on personal computer platforms is due to their capability of running in *real time*. Before the 1990s, SWSS systems were far from being real time. Users had to wait, often for hours, until their code was compiled into files that could be translated into sound, a process that often took

place at another facility altogether. In general, a real time computer system is one in which processing activity must respond to external stimuli within a specified delay time. Systems requiring real-time processing include laboratory monitors, missile guided systems, telecommunications switching or aircraft navigation. Due to the timing requirements of real-time systems and the necessity for a variety of input/output routines with drivers to interface with external devices, real-time systems often blur the distinction between operating system and program. The real time system often works at a level very close to the hardware with only a minimal nucleus of an operating system (Young, 1982). The implication of real time music systems is that they can be used in time-critical situations such as concerts, in which the computer is to respond to a performer's input in order to work in tandem with it, as a player in a duet.

The early versions of the sonifications presented in the next chapter were done in Csound, a descendant of the Music *N* family. More flexibility and sophistication was then obtained in later versions that use SuperCollider, a newer, specialized programming language for real time audio applications. Its effectiveness is due largely to:

- a virtual machine that runs at interrupt level
- dynamic typing
- real time garbage collection
- an object oriented user interface

Each of these features will be discussed in turn.[5]

3.4 Operational Features of SuperCollider

3.4.1 A virtual machine that runs at interrupt level

A *virtual machine* is software that behaves like a processor. The virtual machine is a device module that handles hardware-level operations. A common example is VirtualPC for the Macintosh that makes the PowerPC processor imitate an Intel processor and look to the user like a Windows operating system interface. This is a virtual machine that provides hardware emulation. Another example is the Java

[5]Information on SuperCollider's functionality was obtained through personal correspondence with its creator, James McCartney, as well as through postings from him on the music-dsp internet mailing list.

programming language. Java virtual machines have been written for many computer platforms. With a common user interface for all platform versions of the virtual machine, programmers do not have to modify their code to work on multiple platforms. This allows Java programmers *portability*, a luxury that enables them to write only one version of their programs, yet know that they will run within the Java virtual machine for any platform.

Interrupt level means that a process takes control of the CPU's operations to execute timing-critical functions. Processors are constantly at work updating memory registers, polling active programs for their states, updating the screen, polling input/output (*i/o*) devices, etc. Interrupt level commands suspend the CPU operations until a prioritized action has been carried out. Interrupt level routines may vary in priority level. For example, the highest priority interrupt level command in an operating system is a shut down command, which immediately switches the computer off regardless of what other actions may be in progress.

Interrupt mode allows the CPU to work in tandem with i/o operations, such as storing or retrieving files from disk, or taking input from a keyboard, modem or mouse. Rather than stopping other processes during i/o actions, the i/o sends an interrupt *signal* when it begins. The CPU does what is necessary to initiate the action and then can continue with other tasks until another interrupt is received when the i/o is completed. At that point, the CPU saves the state of its other procedures and performs any necessary completion operations required by the i/o device. When the i/o process is terminated, the CPU returns the system to its former state.

That SuperCollider's virtual machine runs at interrupt level means that its audio routines can preempt CPU operations as necessary to carry out their tasks (within limitations of memory and processor speed). Other features of the virtual machine will be discussed in subsequent sections.

3.4.2 Dynamic typing

Computer data objects exist at a number of conceptual levels[6]. *Simple* data objects have only one value and are *typed* into various categories such as integer, float or character. *Structured* data types hold many values. Two examples are arrays or records. These types, however, are *static* types. The length of an array or record sequence is declared at runtime and cannot be changed subsequently. *Dynamic* types are data objects that may change in size or form during a program's execution. An example is a linked list that may have nodes added or removed during the course of a program. Dynamic data objects are not created by variable declaration but by memory storage procedures. They cannot be referenced directly but must be referenced indirectly by pointers. Dynamic structures change in size or form through pointer operations.

Most SWSS systems are static in nature in that their structure cannot be changed after the program begins running. SuperCollider is interactive in nature in that any component may be altered during playback due to user input (such as pressure on a key, the pitch of a key, the position of a graphical slider), the number of times a function has been called, or anything else. In a sonification model described in the next chapter, a continuous sound event is continuously updated so that its harmonic content and tremolo rate are determined by the current HRV data value. There are only a small number of SWSS systems that offer this level of flexibility.

3.4.3 Real time garbage collection

Garbage collection refers to allocating and de-allocating memory. As a program runs, memory for dynamic objects is taken from a temporary storage area—the *heap*. When these objects are no longer needed they are termed *garbage* and their memory cells may be reclaimed. If memory is not reclaimed often enough the program's operation will be hindered by a shortage of heap space, termed a *memory leak*. Care must be taken, however, to ensure that when objects are reclaimed they are not referenced by any pointers originating from objects still in use. The result will be a *dangling pointer* and problems can arise if new pointers

---

[6]In this and the next section, the term *object* is used generally to refer to any item in memory that is part of a program's computation. A more specialized definition will be introduced in the section on object oriented programming.

are created that point to the same memory cells, most particularly if these pointers are from new objects that are of a different type than the original object. Changes to the new object may bring about unexpected *side effects* in the original object that still points to the same memory cells. Such a condition may lead to unpredictable (and often fatal) problems due to *memory conflicts*.

In languages such as C/C++ or Pascal, the programmer must reclaim memory explicitly with commands such as "free" or "dispose". Languages such as LISP, Smalltalk and SuperCollider, reclaim memory automatically by a hidden process that identifies data objects no longer referenced. There are various garbage collection methods, most requiring significant overhead. Due to the varying sizes of dynamic objects, unpredictable amounts of time (numbers of CPU cycles) may be necessary for the garbage collection routines to be carried out. Thus, automatic garbage collection is problematic in real-time systems, as lengthy garbage collection routines can interfere with time-critical operations, particularly with dynamic data types. However, garbage collection is essential for real time operations of an indeterminate length. For a non-real time environment, the space required to store a sound signal is computed and allocated before any of the computational work begins. For a real time system, the sound signal is produced incrementally. Samples are created for the next time increment, after which they are reclaimed. The requirement is that the time to compute samples for the next increment be less than the time interval spanned by that increment (Dannenberg and Mercer, 1992).

SuperCollider's memory efficiency is due to *incremental garbage collection*, as described by Wilson and Johnstone (Wilson, 1992; Wilson and Johnstone, 1993). Incremental methodologies create garbage collectors that work in small steps between operations of the main program rather than in uninterrupted sweeps. The identification of garbage objects is carried out through pointer traversal from the *root set* that includes global variables, local variables in the activation stack and registers used by active procedures. Any objects that can be reached by pointers descending from the root set are considered *live*. Objects not reachable from the root set are dead to the main program, as they cannot affect future events. They are thus considered "garbage" and may be marked for reclamation. Reclamation may take place immediately or the object's location may be stored in a list that

contains locations of objects to be reclaimed when there is a sufficient percentage of CPU available.

Care must be taken, however, since the structure of a program may change the graph of pointer traversals during the course of its operation between garbage collection increments. If a pointer to an unexamined object is modified so that it originates from an object that has been examined in an earlier increment, the garbage collector needs to be updated. Otherwise the object may be "lost"—subject to reclamation and a dangling pointer.

The incremental identification is conceptually illustrated by a tricolor marking scheme. White objects are those that have not yet been scanned. Grey objects are those that have been reached from the root set, but which have not had all their pointers traversed. Black objects are those that have been reached from the root set, and all of their pointers have been traversed. The problem described above occurs if a pointer to a white object is modified so that it originates from a black object. An incremental updating plan keeps track of changes to black objects' pointers. If any are found to point to white objects then one of the objects is turned grey immediately, which in more practical terms means that one of the objects is placed into the garbage collector's examination queue. This methodology is termed *tricolor invariance*.

The reclamation stage is also optimized for efficiency under Wilson and Johnstone's methodology. It is an improvement on *implicit copying reclamation* in which live objects are copied to another memory region. When all live objects have been identified and moved to a separate area of memory, the original memory may be reclaimed in its entirety without further examination since it implicitly contains only garbage objects. Wilson and Johnstone describe a process of *non-copying implicit reclamation*. Objects are stored into sets that are identified in each objects' header. The sets are kept in doubly linked lists. When an object is found to be live, it can be moved to a second list by reassignment of its pointers, a more efficient procedure than actually moving the object to another memory location. When all live objects have been re-linked to the second list, the first list can be reclaimed in its entirety.

SuperCollider's garbage collector works according to this methodology. Any time anything is allocated, a bit of garbage collection takes place. A running CPU indicator during the program's execution shows that SuperCollider's CPU use is consistently low, even for complicated operations. Much of SuperCollider's elegance lies in its effective solution to the problem of resolving real-time memory needs with the need for garbage collection.

3.4.4 Object oriented paradigm

Programs such as Pascal and C are *imperative* programs, also classified as running under the *procedural paradigm*. Their basis is in modifying storage locations by assigning values to variables. Their operations are carried out via selection, sequencing, iteration, and procedure (function) call. They are characterized by speed and efficient memory usage. As *structured* programming languages, they allow function calls for repeated tasks.

The *object oriented paradigm* (OOP) takes structured programming a step further, allowing larger and more complex programs to be created via the creation of specialized modules. These modules can be modified, added or replaced without compromising the overall functionality of the system. The object oriented paradigm is based on real world modeling. Many elements of its functionality are similar to those of procedural languages, but have different terms in an OOP system. Objects are independent and interacting, sending data to each other to modify characteristics or monitor conditions in another object. The use of objects allows decomposition, breaking an operation into its component parts to change resource allocation or distribution.

Object oriented programming is an environment suitable for the complexity of modern programs that may consist of many components and many release versions. It allows code to be highly reusable: components can be (virtually) wired together for the creation of new objects. OOP is also optimal for *graphical user interfaces* (GUIs). A GUI exists as an object that can activate functions within other objects within the system, either to display their status or to modify them. OOP software is also amenable for network management, as interconnected workstations are well represented by interacting software objects.

An *object* is derived from a *class*. A class is an extension of the C *struct* or the Pascal *record*, in which a variety of variables is contained within a preset structure. The data variables contained by a class are known as its *instance variables*. The class extends this idea to include functions, called *methods* in OOP parlance. This enables a class to store various types of information and carry out certain operations. It is a template for the objects that will be used in the program's operation. Each object is an instance of a declared class. As many objects may be created as needed for a particular program.

The use of objects and classes involves three characteristics: encapsulation, inheritance and polymorphism.

Encapsulation refers to hiding the steps by which a class carries out its methods. The program user is not aware of these steps, but simply calls the methods needed to carry out the necessary actions. Activating a function is termed *sending a message* in OOP parlance. Objects typically are method-oriented in that their data is private. The status of its instance variables is generally only modifiable via a method call to that object. Encapsulation renders objects into "black boxes" where, given a certain input, a certain output can be expected without the user needing to worry about how the result is calculated.

Inheritance allows variation on a class via the creation of *subclasses*. A subclass inherits all methods and instance variables from its *parent class*. A subclass may also contain additional instance variables and methods or it may *overwrite* the methods of its parent class. Overwriting involves changing the steps of a method without changing the name of the method. Thus, inherited classes benefit from encapsulation in that the same method call may be used though the inherited class may carry out the method differently.

Overwriting method names is an example of *polymorphism* in which identical calls may activate different types of methods in different types of classes or inherited classes. The names of methods and instance variables may be shared by various class types, allowing encapsulation.

All objects are descendants of a master class called simply Object. This overriding template may have few or no methods and instance variables as it is

simply the basis for subsequent inherited classes. Object is often an *abstract class,* which means that it contains only placeholders for methods that are to be specifically defined in its inherited classes.

As a real-world analogy, consider the components of a computerized orchestra[7]. All members may be descended from a top-level class called *Musician*, which may have methods such as *play*, *stop*, *louder* and *softer*. *Musician* would be an abstract class, as methods are only listed but not defined, leaving the actual methodology to be filled in by subclasses. Subclasses of *Musician* might include *String*, *Wind* and *Percussion*. The *play* method could be written for each of these classes so that string players would use the bow, wind players would blow and percussion players would strike an object. The *stop* method would cause them to cease the playing activity. Each of these classes may also have subclasses. Wind, for example, may have *Brass* and *Woodwind* classes with overwritten methods for blowing to suit these instrument types. These subclasses would also contain new methods to define the articulations for each instrument. Strings would have methods for *play*ing techniques such as *sul ponticello, jeté* and *martello*. Finally, there would be classes corresponding to each instrument that would contain methods to determine the individual characteristics for each.

Object oriented systems are dynamic by nature. Memory is allocated for objects when they are created and reclaimed when objects are destroyed. The binding of variable names to variables is also dynamic, in that any variable name can be assigned to any type of object, and subsequently re-assigned to another object type. The penalty for this dynamic nature is in overhead time, as the system must constantly allocate memory as needed, and check variable types before carrying out operations by a given variable.

SuperCollider makes the best of object oriented and procedural languages. Its virtual machine is written in C so that the hardware interactions are carried out with optimal speed and efficiency. To the programmer, however, SuperCollider's semantics are like Smalltalk as the virtual machine creates an object-oriented language. It is entirely object oriented, with the benefit that all classes have similar functions and operations. All sound modules, for example, can respond to

[7]I am indebted to Zack Settel for this analogy.

the *play* method. New classes of objects can also be created by users, permitting a high degree of customization.

3.5 SuperCollider Syntax

Fundamentals of SuperCollider coding can be appreciated by the functionality of the following fragment (de Campo, 1999):

```
Synth.play( { FSinOsc.ar( 800, 0.1 ) }, 5 )
```

Figure 3_3: SuperCollider code example

The instructions in the above fragment can be summarized as follows:

- An instance of class *Synth* is created, and is passed the *play* message. *Synth* is a *container* for a group of signal generators that execute together.
- Specific instructions on how to carry out *play* are contained in parentheses. To carry out the play method, two instructions, called *arguments*, are provided, enclosed by parentheses. The first, `{FSinOsc.ar(800,0.1)}`, describes the signal generators to be executed; the second argument, the number 5, specifies the duration over which to play.
- The signal generators specified in the first argument are contained within curly braces, *{ }*. These braces create an instance of the class *Function*, which contains a set of instructions to be carried out. Unnamed functions, created "on the fly" in this manner, are called *closures* because they operate as sealed (closed) entities within the overriding environment.
- The first argument to the *play* method is a function (closure) containing *graph* of signal generators.

  A graph is a topological term referring to a collection of nodes (or *vertices*) connected by links called *edges*. Graphs appear in numerous computer science contexts (Standish, 1994). One example might be a transportation network in which each vertex represents a city and each edge represents the distance from one city to another. A *shortest path problem* would investigate the path with the fewest stops or the shortest overall distance between two cities. Trees and linked lists are subsets of graphs. For operations that do not contain cycles, the topological ordering is represented by the edges pointing in a given a direction (output to input)

and no feedback cycles (otherwise it is impossible to establish an order of operations). These types of graphs are called *directed acyclic graphs* (DAGs). An example of a DAG might be vertices that represent university courses with edges from one vertex to another indicating that the first course is a pre-requisite for the second. SWSS systems use DAGs that are illustrated in the unit generator flowcharts shown earlier.

In SuperCollider, the first argument to the play method is a DAG of unit generators, which are created as a function and are thus contained by curly braces.

- In this simple example, the graph contains only one signal generator, an instance of the class *FSinOsc* (Fast Sine Oscillator). The oscillator is passed the method *ar*, which means to generate samples at the CD audio rate of 44,100 samples per second.
- Two arguments to the FSinOsc are contained in parentheses. The first, the number 800, specifies frequency; the second, the number 0.1 specifies amplitude. Different signal generators have different sets of arguments.
- After the function is closed, the second argument to *Synth* is given, directing it to play over a five second duration.

The simple example of Figure 3_3 is meant to introduce important features of SuperCollider's syntax. A more complex sample will be shown at the end of this chapter.

3.6 Other Features of SuperCollider

3.6.1 Graphical User Interface

In keeping with its object oriented environment, SuperCollider allows easy creation of GUIs to control and observe sound playback from the screen. Any parameter of a sound playback system can be associated with a GUI element such as a data slider, number box, checkbox or radio button. In the HRV sonification model presented in the next chapter, a GUI allows various elements of the data set to be adjusted during playback.

3.6.2 Ease of use

In languages such as LISP and Smalltalk, pointers are implicit. Items can be added and removed from dynamic structures such as lists without the added pointer housekeeping required in languages such as C or Pascal. These programs also streamline the compile-run cycle found in these procedural languages. Testing a program simply involves highlighting its code and pressing the ENTER key. The code will then execute immediately. This flexibility allows changes to be made and auditioned with ease. Larger programs can be constructed incrementally by creating each step and verifying its results before integrating it into a larger set of operations.

3.6.3 Spawning events

Figure 3_3 contains one sound event lasting for five seconds. With a methodology original to SuperCollider, a series of events can be *spawned* (generated) through the use of a class that allows the user to specify the type of event to spawn, the frequency with which to spawn events, and a terminating condition for the spawning process. In the HRV sonification model, sound events are spawned for each member of the data set.

3.6.4 Collection classes

SuperCollider allows the creation and manipulation of list and array objects, which are part of the *Collection* class. Collections allow list processing operations. The upcoming code example that demonstrates the effect of randomized phases will create twenty-five odd harmonics of a fundamental frequency by employing the following line of instructions:

```
Array.fill(25, { arg item; (2*item+1)*440 })
```

The code creates an instance of the class *Array* and passes it the *fill* method. The *fill* method is carried out by two arguments: the number of items to go into the *Array*, and the instructions for creating each item. The instructions are in the form of a function that is iterated the number of times specified by the first argument, in this case twenty-five. Each time the function is iterated it is given an argument, *item*, which gives a count of the current iteration numbered zero to twenty-four. This function creates a set of twenty-five odd harmonics to the frequency 440.

The collection classes can be used as arguments to signal generators in a process known as *multi-channel expansion*. If the FSinOsc in Figure 3_3 above had a frequency argument that was an array of two values, for example `FSinOsc.ar( [400, 800], 0.1 )`, the result would have been the creation of two FSinOsc objects. One would produce a sine wave at a frequency of 400 Hz, the other would produce a sine wave at a frequency of 800 Hz. Both oscillators would have amplitudes of 0.1. Each FSinOsc would be sent to a different output channel, left and right on a stereo playback system.

3.6.5 Sample Accurate Scheduling of Events

Many synthesis languages compute samples in groups, called *blocks*. Greater efficiency is gained by computing samples in blocks rather than individually. Computing samples in groups saves the computation time that would be necessary to carry out setup routines for each individual sample. The block size is determined by the *control rate*, a user-settable parameter that determines the update rate of synthesis parameters. In many languages, the block size is constant for the duration of the synthesis operation. Note event times must occur at block boundaries (Dannenberg and Mercer, 1992). For example, a block size of 100 samples means that at standard audio rate, there will be 441 blocks per second. This means that event times are quantized at a resolution of $1/441 \approx 2$ msec.

SuperCollider allows each event to have its own block size. This flexibility allows *sample accurate scheduling*, meaning possible event start times are quantized at the sampling rate. This is particularly important in scheduling many events of extremely short duration. For the sonification model presented here, multiple arrays of data parameters are sonified at a rate determined by the user. The sample accurate quantization of event times means that the information from the arrays will be processed in synchronization, and that the playback rate may be altered on the fly without any form of distortion.

3.7 Another Example: Can the Ear Detect Randomized Phases?

A final example presents a test of the tenet presented in Appendix 1 that the ear is insensitive to the phase of steady state tones.

Three arrays are created by the list processing routine described above. *harmoniclist* is a set of twenty-five odd harmonic partials of the frequency 440. *amplist* takes the order of these odd harmonics and inverts each of them. *phaselist* creates an array of twenty-five random values, all of which fall between 0 and 2π.

A Synth object is created, and passed the *scope* method, that plays and displays the sound wave in oscilloscope fashion. The signal-generating graph consists of an instance of the *SinOsc* class, which takes arguments for frequency, phase and amplitude. With *harmoniclist* and a scaled version of *amplist* as the frequency and amplitude arguments, twenty-five *SinOsc* objects will be created, with corresponding frequencies and amplitudes taken from corresponding members of the two arrays. The first will have a frequency of 440 and an amplitude of 1, the second will have a frequency of 440x3 and an amplitude of 1/3, etc. The phase will be 0 for all *SinOsc* objects. The *Mix* object encloses all of them, mixing their output to one playback channel.

With all phases set to zero, the output will consist of the square wave, the shape of that will be evident as it is shown in the oscilloscope window. Running the code a second time with *phaselist* as the second argument to *SinOsc* will randomize the phases of each *SinOsc* object. On playback, the visual image in the *scope* window will look distinctly different, while the square wave sound will be indistinguishable from the first time the code was run with phases set to zero.

```
(
var length, fundamental, harmoniclist, amplist, phaselist;
length=25;
fundamental=440;
harmoniclist=Array.fill(length, {arg item;
(2*item+1)*fundamental});
amplist=Array.fill(length, {arg item; 1/(2*item+1)});
phaselist=Array.fill(length, {2pi.rand});
Synth.scope( { Mix.ar( SinOsc.ar(harmoniclist, 0, amplist*0.5) )
})
)
```

The coding methodology and structure of SuperCollider environments may have a steeper learning curve than other SWSS packages but the long-term advantages are clear from the above example.

Some SWSS environments are visual, such as the program Max/MSP. In these, users define graphic objects to appear on the screen and connect them with *patchcords*. Users may define an object of a certain type, then connect them by drawing a patchcord from an *outlet* of one object to the *inlet* of another object. The result is a group of interconnected objects, similar to the unit generator flowcharts shown earlier. While these visual environments are more intuitive, they also raise problems. One is the issue of CPU overhead. The CPU usage of Max/MSP is typically far greater than is necessary for SuperCollider, due in part to the additional processing necessary to maintain the screen graphics. The graphical nature of these programs also makes them inherently less flexible. An example such as the one above would be a laborious affair to create as it would involve defining twenty-five sine oscillators with three values connected to each. Furthermore, the SuperCollider patch can be explored by simply changing the values assigned to variables *length* and *fundamental*. Changing one number will affect the subsequent values and signal generators created. In a visual environment, such a change would require further creation or deletion of graphic elements and ensuring that they are patched together properly. In SuperCollider, the patch may be changed with just a few keystrokes.

The next chapter will expand on the techniques covered here to describe the creation of the HRV sonification model.

4. Description of HRV Sonification

4.1 Development of a Heart Rate Variability Sonification Model

The perceptual issues of auditory displays discussed in the Literature Review chapter in the section Elements of Auditory and Visual Displays are the result of a series of sonification models for heart rate variability. This chapter will describe each of these models.

4.1.1 Sonification of Heart Rhythms in Csound

4.1.1.1 Description of Csound Model

The first stage of this work was carried out in 1996, and is described in (Ballora, Pennycook and Glass, 2000). The first decision was the type of software to use. A MIDI implementation seemed too constrained: the basic MIDI specification calls for values within the range of 0-127. Some compromise would have been necessary to map, for example, NN intervals to pitches. The wide range of values in the data set would either have to have been divided into bins, thus losing precision, or significant computational overhead would have been necessary for a procedure such as adding pitch bend values to each MIDI note event. To avoid these compromises, and to gain the flexibility of mapping data values to any synthesis parameter, the SWSS program Csound was used, that is a member of the Music *N* lineage. A quadraphonic file was created, with data values mapped to note-entry time, pitch, amplitude, timbre and localization.

Csound creates sound files by taking information from two text files. One file contains specifications of the synthesis algorithms, grouped as a series of instruments. This is termed the *orchestra file*. The second file contains a list of musical events and a set of wavetable specifications, and is termed the *score file*. In the score file, instructions for each musical event are arranged in columns of information. Figure 4_1 shows the opening lines of the score file for the heart rate variability sonification.

```
f1 0 8192 9 1 1 0   4 .2 0   9 .1 0   12 .1 0   15 .1 0   21 .1 45 ;glassy
f2 0 8192 10 .3 0 0 0 .1 .1 .1 .1 .1 .1 ;fundamental+higher partials
f3 0 8192 9 3 1 0 4 1 0 5 1 0 6 1 0     ; partials 3,4,5,6
f4 0 8192 10 1 0 .3 0 .2 0 .143 0 .111  ;square

;       starttime       sus(delta)
i1      0               0.0101504       np3     ;3,551 time values, divided by 100
i1      0.0101504       0.0107519       .
i1      0.0209023       0.0106767       .
i1      0.031579 0.0106767      .
i1      0.0422557       0.0106015       .
```

Figure 4_1: Sample code from Csound score file

The first four lines are four wavetable descriptions. Following the wavetable descriptions, the columned section describes each musical event. The first column specifies which instrument from the orchestra file is to play the event. The second column specifies the start time for each event, and the third column is a value for duration. Additional columns are optional, and may contain any parameters used by the orchestra synthesis algorithms so that parameters may be modified with each musical event. Wavetables are referred to by number in the orchestra file, corresponding to their number in the score file. The orchestra file may also contain variables that reference values taken from a given column in the score file.

The HRV orchestra file contained four instruments, each of which corresponded to a quadraphonic channel. Due to the complexity and density of the data set, the synthesis algorithm was left as simple as possible to allow the listener to focus on the properties of the data. A single wavetable oscillator performed each channel's sonification. The score file was created with the aid of a spreadsheet. Each data point was multiplied by a fractional amount that determined the playback rate. This amount was arbitrarily chosen as 1/100, so that three thousand data points would play back over approximately thirty seconds. The duration of each event, contained in the third column, was the data point divided by one hundred. The note-entry time of each event in the second column was a running total of each event in the third column, so that each new note began just after the previous note had ended. The values in the third column were used as variables for various synthesis parameters in the orchestra file.

The pitch of each event was derived by multiplying each member of the third column by 100, taking the inverse of each result and multiplying it by 440. Thus, each pitch was centered about Middle A. The amplitude of each event was taken by converting each value to decibels and multiplying it by a constant (3000). The timbre was taken from one of four wavetables, depending on which of four bins the data point was assigned (0-0.8, 0.8-0.95, 0.95-1.1, greater than 1.1). Early

versions of the instrument created a continuous glissando from note to note. A second oscillator was also employed to create either vibrato or tremolo based on the current data value. This was accomplished by the fourth column Figure 4_1, with the annotation `np3`. This is a directive to assign the next event's third column value to an element of the present event. (The period in all rows other than the first specifies that the previous value should be used again, in other words, all events should have an `np3` in the fourth column). This allowed instruments to be created in the score file that specified that the frequency should slide from the value in the third column to the value in the fourth column over the course of each event. Due to the high playback speed, however, none of these changes were audible, so these elements were discarded to avoid unnecessary computational overhead. The directive was left in the score file in case it should prove useful in the future.

Each data point was also assigned to a quadraphonic localization, using the Ambisonics algorithm described by Malham and Myatt (1995). Ambisonics is a localization formula that emulates the signal received by a Soundfield microphone, which is actually four microphones in one. Three perpendicular figure-eight microphones form *X*, *Y* and *Z* axes, with an omnidirectional microphone acting as an overall scalar. The Ambisonics algorithm is meant to emulate the four signals recorded by each of these microphones, which, when combined, may be used to create the illusion through interaural intensity differences that a musical event occurs at any specified point around the listener. Localization may be either quadraphonic, placing the listener at the center of a square of four speakers, or octaphonic, placing the listener within a cube of eight speakers. The Csound Ambisonics algorithms described by Malham and Myatt allow each note event to contain a polar angle in radians, with 0° being the direction to the listener's right. With this orientation, quadraphonic speakers at four corners fall at radian positions $\pi/4$, $3\pi/4$, $5\pi/4$ and $7\pi/4$. For instrument one, each data value was multiplied by 0.7854, an approximation of $\pi/4$, so that this instrument's events would "hover" in positions centered about the right front speaker. Since four, not eight, speakers were used, the vertical coordinate was set to zero for all events.

Instruments two through four were all based on this same model, plus some modifications. As each was meant to play from a discrete quadraphonic channel,

the localization multiplication was changed for each of the instruments so that each would be centered about the speakers at the left front, left rear and right rear. To investigate whether there might have been any fractal ordering in the data, successive averagings of the data points were assigned to each channel. Channel 1 played all data points, each multiplied by 100, as described earlier. Channel/instrument 2 was the average of every two values; each value was divided by 50, so that the playback duration would be approximately the same as that of channel/instrument 1. (Since the values were interbeat intervals, and not elapsed time, it was unlikely that half the number of beats occurred over exactly half the time of the full beat set). In the same manner, channel/instrument 3 was an averaging of every four data values, with each value divided by 25; channel/instrument 4 was an averaging of every eight data values, with each value divided by 12.5.

Use of the Ambisonics algorithm is a two-step process. The musical events and their locations are encoded, with the compiled sound file acting as an intermediate data file. This sound file is then imported into a second Csound orchestra file, where decoding equations are performed on each channel. This second instrument creates the final quadraphonic sound file.

4.1.1.2 Flowchart Illustration

A flowchart illustration of the encoding instrument 1 is shown in Figure 4_2. The Csound code for the quadraphonic instrument is in Appendix 4. The two-channel stereo version of the sonification can be heard on the accompanying CD on audio track 1.

4.1.1.3 Evaluation of the Csound Model

This first sonification model showed that a software synthesis program could be used as a spreadsheet, performing calculations on a set of values and displaying them in an auditory graph. The result was an interesting and pleasant listening experience. Further work was needed, however, to create an effective diagnostic tool.

The Csound model contained a number of arbitrary elements. The choice to derive pitches by multiplying each value by 440 was arbitrary since the value of

instr1

Insts. 2, 3 and 4 are identical, except that X-Y values are derived from $\frac{3\pi}{4}$, $\frac{5\pi}{4}$, and $\frac{7\pi}{4}$.

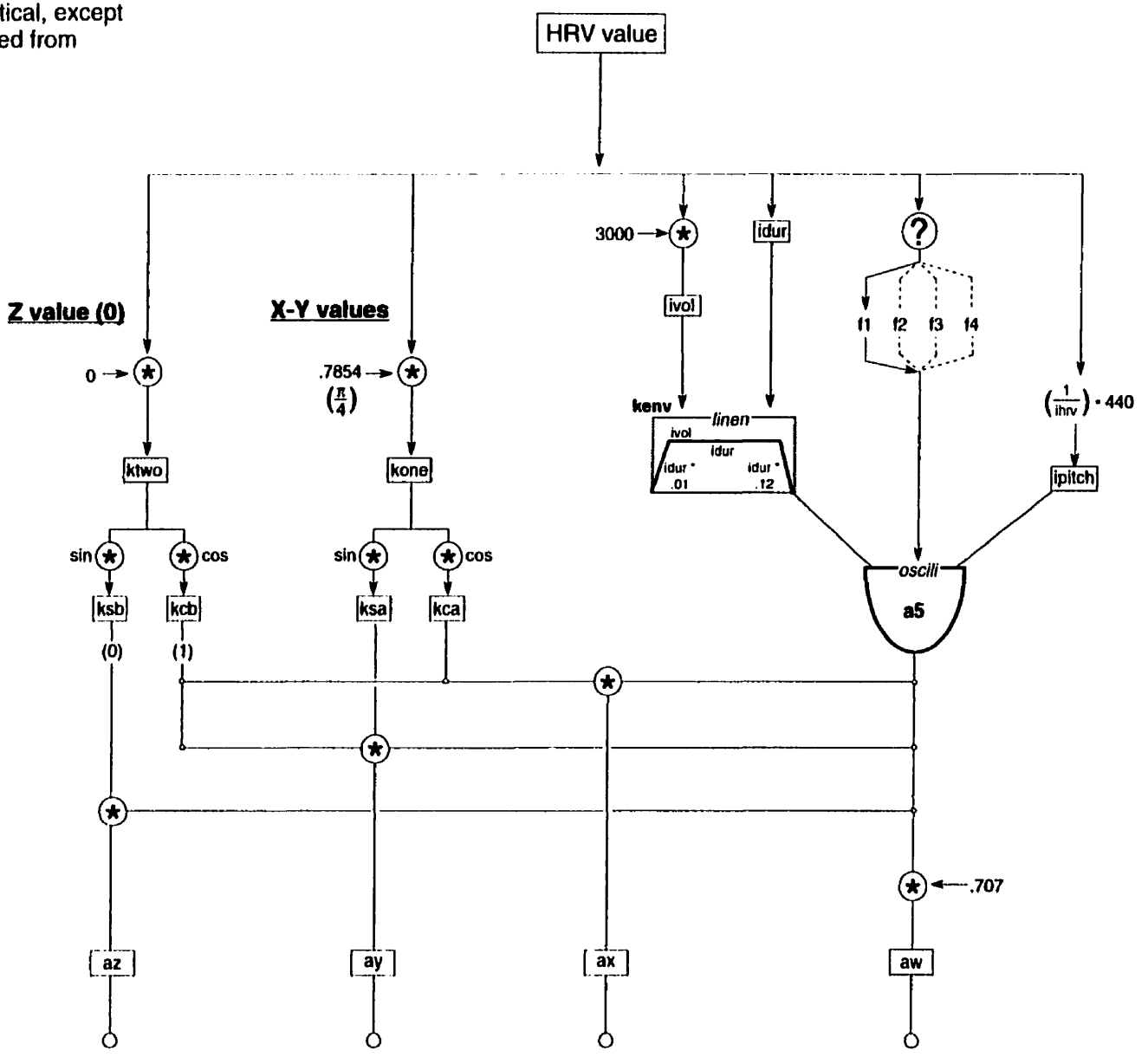


Figure 4_2

440 had no inherent relation to the data, but was simply a convenience due to its function in Western music as a tuning reference. Furthermore, deriving pitches by multiplication creates an uneven distribution of pitches due to the logarithmic nature of the auditory system's perception of pitch, as described in Appendix 1. Taking the inverse of each data point, a value of (1/1.0) will produce the multiplier, 440. A data value of (1/0.5) will produce a frequency of 880, an octave above the multiplier pitch. A change to the same degree in the opposite direction, a data value of (1/1.5), produces a frequency of 293, a perfect fifth below the central value. Thus, equal changes above and below 1.0 do not produce equal pitch intervals above and below the multiplier.

The successive averaging of data points to explore possible fractal relationships also had the shortcoming of being arbitrary. While geometric progressions of this type bring about geometrically fractal images, the fractal nature of data sets, as discussed in the previous chapter, is more often statistical in nature. A statistical fractal analysis is a more complicated procedure, involving either a correlation function or the spectrum of an integrated data set.

The division of data values into four bins, delineated by four timbres, was meant to highlight any possible tendencies of the data to certain value ranges. If data points were predominantly within one of the bin ranges, the timbre would give a coarse approximation of the value. Such distinct delineations, however, run the risk of distorting the data. Given the four arbitrary bin divisions

```
{         - 0.8
    0.8   - 0.95
    0.95  - 1.1
     1.1  -         },
```

a change in data value from 0.95 to 0.96 would produce minimal change in pitch but a change in timbre, while a larger change from 0.81 to 0.94 would bring about a more discernible change in pitch but no timbral change whatsoever. A more effective system would avoid potential mismatches such as this.

Another problem with programs such as Csound, as discussed in the last chapter, is that the structure of a synthesis patch is static, and cannot be changed easily while the sound is being produced. The configuration here is also cumbersome

due to the need for two compilation cycles, the first of which creates the encoded Ambisonics file that must then be recompiled to produce a decoded sound file.

4.1.2 Unit Generators Used in SuperCollider Sonifications

To achieve higher levels of flexibility, the software package SuperCollider was employed in subsequent sonification models that were aimed at improving the diagnostic potential. The following is a description of the SuperCollider unit generators that were used in this sonification.

4.1.2.1 PSinGrain

This unit generator produces a sine wave with an inverted parabolic envelope, as shown in Figure 4_3. The waveform may be described as the equation $(1-x^2)\sin kx$ for some frequency $k2\pi$, within the domain -1 to 1.

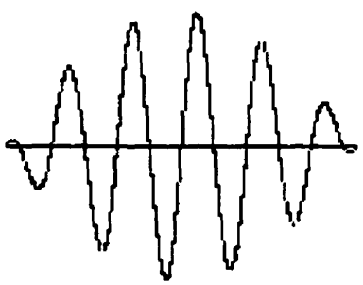


Figure 4_3: PSinGrain waveform

Its parameters are frequency, duration and amplitude. This type of unit generator is effective at creating musical events of very short durations, often termed "grains," in contexts such as the sound clouds created by Xenakis that are described in Chapter 2.

4.1.2.2 Phase Modulator

Phase modulation is a general implementation of *frequency modulation*, described by Chowning (1974). In the early 1970s, Chowning developed synthesis techniques based on frequency modulation (FM). A simple FM configuration involves a pair of sinusoidal oscillators, with one oscillator, the *modulator*, sending its output into the frequency input of the second oscillator, the *carrier*. While frequency modulation had long been in use for radio transmission, in broadcasting the carrier wave is demodulated by the receiving antenna, leaving

the modulating signal to be heard by the listener. Chowning instead focused on the modulated carrier wave. At sub-audio frequencies, the result was a vibrato. As the modulating frequency moved into the audio realm, above 20 Hz or so, the result was a complex set of harmonics, the frequency and respective amplitudes of which could be determined from the modulator:carrier ratio and the amplitude (modulation index) of the modulating oscillator. This was an extremely economical method of synthesis, as only two oscillators were required to create a wide range of timbres. Commercial implementation of FM synthesis led to the widespread adoption of digital synthesis technology in the 1980s.

It has since been reported (Bate, 1990; Holm, 1992; Beauchamp, 1992) that the initial phase of the modulator had a significant effect on the spectral content. In commercial FM implementations, the modulator was given a 90° phase shift, so that in a simple unit generator pair, the carrier was a sine wave and the modulator was a cosine wave. This variant, an example of phase modulation, is implemented in many software synthesis programs[8].

In SuperCollider, the phase modulator unit generator has parameters of carrier frequency, modulator frequency, modulation index, modulator phase and overall amplitude.

4.1.2.3 Wavetable

A *wavetable* is a more general oscillator than the sine oscillator used in the above illustrations. A wavetable contains samples that can describe any waveform; a sine oscillator is one example of a wavetable oscillator.

In SuperCollider, the wavetable unit generator has parameters for the wavetable itself, frequency, phase and amplitude.

---

[8]More generally, phase modulation is based on the definition that a frequency may also be expressed as the derivative of a signal's phase, divided by 2π. The audible effects of frequency modulation may thus be produced in two ways. One way is to modulate the carrier frequency, as described in the text. The other is to integrate a change in phase of the modulator.

4.1.3.4 Band Limited Impulse Oscillator

This unit generator (abbreviated Blip in SuperCollider, and called BUZZ in Music *N* programs) produces a spectrally rich waveform consisting of harmonics of the fundamental frequency, all at equal amplitude, up to the Nyquist frequency (half the sampling rate). In SuperCollider, the parameters for this unit generator are frequency, number of harmonics, and amplitude.

4.1.3.5 Klang

Klang creates a bank of sine oscillators. Its specifications include three arrays that define the frequency of each oscillator, their amplitudes and their phases. This unit generator is highly optimized, making it far more efficient than specifying a group of individual sine oscillators.

4.1.3.6 Pan

Musical events may be localized within a two- or four-channel stereophonic listening space by using a unit generator that employs intensity panning. The first argument to a pan generator is a unit generator graph. The second argument defines the pan position. A position of 0 places the sound center, a position of -1 pans the sound fully left, and a position of 1 pans the sound fully right. As is the case with all SuperCollider objects, any argument may be defined by a unit generator. Thus, a continually moving source can be created by using a Pan unit generator with the position argument defined by a sine oscillator.

4.1.3.7 Envelope Generator

Another unit generator common to virtually all synthesis programs, an envelope generator produces a time-varying change in signal level. Envelope refers to the changes in volume over time in a tone (or one of a tone's partials). An envelope generator typically has parameters for envelope shape, maximum amplitude and duration. The illustration in Figure 4_4 illustrates the envelope shape as a box containing a graph of signal level as a function of time. The figure shows a common four-segment envelope, consisting of attack time, decay time, sustain level and release time segments (also called an ADSR envelope).

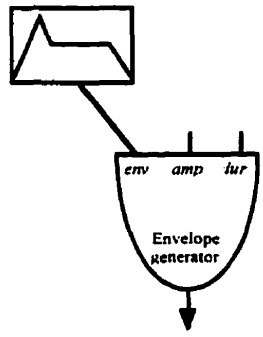


Figure 4_4: Envelope generator with ADSR envelope type

4.1.3 SuperCollider Sonification 1: Cumulative Variation Amplitude Analysis

4.1.3.1 Components of the CVAA Sonification

The first sonifications created in SuperCollider rendered data from four data sets taken from the cumulative variability amplitude analysis (CVAA) described in the Literature Review. The sonification creates mappings taken from the NN intervals, the wavelet-based filtering of the NN intervals, the amplitudes taken of the wavelet values when combined with their Hilbert transform, and the values taken from the median filtering of the Hilbert amplitudes.

The basis of the SuperCollider sonifications is the use of the Spawn unit generator, described in Section 3.6.3. There are a number of unit generator classes that derive from Spawn, including *OrcScore*, which creates musical events in a manner similar to Csound. The first argument, the "orchestra," is a list containing graphs of unit generator functions. Each item in the list is an "instrument," indexed by its position in the list. The second argument, the "score" is a list of lists. The first two arguments of each sublist specify event time and instrument number, followed by optional arguments that may refer to parameters of the instruments. Both the orchestra and the score may be separate files that are read by the SuperCollider patch. Thus, SuperCollider can function exactly as a Csound patch.

It is far more flexible, however, to read in each data file separately and treat them as list variables. The main Spawn class can then be employed. All Spawn classes contain an automatic incrementer that keeps track of how many events have been spawned. This increment value can be used as an index value that increments

through the data lists and spawns musical events based on the value contained in the list at the position corresponding to the value of the incrementer. Flexibility is also gained by having the playback speed set as a global variable that the Spawn object uses to determine the timing of successive events. With this methodology, different data sets can be easily added or removed from the patch without the need to assemble "score" files every time a change is needed. One advantage is that the playback speed can be altered by adjusting the global variable, without any need for making adjustments to a "score" file. This may be described as a "multi-track" approach, with each data set representing a track that may be added or removed from the overall "mix."

To obtain a better relationship between data values and pitch than was obtained in the Csound model, the data value was used as an exponent. Due to the logarithmic nature of the auditory system's perception of pitch, changes in data of the same magnitude in a positive or negative direction produce pitches at equal musical intervals up or down. This approach, however, still does not solve the problem of data sets that have a wide range of values. The values obtained by the wavelet filtering, for example, are frequently very close to zero, so that it becomes difficult to find a mantissa large enough to bring the resulting frequency into audible range. The solution is to have the data value be an exponent applied to a mantissa of two. This mapping function can then be transposed up any number of octaves by multiplying it by some number that is a power of two. Thus, the pitch mapping function employed can be described as a power-of-two mantissa that is multiplied by two to the power of the inverse of the current data value. Data values at or near zero will produce pitches at a frequency of the mantissa, and data values at equal distances above or below zero will produce pitches at equal intervals above or below the mantissa.

The user can control elements of playback via a GUI, shown in Figure 4_5. The GUI panel is modeled after an audio mixing board, with which each track of a multi-track recording may be controlled individually for elements such as volume, equalization and stereo pan. This panel is meant to allow users their own "mix" of the HRV sonification. Two number boxes at the top of the panel display the current NN interval and median filter values for reference. The eight sliders control the volume of eight simultaneous sonifications, each of which will be described in turn.

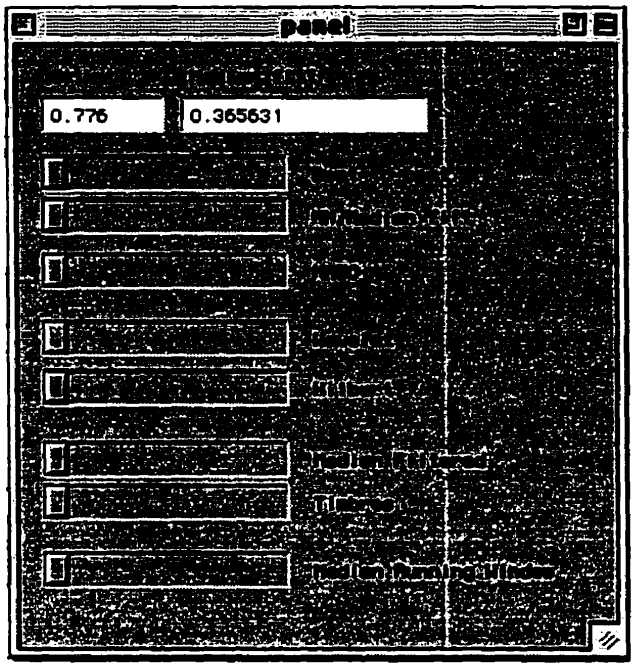


Figure 4_5: GUI for CVAA sonification

Beat-to-beat

This slider controls the volume of a sonification of the NN intervals. A wavetable with a "glassy" sound (reminiscent of the sound created by rubbing a fingertip around the rim of a wineglass) plays a pitch that is associated with each data point. The frequency of each pitch is a function of the current NN interval:

$$f(NN) = 128 \times (2^{\frac{1}{NN}}) \qquad (4\text{-}1)$$

NN/Median filt

A second sonification of each NN interval uses a phase modulator. The carrier frequency is the same pitch mapping as that used for the Beat-to-beat sonification described above with equation (4-1). The modulator frequency is derived from the current median filtered value:

$$f(Med) = 512 \times (2^{\frac{1}{Med}}) \qquad (4\text{-}2)$$

This sonification produces events with the same pitch as the beat-to-beat sonification, but the modulator frequency formula produces a richer, non-harmonic timbre for events that correspond to a higher median filtered value.

NN50

As described in the section on statistical measurement of HRV, the NN50 count is the total number of successive interbeat intervals that differ by more than 50 milliseconds. To give some indication of the occurrence of such intervals as they appear, an additional annotation is given to them in the sonification. As each NN interval is spawned, it is compared to the last. If the difference exceeds an absolute value of 50 ms, the volume of another phase modulator is set to a value proportional to the position of the GUI slider. The carrier frequency is the same as that of the beat-to-beat interval, derived according to equation (4-1). The modulator frequency is this same value, multiplied by 15. The index has a value of 3. The volume envelope is percussive, a decaying exponential curve. The high modulator to carrier ratio and the abrupt attack of the envelope create a "tinkling" sound to identify these beats. If successive beats do not differ by a value greater than 0.05, the volume of the phase modulator is set to zero, and no sonification is produced.

Wavelet

Each value of the wavelet-filtered data set is sonified by a phase modulator. The carrier frequency is derived in the same way that the modulator frequency is derived for NN/Median, according to equation (4-2).

The modulator frequency is the current carrier frequency value multiplied by five, and the value of the index is set to three. The effect is that of a resonant buzzing. The oscillations of this data about zero are further sonified through stereo panning. The phase modulator is placed within a Pan unit generator, and each wavelet data point also functions as the position argument.

Hilbert

The amplitude values derived from the combination of the wavelet-filtered signal with its Hilbert transform are sonified by a square wave. As described in Appendix 1, this wave shape will produce a vaguely clarinet-like timbre. The frequencies used for each pitch are derived by equation (4-2), the same formula as that used to create pitches from the wavelet-filtered data set.

Median Filtered

A running window of the last thirty-two values (corresponding to the size of the wavelet signal) from the median filtered data set is used as the frequency argument to a Klang unit generator. Each pitch is derived according to the formula:

$$f(Med) = 256 \times (2^{\frac{1}{Med}}) \qquad (4\text{-}3)$$

The amplitude argument to the Klang is a linearly decreasing set of values, so that the most recent median filtered value sounds at the greatest amplitude, and the value 32 data points earlier is at the minimum value. The result of this sonification might be described as a "resonant throbbing," the timbre of which becomes brighter with higher-valued data points.

Timbres

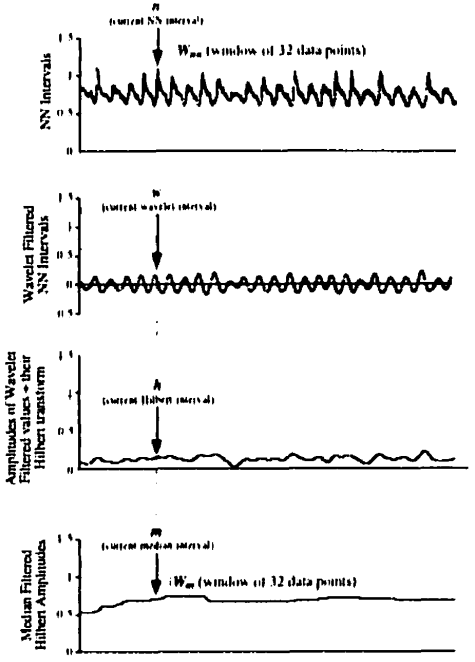
A second sonification of the median filtered data sonifies the current data value according to equation (4-1). The sonification is produced by one of several wavetable oscillators. The wavetable employed depends on the value of the current data point. The range of values is broken into five regions, each of which produces a different timbre when data values are within its sub-range. When data points cross the apnea threshold, as described in the Literature Review, the pitch is transposed up a perfect fifth. When values fall within the highest possible range, the pitch is transposed up an octave. Since the median filtered values remain constant over extended periods, the effect of this sonification is a drone that changes infrequently in timbre and sometimes pitch as well.

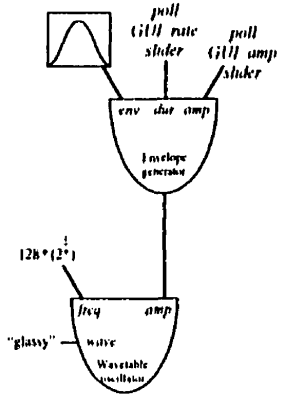
Median Running Window

An "on-the-fly" median filtering is performed with a running window of 32 data points in the NN interval set, with the current interval at the window's mid-point. The median of these values is determined by equation (4-1). The pitch is sonified by a wavetable that produces a sound that might be described as a "hollow ringing."

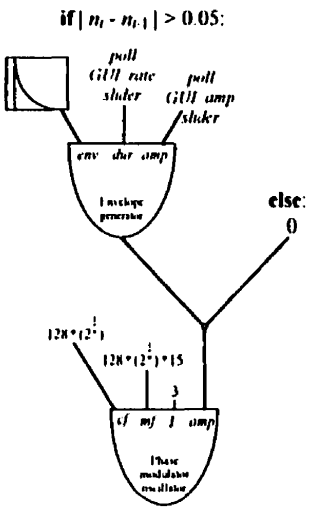
Figure 4_6
HRV Sonification: Cumulative Variation Amplitude Analysis

At each spawned event:
- **Increment *n* by 1**
- **Check whether median filtered value exceeds threshold: *MF > T***
- **Calculate median of *Wnn: Mnn***
- **Poll GUI rate slider to determine time until next Spawn**

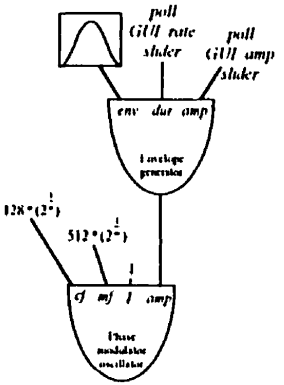




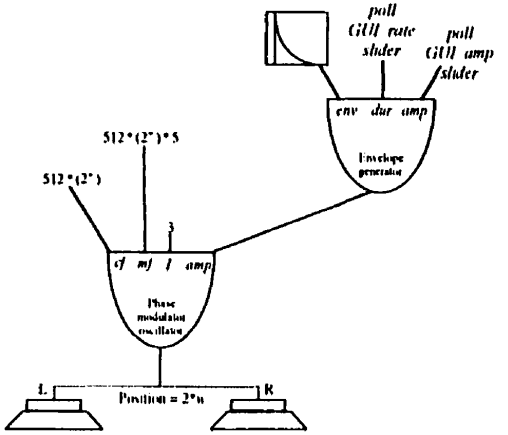
NN Intervals



NN50 Intervals



NN Intervals, phase modulated by Median Filter values



Wavelet values

4.1.3.2 Flowchart, Code and Demonstration

The code for the patch is contained in Appendix 5.1. Figure 4_6 is a flowchart illustration of the CVAA sonification patch. A demonstration of the patch can be run from the CD-ROM portion of the accompanying CD by launching the SCPlay program and running the file `cvaa.lib`.

4.1.3.3 Evaluation of the CVAA Sonification

This second sonification contains significant improvements over the original Csound sonification. The ability of SuperCollider to combine list iteration with the spawning of musical events opens up a far greater range of flexibility. Its operation of simply highlighting data and pressing ENTER allows quick evaluation and easy changes to parameters such as playback speed. The mapping of pitches on a logarithmic basis is also much more workable as it can accommodate a wide range of values, positive or negative. This patch also contains a number of interesting synthesis algorithms, providing a compelling electroacoustic listening environment.

The diagnostic value of this patch, however, is far from certain. While listening to successive stages of the CVAA process may have some pedagogical interest in letting listeners appreciate the similarities and differences in each step, bringing up all sliders at the same time produces a sound mass of such complexity that it would take some time (if ever) for any listener to learn to differentiate among all of its aspects. Furthermore, the CVAA itself is a speculative process that is meant to illuminate a very specific set of properties about an HRV data set. Sonifying each of its steps does not provide any immediate insights, although as research in this direction continues, more value may be found in modifications of this sonification.

The objective of the next step was to employ methodologies gained in this second model towards the construction of a more general model of heart rate variability sonification. The number of elements sonified was pared down, and their nature was simpler, involving more straightforward calculations. This general model is designed to allow listeners to be able to differentiate among four cardiological diagnoses: healthy, congestive heart failure, atrial fibrillation and obstructive sleep apnea. Once this basic level of differentiation is attained, the model may be

appended to represent whatever complementary data manipulations may appear useful.

4.1.4 SuperCollider Sonification 2: A General Model

4.1.4.1 Components of the Sonification

Since it is far from clear what an optimal playback rate might be, the general model allows the playback rate to be adjusted while the sonification is being carried out. Rather than using a single global variable to determine the playback rate, as in the previous example, a slider is added to the GUI. This slider is polled with each spawned event, and its position is used to determine the elapsed time after which the next event is to be spawned. The number of beats to be sonified per second may be set via moving the slider or entering a value into the number box that reflects the slider's value. The GUI for the general model is shown below:

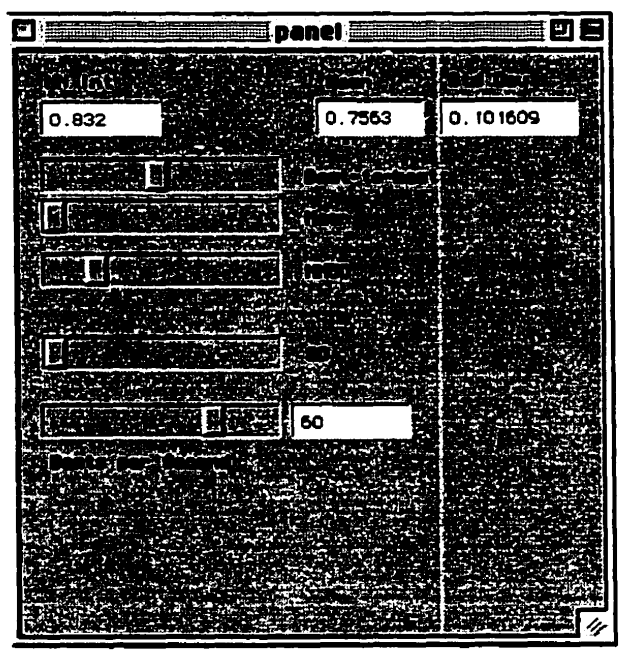


Figure 4_7: GUI for the general model sonification

The sonifications that correspond to each of the five sliders will be discussed separately.

Discrete Events

NN Intervals

The most fundamental element of the sonification remains that of mapping each NN interval to a pitch. The frequency of each pitch is taken from equation (4-1), the same mapping formula as that employed in the previous model. Due to the density and fundamental nature of the data set, this sonification employs a simpler timbre to avoid the possibility of any misrepresentations due to interference of overtones in successive values. In the general model the NN intervals are sonified by a PSinGrain unit generator. The duration of the event is entered by the user via the GUI rate slider. The volume value is also entered by the user, via the amplitude slider. For a visual reference, the current NN interval is displayed in a number box.

NN50 Intervals

This element is unchanged from the CVAA sonification, with a phase modulator unit generator and frequencies derived from equation (4-1).

Continuous Events

The other two data parameters contain data averages. The current NN interval is considered as the center point of a window that contains 300 interbeat intervals, thus corresponding to approximately five minutes of cardiac activity. The mean and standard deviation of this window are determined and updated for each data point to create a sliding window that reflects beat-to-beat changes in local mean and standard deviation. The values are pre-computed in a C program that implements a circular queue. The first 300 data points are read, stored in a linked list, and their mean and standard deviation are taken. The first list member is then discarded, the next data value is added to the end of the list, and the mean and standard deviation are computed again. This process is repeated until all data values have been read from the source file. The C routine saves the mean and standard deviation values into two files that are "SuperCollider-ready" in that they are formatted in the format that SuperCollider reads lists. A list is demarcated by the presence of square brackets, with all list members separated by commas. A pound sign before the open bracket signifies that the list contains literals, rather than variables, which enables SuperCollider to read through it more quickly.

```
#[ 0.147, 0.148, 0.503, ... ]
```

Figure 4_8: Format of a SuperCollider list variable

The mean and standard deviation parameters are not sonified with separate events for each beat, but each is reflected by an event that lasts for the duration of the sonification, and is updated with each spawned event.

Mean Value

The mean value of the window is used as the variable in the same function that determined the pitch of the NN intervals and NN50 intervals, equation (4-1). The result is used as the frequency input to a wavetable oscillator. The wavetable is the set of harmonics with a "glassy" sound, used in the previous sonification. The amplitude value is entered by the user via a GUI slider. For a visual reference, the current mean value is displayed in a number box.

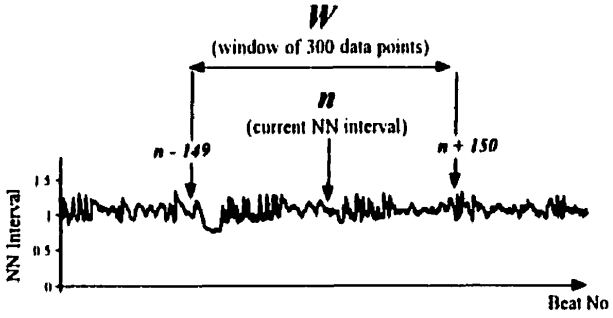
Standard Deviation Value

The standard deviation of the window is sonified by a band limited impulse oscillator. The frequency input is the frequency value derived from the mean of the window by equation (4-1). The number of harmonics is a multiple of the window's standard deviation, with the result that higher standard deviations produce a brighter sound. The volume of the impulse oscillator is a tremolo, controlled by a sine oscillator. The frequency of the sine oscillator is the standard deviation value for the window, with the result that higher standard deviation values produce a faster tremolo. The amplitude of this modulating oscillator, the overall amplitude of the standard deviation sonification, is the value is entered by the user via a GUI slider. For a visual reference, the current standard deviation value is displayed in a number box.

4.1.3.2 Flowchart Illustration, Code and Demonstrations

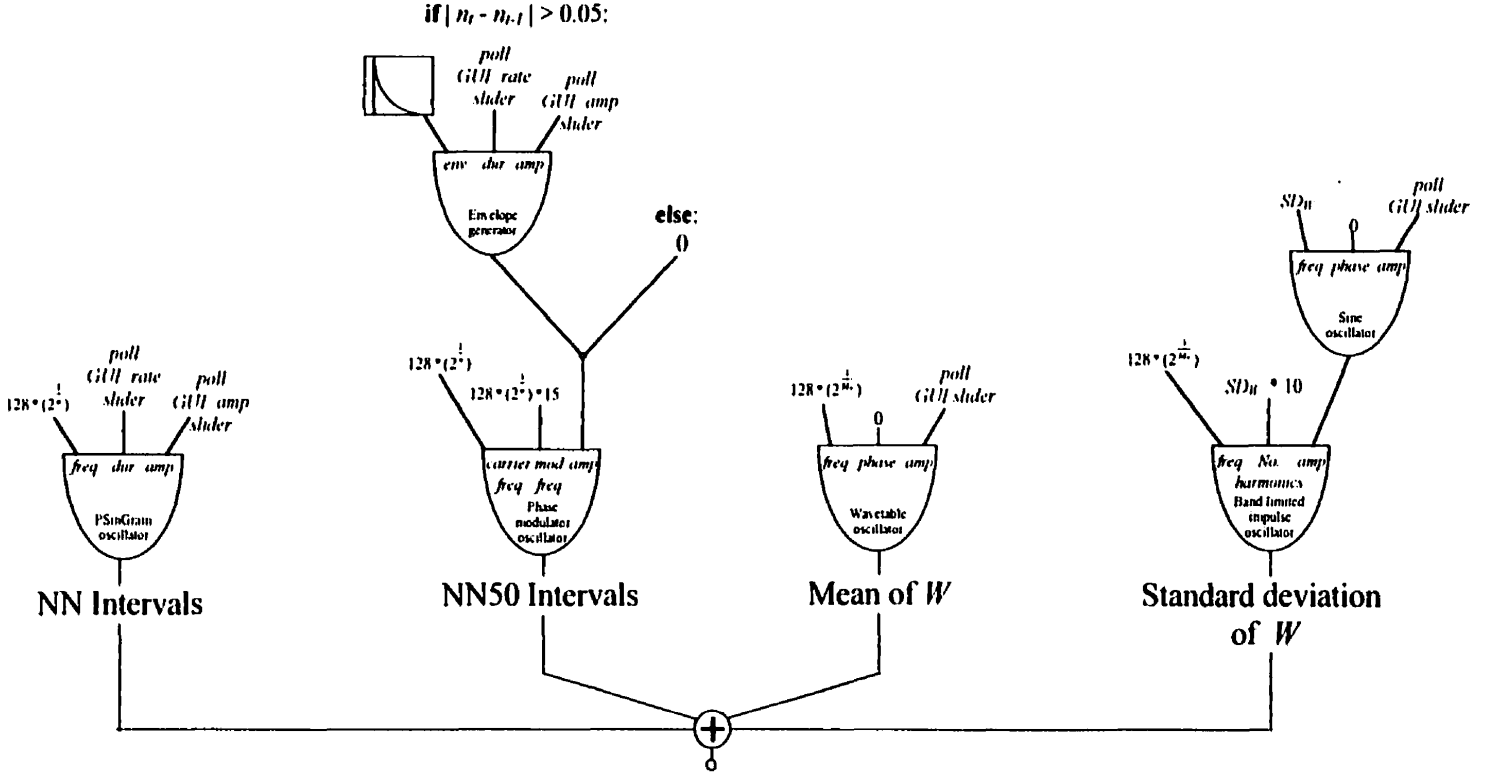
The code for the patch is contained in Appendix 5.2. A flowchart illustration of the processes described above is shown in Figure 4_9. Demonstration patches of healthy, congestive heart failure, atrial fibrillation and obstructive sleep apnea may be run from the CD-ROM portion of the accompanying CD by launching the SCPlay program and running the files `GenModel_Healthy.lib`, `GenModel_CHF.lib`, `GenModel_AtFib.lib` and `GenModel_Apnea.lib`.

Figure 4_9
HRV Sonification: General Model



At each spawned event:

- **Increment *n* by 1**
- **Calculate mean of *W*: M_W**
- **Calculate standard deviation of *W*: SD_W**
- **Check whether median filtered value exceeds threshold: $MF > T$**
- **Poll GUI rate slider to determine time until next Spawn**



4.1.3.3 Evaluation of the General Model

The general model offers a feasible basis to make auditory distinctions among different cardiac conditions. The four parameters are distinct enough to be perceived separately, yet they all blend to the degree that hearing to the four of them simultaneously may be described as an intelligible and pleasant listening experience. The NN intervals form the data set with the most variability. Sonifying them with a sinusoidal unit generator, which produces a simple timbre, allows the variability to be perceived without excessive high overtones creating a grating or irritating sensation. The NN50 intervals are sonified with the same pitch as the NN intervals, yet sound distinct from them due to the presence of higher harmonics and the percussive envelope. The glassy tone assigned to the running mean, being based on the same pitch formula, will always be similar in pitch to the current NN interval sonification. Timbrally, the glassy tone is sine-like, but the presence of higher harmonics, plus the continuous nature of this sonification, allows this tone to blend easily with the NN intervals while still remaining distinct from them.

The standard deviation tone was more difficult to map to pitch, due to the different scale from the mean and NN interval values, as well as the range. The standard deviation may differ by more than a hundred fold over the course of a data set, making a pitch mapping that follows the same formula as the other two problematic. The standard deviation sonification offered in this model offers a solution to this problem. Its pitch is the same as that of the mean. The tremolo sonification of the deviations is intuitively related to the nature of a standard deviation, which measures a range of values centered about a mean. By using the same pitch as the mean, there is no possibility of confusion on the part of the listener trying to relate both parameters. The volume oscillations are non-intrusive and easily distinguished from the other elements in the sound field. The tremolo rate with this mapping falls into a low frequency range of roughly 0.1 Hz to 8 Hz. While the range is wide, approximately eighty-fold, changes to the standard deviation occur gradually. Updates to the frequency value occur much more quickly, on the order of thirty to sixty times per second, depending on the setting of the GUI rate slider. While there is the possibility that aliasing may occur in the oscillations due to the fact that the tremolo rate is updated several times during the course of one cycle, in practice this has not been a problem. At

this comparatively fast update rate to the tremolo oscillation frequency, any distortions of the tremolo rate are not perceptible. The standard deviation sonification also provides a workable use of timbral changes to convey information. The number of harmonics is proportional to the standard deviation value, just as is the tremolo rate. While the timbre is not the primary cue, the higher harmonics, which are associated with higher tremolo rates, become a reinforcing factor, aiding in the perception of "a higher degree of something."

The use of these four parameters allows distinction among the four cardiological conditions. A healthy set sounds "regularly irregular," with sporadic, but not extreme, fluctuations in all parameters. Periodic changes in the mean and standard deviation are easily perceived, and there are patches of higher variability that produce clusters of NN50 interval sounds.

Congestive heart failure, on the other hand, sounds monotonous, corresponding to a data set with greatly reduced variability. The NN pitches are fairly constant, and the running mean is virtually constant as well. The standard deviation sonification has such a low oscillation rate and such a reduced harmonic content that it is almost lost altogether. The NN50 intervals are virtually nonexistent.

At the opposite extreme is atrial fibrillation, which produces a highly agitated sonification that might be described qualitatively as "everywhere at once." The NN interval sonification is reminiscent of boiling water. The NN50 sounds are constant throughout. Due to the high activity, the mean does not change markedly, but the standard deviation is continually at a high rate.

Obstructive sleep apnea sounds similar to a healthy set, due to the fact that apneic episodes may be sporadic, and not a constant factor. Apneic episodes, however, do become perceptible as oscillations in the NN intervals. While the heart rate speeds up and then normalizes, it does not tend toward a constant rate, presumably due to the constant state of oscillatory flux. The result is that the normalizing of the heart rate is characterized by a high number of NN50 intervals, which are heard in regular "clumps" during apneic episodes.

These perceptions, however, must be verified by untrained listeners before any claims can be made regarding the effectiveness of this model. Having arrived at a

(theoretically) workable sonification model, a perception test was conducted to verify its effectiveness in conveying information.

4.2 Listening Perception Test

4.2.1 Purpose of the Test

The test described here explores the viability of the sonification model outlined in the previous section. Would an auditory display of this type be a valuable tool for cardiologists in making diagnoses? Exploring this type of question is an inherent component of any auditory display presentation. As observed by Kramer in the ICAD white paper prepared for the National Science Foundation (1999):

> Sonification efforts must be carefully evaluated with appropriate user validation studies . . . [t]he absence of such studies in the early days of visualization slowed its acceptance. Without this multidisciplinary approach, the field of sonification will mature slowly or not at all; instead, applications of sonification will be developed occasionally on an *ad hoc* basis, but no theoretical framework guiding effective sonification will result.

To this end, a simple listening recognition test was conducted to provide a starting basis for further study. To explore the issue of how clearly the heart rate variability sonification model presents information, the test addresses two questions:

- Can untrained listeners differentiate auditory displays representing four cardiological diagnoses?
- As a diagnostic tool, are auditory displays of the information as effective as visual displays?

Ideally, such a display would require minimal (or no) training to be comprehensible. Music students typically invest years in musicianship and ear training courses in which recognition of pitch intervals and rhythmic patterns are considered an essential component of their professional competence. While cardiologists must develop acute listening sensitivities to detect heart rate patterns via a stethoscope, the HRV sonification model presented here requires a different type of analytic listening. It certainly would not be feasible to expect cardiologists to undertake music training in order to sensitize themselves to subtle differences in auditory stimuli. Therefore, the information presented by an

auditory display must be evident with only a minimal amount of training time. It is hoped that an ideal auditory display would contain layers of information that become meaningful to experienced listeners, but some benefit must be immediately apparent before any deeper examination can be undertaken.

4.2.2 Method

Thirty-nine undergraduate students in a session of the class "Math and Physiology" consented to participate in this study.

The participants were asked to try to identify cardiological diagnoses presented as two Conditions: Auditory and Visual. A test was prepared with stimuli representing examples of four cardiological diagnoses: healthy, congestive heart failure, atrial fibrillation and obstructive sleep apnea. For the Auditory Condition, four ten-second samples of each diagnosis were prepared consisting of sonifications of the NN intervals and the NN50 intervals, as described in the last section. The sonifications presented sixty NN intervals per second, so that the ten-second samples represented approximately ten minutes of heart rate activity. In addition to these sixteen samples, two examples of each diagnosis were repeated, making a total of twenty-four auditory stimuli. This repetition was done in order to verify that the participants responded similarly to identical stimuli. For the Visual Condition, four visual graphs of each diagnosis were also prepared, plotting 600 NN intervals as a function of beat number. The visual displays illustrated the same interval sets as those presented by the auditory displays. Two examples of each diagnosis were also repeated, just as they were for the Auditory Condition, for a corresponding total of twenty-four visual stimuli.

The participants received approximately ten minutes of training before the test began. The training included a brief introduction to the subject of heart rate variability, the four diagnoses under consideration and the auditory display methods employed. The full text of the ten-minute training session is contained in Appendix 6.1. Following the explanation of each diagnosis, an auditory display example was played. After all four diagnoses had been explained and illustrated the four examples were played again without interruption. To aid in the identification of each diagnosis, participants' attention was directed to their

response sheets, which contained a visual display of each of the four examples for their reference. The response sheets are shown in Appendix 6.2.

The test began with the Auditory Condition, in which the twenty-four stimuli were played without interruption in random order. The four demonstrations and the twenty-four stimuli can be heard on the accompanying CD, audio tracks 2-29. There was a pause of eight seconds between stimuli to allow participants to select one of the four diagnoses and mark the corresponding answer on the response sheet. Following the twenty-fourth stimulus, the response sheets for the Auditory Condition were then collected. Response sheets were then distributed for the Visual Condition.

For the Visual Condition, visual displays that corresponded to each of the auditory displays were presented in a random sequence that was different from the sequence of auditory stimuli. Each was projected onto a screen for ten seconds, with a pause between projections to allow participants to identify each image and mark the corresponding answer on the response sheet. The twenty-four visual displays are shown in Appendix 6.3. The response forms used for the Visual Condition were identical to those used for the Auditory Condition, containing a visual display of each of the four examples for reference.

4.2.3 Results

Figures 4_10 and 4_11 summarize the responses to the Auditory and Visual Conditions, respectively. The graphs present the breakdown of responses to each stimulus, the correct identification, and which stimuli were repeated.

A cursory examination of the response summaries shown in Figures 4_10 and 4_11 reveals that the majority of participants were most often correct in their identification of the four diagnoses, both for the Auditory and the Visual Conditions. Of the eight exact repetitions of the auditory samples, six show a higher number of correct identifications for the repeated display. The increased level of accuracy in identifying the second display suggests that the participants experienced some level of learning during the course of the test. To explore the

Figure 4_10
Auditory Display Response Distribution
39 participants

Response breakdown for each stimulus
(correct identifications shown in **bold**)

Arrows indicate matching stimuli

Stimulus No.	Healthy	Congestive Heart Failure	Atrial Fibrillation	Sleep Apnea
1	1	**35**	1	1
2	9	0	10	**20**
3*	4	0	**32**	2
4	**23**	8	0	8
5	6	1	**29**	3
6	**23**	0	6	10
7	0	**38**	0	1
8*	**26**	0	1	11
9	15	1	1	**22**
10	1	0	**37**	1
11	1	**17**	1	20
12*	**31**	3	1	3
13	6	0	9	**24**
14	0	**38**	0	1
15	14	0	10	**15**
16	7	0	8	**24**
17	**29**	2	3	5
18	**27**	4	7	1
19	1	**36**	2	0
20	7	0	10	**22**
21	7	1	**31**	0
22	4	1	**29**	5
23	17	0	**19**	3
24	1	**37**	1	0

* Responses left blank by one or more participants.

Figure 4_11
Visual Display Response Distribution (unscrambled)
38 participants†

Response breakdown for each stimulus
(correct identifications shown in **bold**)

Arrows indicate matching stimuli

Stimulus No.	Healthy	Congestive Heart Failure	Atrial Fibrillation	Sleep Apnea
1	0	**39**	0	0
2*	8	0	5	**25**
3	0	1	**38**	0
4	**8**	27	0	4
5	0	0	**39**	0
6	**37**	0	1	0
7	0	**39**	0	0
8	**24**	0	0	14
9	16	0	2	**20**
10	0	2	**37**	0
11*	5	**29**	1	3
12	**11**	18	0	10
13*	8	0	29	**1**
14	0	**39**	0	0
15	14	0	6	**19**
16	10	0	27	**2**
17	**25**	0	0	14
18	**31**	0	6	2
19	0	**39**	0	0
20	12	0	2	**25**
21	0	0	**39**	0
22	1	0	**38**	0
23	0	0	**39**	0
24	0	**39**	0	0

† One participant out of the 39 left the room briefly and did not mark responses to the first five displays.

* Responses left blank by one or more participants.

effectiveness of repeated stimuli further, statistical analyses of the responses were performed.

The participants' responses for each stimulus of each Condition were scored with either a 1 for a correct identification or a 0 for an incorrect identification. To assess the reliability of the test, responses to identical items were compared through *t* tests. A *t* test measures the difference between two sets of means, and is used to test a hypothesis about a population. The hypothesis here was that the repeated displays would be identified identically both times they were displayed. The mean number of correct responses to each display of the repeated stimuli was compared. The value of *p* for each pair of stimuli represents the degree of error present in the hypothesis, by giving a percentage of the time the hypothesis will likely prove to be correct. In perception tests of this type, a *p* value greater than .05, indicating that the hypothesis is acceptable more than 5% of the time, is seen to support the hypothesis. When *p* is greater than .05, the result is summarized as *ns* (not significant) to indicate that there is no significant difference in the identifications of identical stimuli. A value of *p* that is less than .05 is seen as a significant difference in the identification. Table 4_1 presents the results of the *t* tests.

Table 4_1
***t* Test Comparison of Same Items**

Items	Auditory	Visual
1 & 14 (CHF)	*ns*	*ns*
4 & 12 (Healthy)	$p < .05$	*ns*
5 & 21 (Atrial Fibrillation)	*ns*	*ns*
8 & 17 (Healthy)	*ns*	*ns*
10 & 23 (Atrial Fibrillation)	$p < .05$	*ns*
13 & 16 (Apnea)	*ns*	*ns*
15 & 20 (Apnea)	$p < .05$	$p < .05$
19 & 24 (CHF)	*ns*	*ns*

Table 4_1 shows that for the Visual Condition, subjects responded differently to items 15 and 20, which represented apnea. This indicates that subjects identified this apnea stimulus more accurately the second time it was presented, suggesting that some learning had taken place. No other differences were found between the responses to identical stimuli in the visual test.

For the Auditory Condition, differences between identical stimuli were found for one healthy stimulus, one atrial fibrillation stimulus and one apnea stimulus. As shown in Figure 4_10, participants identified the healthy and the apnea diagnoses more accurately the second time they heard the stimuli, which also suggests a degree of learning. However, when identifying the atrial fibrillation stimuli, participants provided more accurate responses the first time they listened to the stimulus than the second time. It should be noticed that differences to identical stimuli were found for only three of the eight repeated stimuli, indicating an acceptable level of reliability for the test.

4.2.4 Other Descriptive Statistics

In order to understand better the effectiveness of the displays, a comparison of the total number of correct identifications the participants provided for both Conditions is shown in Figures 4_12 and 4_13. Given that there were 24 stimuli, scores could range from zero to 24. The spread of correct identifications is greater for the Auditory Condition. While the lowest score for the Visual Condition was 13 correct identifications, there were seven scores below 13 for the Auditory Condition. At the high end of the spread, the highest score for the Visual Condition was 21 correct identifications, while there were three scores greater than 21 for the Auditory Condition, including one perfect score. While there are more low scores for the Auditory Condition, there is also a greater proportion of high scores. The median number of correct responses for the Auditory Condition was 18.5. This score is slightly higher than the median score for the Visual Condition. One participant left the room during the first six Visual displays. That participant's score was 13 correct out of 19. If this score is included, the median for the Visual Condition is 17; if it is not, the median is 18.25.

Figure 4_12
Auditory Displays – Correct Answers
39 participants

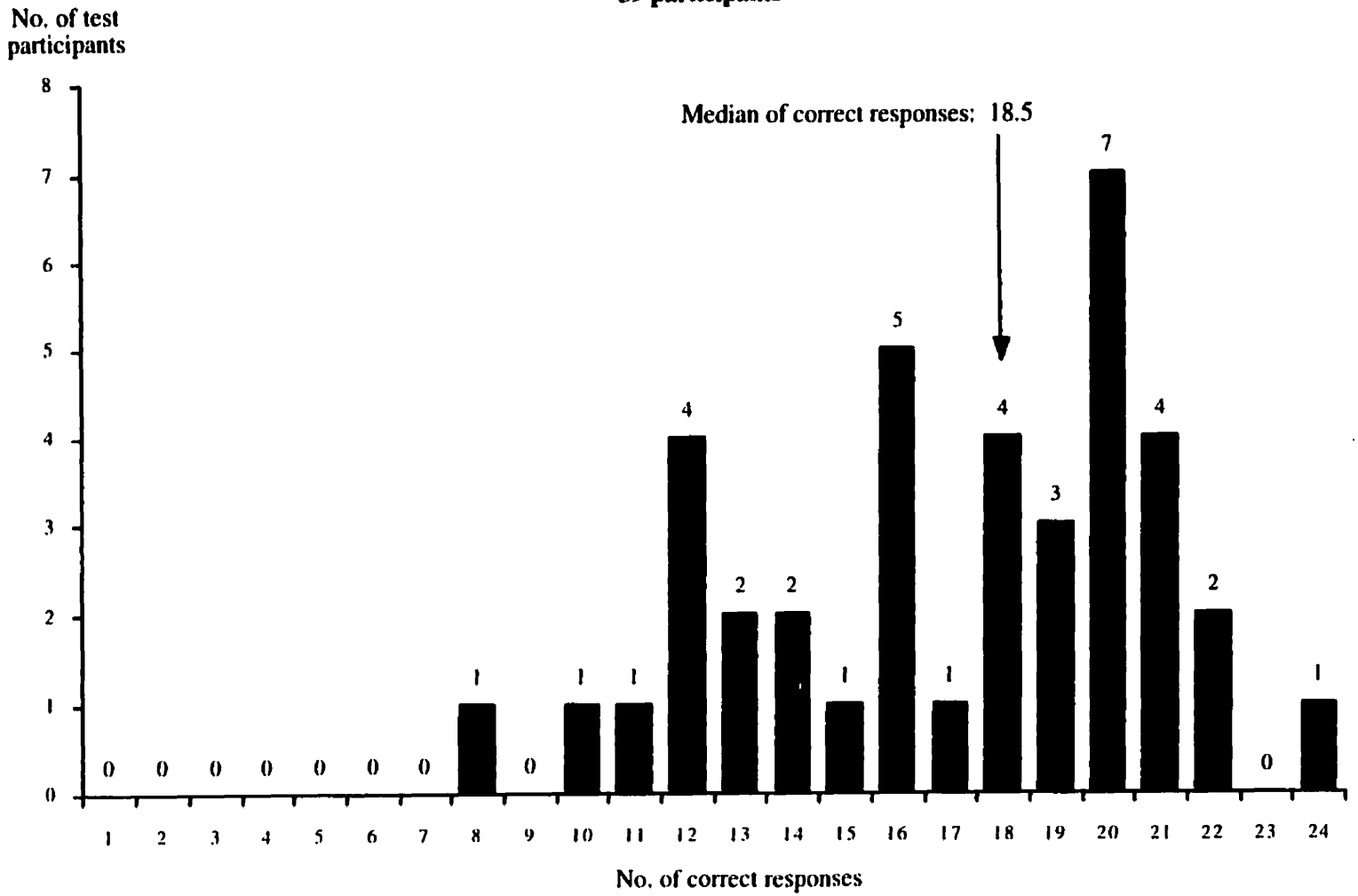
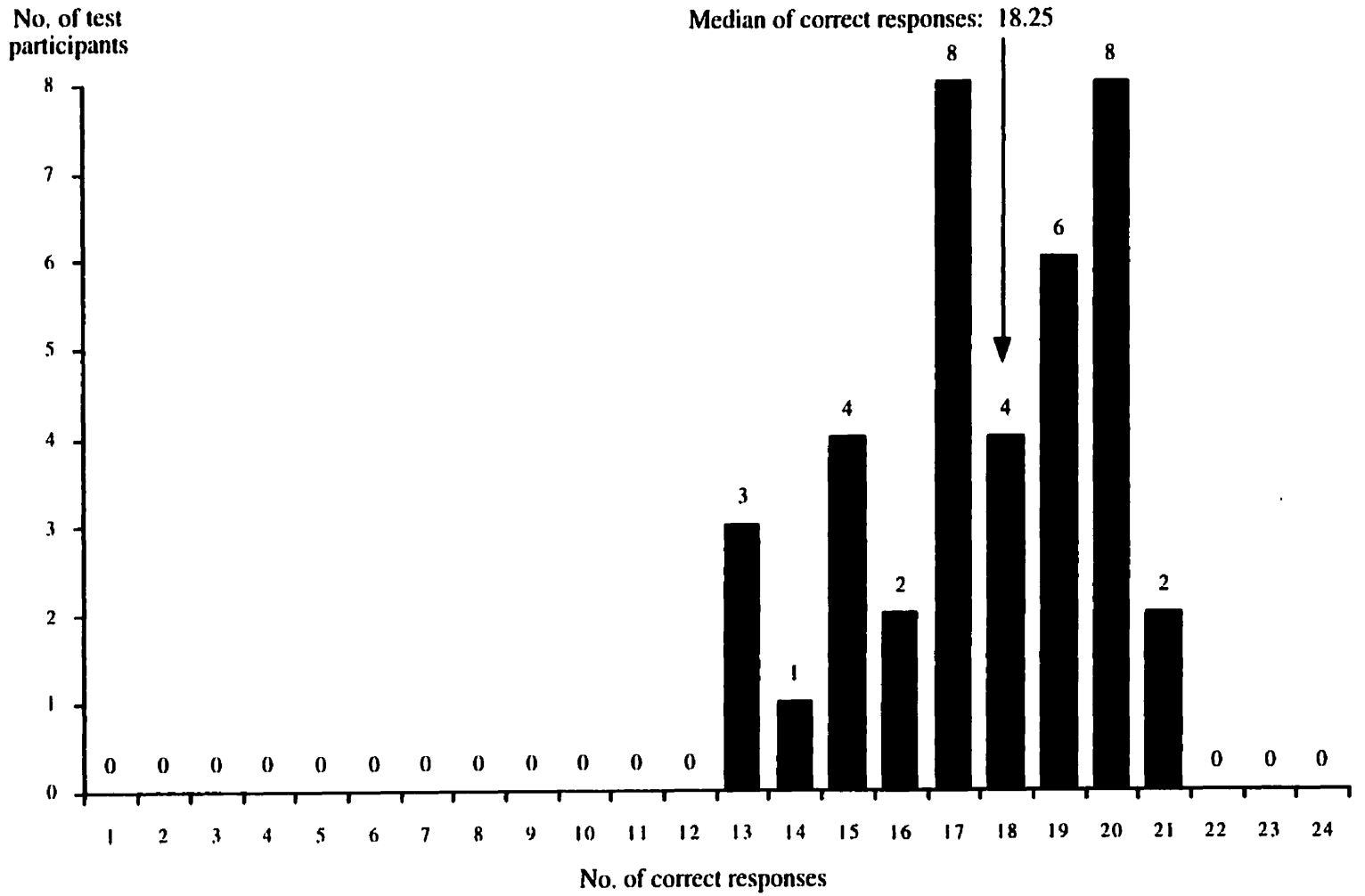


Figure 4_13
Visual Displays – Correct Answers
38 participants*



* One participant left the room briefly and did not mark responses to the first five displays.
This student's score was 13 correct out of 19 responses.
If this participant is included in the total, the median is brought down to 17.

Table 4_2 presents the mean number of correct responses for each diagnosis. The values are determined by totaling the correct responses to the six stimuli presented for each diagnosis, shown in Figures 4_10 and 4_11, and dividing the total by six. Thus, the maximum possible score would have been 39: if all six presentations of a diagnosis were identified correctly by everyone the mean value would be (39*6)/6 = 39. The diagnosis that was most difficult to identify was clearly obstructive sleep apnea. Equally clear is that congestive heart failure and atrial fibrillation were the easiest to identify. This is not surprising since these two diagnoses are characterized by extremely low and high degrees of interbeat variability. Interestingly, the total correct identifications of healthy and obstructive sleep apnea diagnoses had a higher average with Auditory than with Visual stimuli. A more rigorous examination of response breakdowns in the four diagnosis categories is presented below.

Table 4_2
Mean Number of Correct Identifications of Each Diagnosis

	Auditory	Visual
Healthy	26.5	22.6
Congestive Heart Failure	33.5	37.3
Atrial Fibrillation	29.5	38.3
Obstructive Sleep Apnea	21.16	15.3

4.2.5 Results for Each Diagnosis

The main purpose of the experiment was to determine whether presenting subjects with auditory and visual information would yield similar numbers of correct identifications of four cardiological diagnoses. Statistical analyses were performed to examine whether one condition elicited more correct identifications of the diagnoses than did the other.

Each participant's correct responses for each of the four diagnoses were totaled, allowing a maximum score of 6 and a minimum score of 0 for each of the four cardiological diagnoses. Analysis of variance (ANOVA) with repeated measures for each testing Condition showed no significant differences in the identification of diagnoses presented visually or aurally. The results are summarized in Table 4_3.

Table 4_3

Anova table for a 2-factor repeated measures Anova

Source:	df:	Sum of Squares:	Mean Square:	F-test:	P value:
Testing Condition (auditory or visual)	1	.93	.93	.36	.5481
subjects within groups	76	193.41	2.54		
Repeated Measure (four diagnoses)	3	355.45	118.48	122.35	.0001
Testing Condition x diagnosis	3	63.01	21	21.69	.0001
subjects within groups	228	220.79	.97		

However, results of the analysis showed significant differences in the identification of the four cardiological diagnoses and a significant interaction between Testing Condition and diagnosis. This interaction between testing Condition and cardiological diagnoses was explored further through factorial ANOVAs. An ANOVA for each Testing Condition was performed on participants' scores for the four cardiological diagnoses (Tables 4_4, 4_5). For the Auditory Condition, there were significant differences in recognition among the four cardiological diagnoses. Scheffe comparisons ($p < .05$) indicated that congestive heart failure was significantly easier to identify than obstructive sleep apnea, healthy and atrial fibrillation, and that the apnea diagnosis was significantly more difficult to identify than atrial fibrillation and healthy diagnoses.

Table 4_4

Auditory Condition

One Factor ANOVA

Cardiological Diagnoses 1-4

No. Correct Identifications

Analysis of Variance Table

Source:	DF:	Sum Squares:	Mean Square:	F-test:
Between groups	3	72.79	24.26	13.29
Within groups	152	277.44	1.83	p = .0001
Total	155	350.22		

Model II estimate of between component variance = .58

Table 4_5

Auditory Condition

One Factor ANOVA

Cardiological Diagnoses 1-4

No. Correct Identifications

Comparison:	Mean Diff.:	Scheffe F-test:
CHF vs. Apnea	1.87	12.48 *
CHF vs. Atrial Fibrillation	.62	1.35
CHF vs. Healthy	1.08	4.13 *
Apnea vs. Atrial Fibrillation	-1.26	5.62 *
Apnea vs. Healthy	-.79	2.25

* Significant at 95%

For the Visual Condition, there were also significant differences in the recognition of the four diagnoses. Scheffe comparisons yielded similar results to those reported for the Auditory Condition: congestive heart failure was significantly easier to identify than obstructive sleep apnea, healthy and atrial fibrillation, and apnea was significantly more difficult to identify than atrial fibrillation and healthy (Tables 4_6, 4_7).

Table 4_6

Visual Condition

One Factor ANOVA

Cardiological Diagnoses 1-4

No. Correct Identifications

Analysis of Variance Table

Source:	DF:	Sum Squares:	Mean Square:	F-test:
Between groups	3	345.67	115.22	128.05
Within groups	152	136.77	.9	p = .0001
Total	155	482.44		

Model II estimate of between component variance = 2.93

Table 4_7

Visual Condition

One Factor ANOVA

Cardiological Diagnoses 1-4

No. Correct Identifications

Comparison:	Mean Diff.:	Scheffe F-test:
CHF vs. Apnea	3.36	81.5 *
CHF vs. Atrial Fibrillation	-.13	.12
CHF vs. Healthy	2.26	36.78 *
Apnea vs. Atrial Fibrillation	-3.49	87.85 *
Apnea vs. Healthy	-1.1	8.78 *

* Significant at 95%

This interaction between testing Condition and cardiological diagnosis was explored further through factorial ANOVAs. An ANOVA for both testing Conditions was performed on subjects' scores for each of the four cardiological diagnoses. The results of these four ANOVAs indicated that the Visual Condition elicited significantly more accurate identifications for congestive heart failure and atrial fibrillation than did the Auditory Condition (Tables 4_8 and 4_9). On the other hand, the Auditory Condition elicited significantly more accurate identifications of healthy and obstructive sleep apnea diagnoses than did the Visual Condition (Tables 4_10 and 4_11).

Table 4_8
One Factor ANOVA
CHF – No. Correct Identifications

Group:	Count:	Mean:	Std. Dev.:	Std. Error:
Group 1: Auditory	39	5.15	.9	.14
Group 2: Visual	39	5.74	.44	.07

Table 4_9
One Factor ANOVA
Atrial Fibrillation – No. Correct Identifications

Group:	Count:	Mean:	Std. Dev.:	Std. Error:
Group 1: Auditory	39	4.54	1.29	.21
Group 2: Visual	39	5.87	.41	.07

Table 4_10
One Factor ANOVA
Healthy – No. Correct Identifications

Group:	Count:	Mean:	Std. Dev.:	Std. Error:
Group 1: Auditory	39	4.08	1.46	.23
Group 2: Visual	39	3.49	1.12	.18

Table 4_11
One Factor ANOVA
Apnea – No. Correct Identifications

Group:	Count:	Mean:	Std. Dev.:	Std. Error:
Group 1: Auditory	39	3.28	1.64	.26
Group 2: Visual	39	2.38	1.41	.23

4.2.6 Discussion

This test provides a preliminary benchmark to evaluate the effectiveness of auditory vs. visual displays. The results of the test indicate that the participants learned to differentiate among the diagnoses over the course of its duration. While some participants did seem to have difficulties with some of the displays, evidenced during the test by use of erasers and furrowing of brows, some of them expressed afterwards a clear preference for the auditory displays. This informally-stated preference for auditory displays perhaps accounts for the greater number of higher scores in the auditory than in the visual presentations. Some difficulty for the participants is perhaps to be expected given that they were an untrained population being asked to confront the type of diagnostic issues that cardiologists spend years studying. Thus, the percentage of correct responses is compelling, particularly when the responses differed for the Auditory and the Visual Conditions. As we can see from Figures 4_10 and 4_11, the congestive heart failure and the atrial fibrillation diagnoses presented little difficulty in being identified correctly, although the degree of accuracy was higher in the visual presentation. Greater difficulty was present in both presentation modes with the healthy and obstructive sleep apnea diagnoses. Interestingly, the Auditory Condition was more effective in eliciting correct identifications for these particularly difficult stimuli.

Limitations of the test included the number of display elements present in the displays, the education level of the test participants, the training time available, and some elements of presentation. It is possible that a higher degree of accuracy for all diagnoses would result with greater training and with a presentation that included of all four of the display elements described in the last chapter. In the interests of time, only the NN intervals and NN50 intervals were presented in this test. Further information may be gained, however, from the running mean and standard deviation sonifications. A future study could likely yield a greater degree of accuracy if these other two display elements were included.

Conducting further experiments with cardiologists who are more familiar with what the data is illustrating may produce relevant results. It would also be valuable to match the visual and auditory response forms from each participant so that the performance of individual participants under each testing condition may be evaluated. Additionally, it would be desirable to work with more than one group of participants with some being tested with visual stimuli first, and some being tested with the auditory stimuli first. Different response forms could be used for different groups, with one group of forms having the visual references and the second group of forms having no visual references. Revisions could also be made to the visual presentation to ensure that the display times and response times matched those of the auditory displays exactly.

The most compelling outcome of the test is the significant difference in accuracy in identifying obstructive sleep apnea. As discussed in the Literature Review chapter, the pervasiveness of this condition is a concern for many physicians. Current diagnostic methods, however, are problematic in the identification of apnea sufferers. The expense of hospital time and respiratory analysis equipment often makes the diagnosis prohibitive with the result that many sufferers are untreated, posing a possible danger to themselves and others. The superior accuracy of this group of participants in identifying apneic episodes through auditory displays indicates that there is high potential for easy and economic diagnosis of sleep apnea through heart rate variability data that is taken from an ambulatory Holter monitor and mapped to an auditory display.

Given the encouraging results in auditory identification of sleep apnea shown by the test, and the current focus in the cardiology field towards identifying sleep

apnea, the general model was refined to focus specifically on the characteristics of sleep apnea. The aim was to create sonifications that provide quick and unambiguous identification of apneic cardiac pathology.

4.3 SuperCollider Sonification 3: Diagnosis of Sleep Apnea

4.3.1 Modifications to General Model

The general model that was used in the perception tests was easily modified to highlight characteristics of sleep apnea. As described earlier, the heart rate during apneic episodes oscillates over a period of approximately 40 beats. These oscillations can be made audible via modifications to the running mean sonification. The general model used a window of 300 beats for running mean and standard deviation values as this figure represents approximately five minutes of cardiac activity. While this is a useful window for the standard deviation value, which requires larger quantities of values to be meaningful, it can blur the representation of a running mean. As described in the section on median filtering, a mean (lowpass) filter tends to smooth transient activity that occurs within its window. To bring out the oscillations that occur during sleep apnea, the window length was shortened. To use the running mean window to highlight oscillations, the window length must be no larger than half the number of beats per oscillation cycle. Since apneic oscillations occur over forty beats, the window should be no larger than twenty data points. The C program that computes the mean and standard deviation values was modified to prompt the user for a window size to allow experimentation to determine an optimal window length.

An additional level of lowpass filtering was achieved by having SuperCollider round each mean value to a given number of decimal places before computing the pitch values used in the sonification. This has the effect of binning (or quantizing) the values that are represented. Through trial and error, it was found that an effective playback rate was 60 beat values per second, with a window size of 15 and with each mean value rounded to the nearest hundredth. With these settings, the sonification of the running mean produces a distinct oscillation, reminiscent of a siren, during the apneic episodes.

Not all oscillatory patterns are as equally straightforward, however. Oscillation patterns may vary within a given data set. The example shown earlier is

illustrated again below, along with an earlier portion of the same data set. The two segments show different wave shapes, the earlier pattern displaying sharper transitions than the rounded patterns found thirty minutes later.

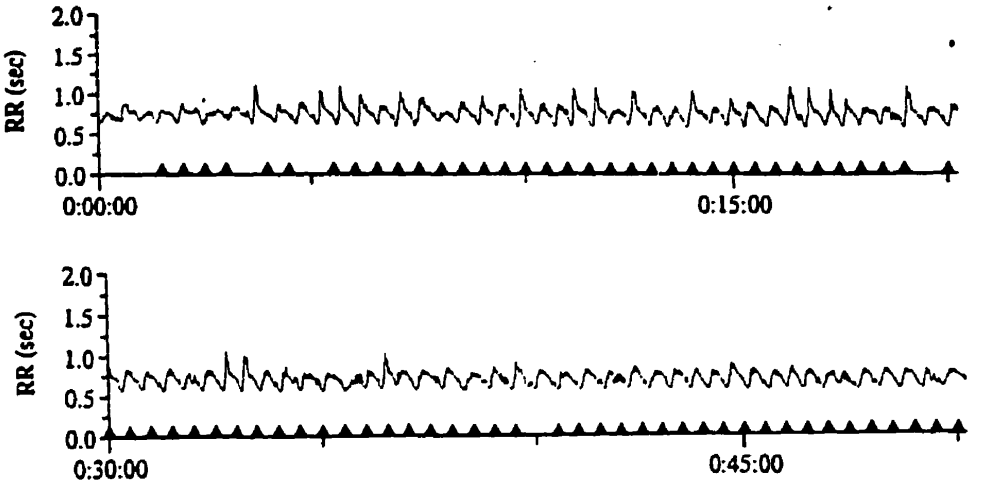


Figure 4_14: Two segments from the same data set, displaying different oscillation patterns

Identifications of apnea can be more problematic with subjects who exhibit more erratic heart rate patterns. The set illustrated in Figure 4_14 was chosen as a base case due to its clear oscillatory behavior that is evident from a quick glance at the visual graph. The illustration in Figure 4_15 is from another data set, in which the apneic oscillations are far less clear. They have been identified by conventional respiratory analyses but are more difficult to detect from a visual representation of the heart rate.

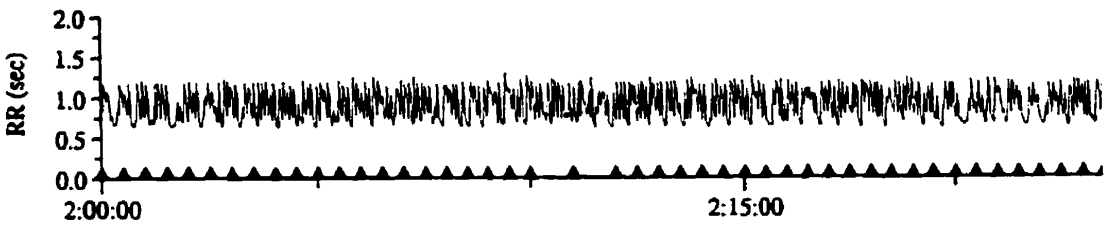


Figure 4_15: A contrasting data set

Figure 4_15 is clearly much more ambiguous in representing oscillatory behavior. The oscillations become clearer through sonification, however, when a second running mean "track" is implemented. In addition to the window of fifteen points, described above, a second set, with a running window of five points, is added. This second window is rounded to the nearest value of 0.21, with the result that its

intervals have a much coarser degree of quantization than the window of fifteen, which is rounded to the nearest 0.01. For many of the apneic episodes, the coarser window sonification simply alternates between two high and low pitches. Other episodes are less binary in nature, with one or two intermediate pitches heard between the higher and lower points of the oscillation. Thus, differing wave shapes may be perceived. More importantly, a more complicated oscillation, such as that shown in Figure 4_15, emerges as an oscillating pattern, though such a pattern is not evident by looking at the illustration. While the original mean sonification contains some oscillation, it is not the manifest siren-call that resulted from the earlier set. The coarser sonification, however, produces a similarly regular up and down alternation. This type of alternation has not resulted when other conditions, such as healthy or congestive heart failure, have been sonified. Thus, an important clue into apneic diagnosis is provided with the use of these two running mean elements.

The two sonification "tracks" complement each other in several ways. The second window is sonified with a square wave, which provides a sufficient blend/distinction balance with the window of fifteen points. The coarser sonification provides a more stable basis for large-scale oscillatory patterns. At a listener-comfort level, it serves to offset a sensation of seasickness that can set in if the constantly oscillating, smooth waves of the finer sonification are heard over extended periods of time. There are also smaller-scale oscillations that are not reflected in the coarser oscillation, but which are accentuated when the finer sonification fluctuates about the unchanging coarser pitch.

This model contains other additions besides the modified mean sonifications. It re-implements the median filtering described earlier as an additional step to the CVAA, and which was sonified in the CVAA sonification model. However, its representation is simplified. Intermediate levels of the median filtering are unimportant; what is significant is to be able to hear when the filtered values have crossed the "apnea threshold" that has been determined for the particular data set. Thus, the sonification only requires that a trigger tone be either off or on, depending on whether or not the threshold has been exceeded.

A steady trigger tone, however, is easily lost among the other sonification parameters when all of the sonification tracks are being heard simultaneously. To

create a sound that appears to "change without changing," six sine oscillators were employed. Three of them produce audible frequencies at 400, 600 and 1100 Hz. While these frequencies share a common fundamental of 100 Hz, the absence of this frequency in the sonification lessens their percept as a single, complex tone. To add variation, the volume of each of the oscillators is controlled by another sine oscillator producing a sub-audio frequency. The effect is a slow tremolo with a different rate for each of the three audible oscillators. The phases of these modulating oscillators differ by amounts that are not multiples of each other with the effect that the amplitudes are constantly in flux and non-periodic in relation to each other. The result is a tone that is constantly oscillating timbrally but does not mask any of the other parameters. The changing nature of the tone keeps it from receding completely into the attentional background. To ensure that the other sonification tracks remain audible, they are increased slightly in volume when the trigger tone is activated by adding a constant to the value represented by each of their sliders. When the trigger tone is de-activated, the constant is removed so that their volumes return to those represented by the slider positions.

The apneic oscillations can also be perceived in the NN intervals. However, the greater complexity of the NN interval set produces a rougher quality that is superimposed on the oscillations. The mean sonification, having been lowpass filtered at two levels, succeeds in creating a smoother, completely unambiguous oscillatory quality that may be quickly recognized even by untrained listeners. (Ary Goldberger, in a rare blend of poetry and cardiology, referred to it as the "siren song of apnea.") The up and down nature of the coarser mean sonification provides a complementary representation that serves to clarify oscillatory behavior, as described above. Listeners accustomed to the sonification, however, may perceive additional information if the other sonification tracks are added. This conclusion was borne out when this apnea model was developed in the company of Plamen Ivanov, a physicist with no formal musical training. In less than two hours, Ivanov was making observations about the sonifications and asking for adjustments in the volume balance to listen to them more closely.

A healthy data set contained intermittent fluctuations, but never produced the consistent siren-like quality of an apneic set. The NN intervals of a healthy set also seemed to sound more turbulent than a set undergoing apneic episodes. With an apneic set, the periodic clustering of the NN50 sonification became evident

during apneic episodes, as described earlier. Congestive heart failure sets sounded consistently flat. Listening to both the running mean and the NN intervals produced beating, since the two were so close in frequency. While CHF sets often undergo mild oscillations due to Cheyne-Stokes respiration, as described in Section 2.3.3.1, these oscillations are distinct from apneic oscillations in both speed and in the absence of NN50 intervals.

The difference between obstructive apnea oscillations and Cheyne-Stokes oscillations is in some ways a moot point as the two conditions are related. Any form of regular oscillatory behavior warrants further examination, whether the cause is central or obstructive. By the same token, intermittent oscillation is normal. There is only evidence of some pathology when there are consistent oscillations for more than three or four distinct periods. Once a consistent oscillatory pattern has been identified, tracing each specific oscillation lessens in importance. Diagnoses of sleep apnea seem to contain some margin of error. We can see in Figure 4_16 below that oscillatory behavior in the heart rate begins near the time of 2:45:00, a full five minutes before the respiratory analysis identifies obstructive sleep apnea. The median filtering extension of the CVAA would cross its threshold at the 2:45:00 mark. Both methods of identification may contain some margin of error. Therefore, the fact that the respiratory and the median filtered identifications may not correlate 100% of the time is relatively unimportant; both methods indicate a high incidence of apneic episodes, warranting some sort of medical intervention.

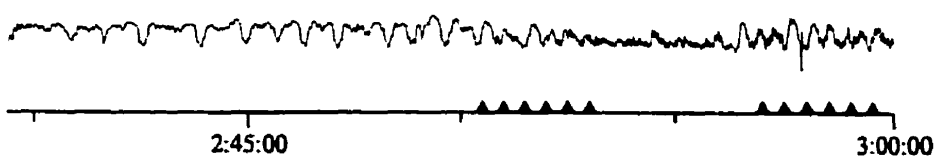


Figure 4_16: Comparison of heart rate oscillations and respiratory identifications of apnea

A further addition to this model was also made to allow a higher level of user interaction. A routine was added to the pre-processing program to keep a

cumulative total of the data values so that its timespan could be tracked. The program created an additional file that was a list that contains elapsed hours, minutes and seconds. This time file is read into the SuperCollider patch, stored as a variable, and is traversed in the same way as are the other external files, adding a time "track" as a component of the sonification. As each musical event is spawned, the index value is also applied to the time list, and the current time is displayed. This feature makes for easier comparisons of the sonifications with visual graphs.

To allow users greater flexibility in listening to selected portions of the data, a checkbox was added to the GUI. Un-checking the box pauses the sonification. Users may then choose a new starting time via a slider. Re-checking the checkbox causes the sonification to resume from the selected point. This feature was accomplished by having the current slider position, rather than the Spawn object's incrementer, function as the global index value to all lists. The time slider is polled periodically at a rate determined by the rate slider. The current position is read and stored into a variable that functions as the global list index value. The time slider is then incremented by one. With uninterrupted playback, the slider acts as a "thermometer" moving gradually from left to right, its positions corresponding to the time value displayed in the number boxes. When the final data point is reached, synthesis stops automatically. The ability to move to desired points in the data was invaluable when testing this model for features such as optimal levels of rounding.

4.3.2 Flowchart Illustration, Code and Demonstration

The GUI of this modified version appears in Figure 4_17. A flowchart illustration of the sleep apnea sonification is shown in Figure 4_18. The SuperCollider code can be seen in Appendix 6.3. Demonstration patches that sonify the data sets corresponding to Figures 4_ 14 and 4_15 may be run from the CD-ROM portion of the accompanying CD by launching the SCPlay program and running the files `ApneaDiagnosis1.lib` and `ApneaDiagnosis2.lib`.

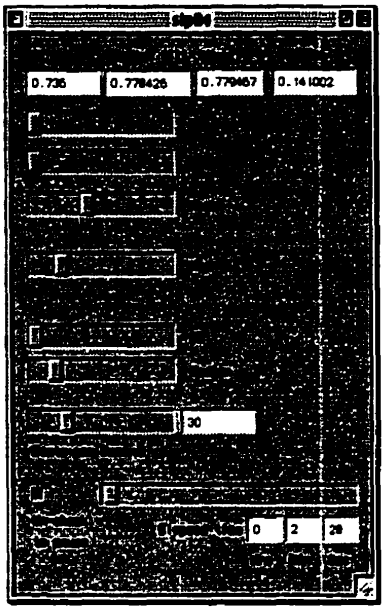
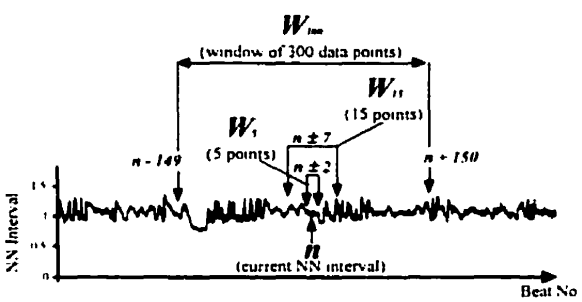


Figure 4_17: GUI of apnea diagnosis sonification model

This sonification realizes the primary goal of this study. A complex data set has had its values mapped to sound parameters in such a way that features of the set that are not evident with a visual illustration can be heard. The identification is not created via a single parameter that could just as easily be represented visually, but through a combination of sonifications of the NN intervals, NN50 intervals and two running means. Adjusting these four tracks depending on the particular data set being sonified allows the characteristics of apneic oscillations to be brought out. It is thus conceivable that this sonification model could prove beneficial to cardiologists in the diagnosis of obstructive sleep apnea. More general conclusions will be discussed in the following chapter.

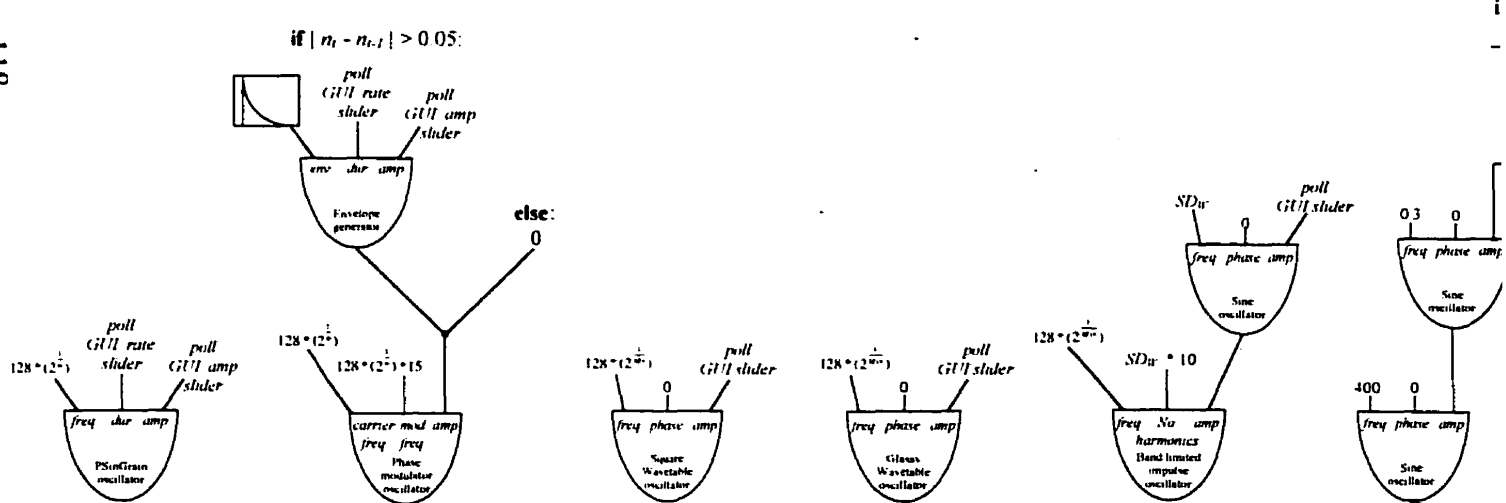
Figure 4_18
Diagram of HRV Sonification for Sleep Apnea Diagnosis



At each spawned event:

- **Poll the time slider, display corresponding time, sonify corresponding**
- **Increment timeslider by 1**
- **Calculate mean of W_5 : M_5**
- **Calculate mean of W_{15} : M_{15}**
- **Calculate standard deviation of W_{300} : SD_W**
- **Check whether median filtered value exceeds threshold: $MF > T$**
- **Poll GUI rate slider to determine time until next Spawn**

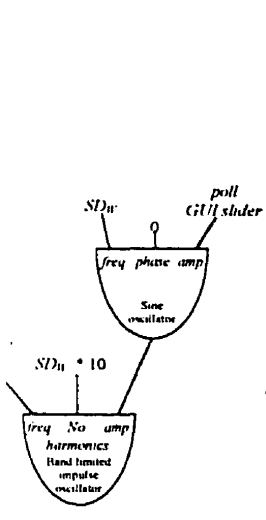
At any time, sonification may be stopped and the timeslider may be adjus



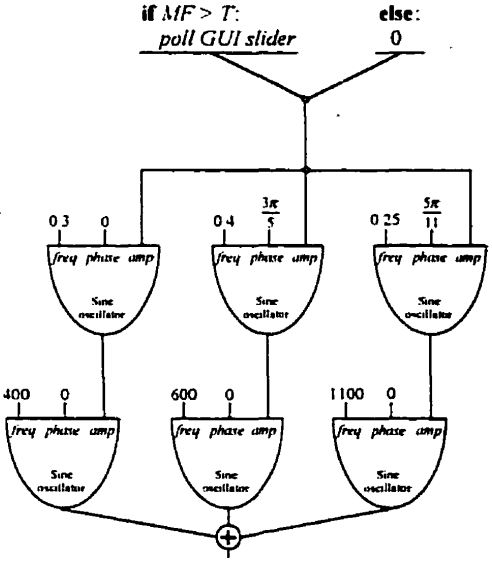
p Apnea Diagnosis

y corresponding time, sonify corresponding data points

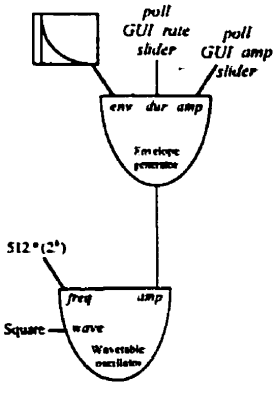
ion of W_{300} : SD_W
tered value exceeds threshold: $MF > T$
ermine time until next Spawn
be stopped and the timeslider may be adjusted.



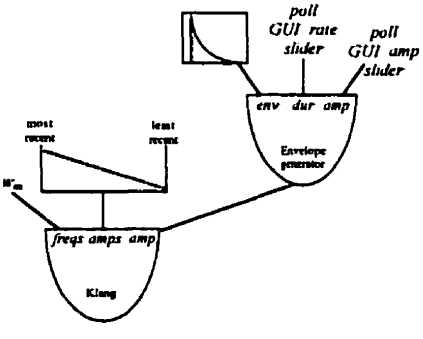
ıdard deviation of W_{300}



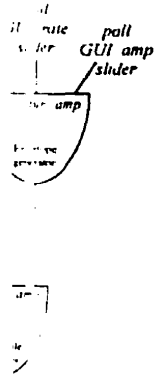
Median filter above threshold



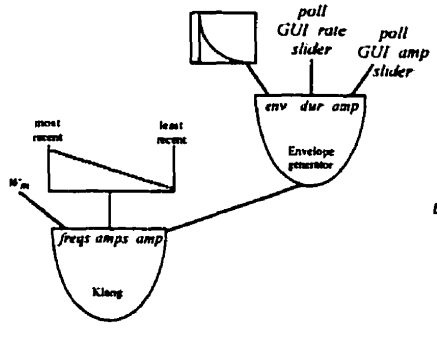
Hilbert Amplitudes



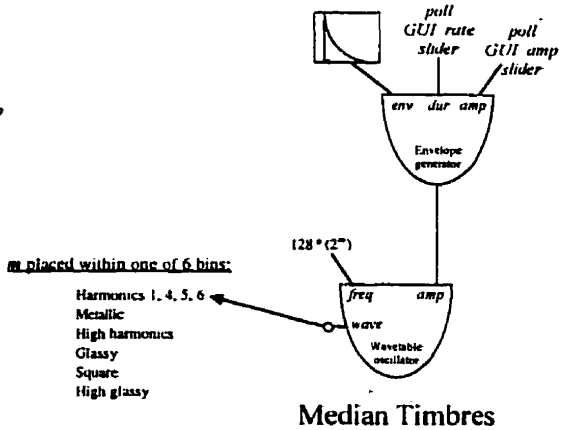
Median Filtered



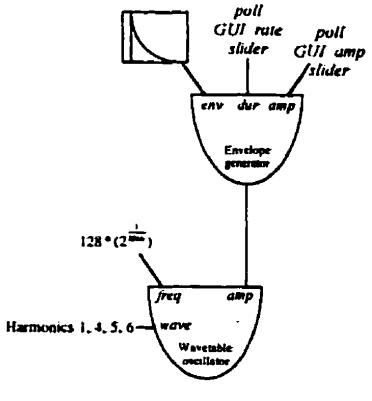
nplitudes



Median Filtered



Median Timbres



Median NN

5. Summary and Conclusions

5.1 Method of Sonification

The methodology behind the sonifications models presented in this thesis is termed the "multi-track" model: the data is presented as a series of simultaneous tracks, each of which represents a different signal processing operation. This methodology is distinct from the methodology taken by Kramer with his stock market analysis sonification, described in Chapter 2. Kramer's approach might be termed the "gestalt" model: multiple data streams are consolidated to represent different aspects of each sound event. The concern with this methodology is that it is questionable whether all of the cues employed function equally well perceptually. Of the five cues Kramer employs—pitch, pulse speed, brightness, detune and attack time—the first two are the most likely to reflect changes effectively on both small and large levels. The last three, as discussed in Appendix 1, are perceptually interrelated. Thus, a change in one, such as attack time, may be confused with another, such as brightness. Similarly, the level of detune perceived may depend on the harmonic content (brightness) of the tone at the time the detune factor changes. Granted, Kramer's intention is not for listeners to perceive these factors individually, but to sense changes based on multi-faceted impressions. But these impressions would be difficult to quantify given the conflicting nature of the factors employed. Such blending of parameters is an attractive option for the creation of music, in which "chimeric" effects, as termed by Bregman, may be obtained through the creative blending of timbre and envelope shapes. Unusual instrument combinations may surprise the ear, producing sounds that are not normally associated with the instruments producing them. For analytical purposes, however, this ambiguity is a detriment.

These comments are not meant to discredit the use of a gestalt display entirely, only to justify why the approach was not used in the models presented here. Whether the sonification methodology is gestalt or multi-track, due consideration must be given to the matter of which sound parameters are meant as *primary* cues and which are meant as *supporting* cues. A primary cue would be one that reflects changes on a moment-to-moment basis in the data; a supporting cue would either enhance a primary cue or provide distinction among various primary cues. In the general model and sleep apnea diagnosis models, pitch and tremolo

rate function as primary cues. Each track is related in pitch to the other tracks, producing a harmonic blend of sounds. The tremolo rate reflects changes in standard deviation. Changes to either of these parameters are easily distinguished. Since the two effects may be perceived simultaneously without being confused with each other, they may be termed orthogonal percepts of the sonification.

Timbre, on the other hand, is a supporting parameter. It is used in conjunction with the tremolo that reflects the standard deviation, adding salience to faster tremolo speeds with higher harmonic content. Timbre is also used to differentiate among the different data tracks. The different data sets are assigned to timbres that are meant to blend amongst themselves, while remaining distinct if the listener focuses on them. Thus, the multi-track sonification employs the "cocktail party effect" discussed in Section 2.2.1 to enable listeners to choose which stream to focus on. Changes in volume are also used as a supporting parameter, lending another level of distinction among the tracks. The volume changes in the median filtered data track give it a slow shimmer effect that keeps it from becoming lost in the sound field. Other tracks are distinguished by the presence or absence of volume changes. The mean and standard deviation tracks are continuous and thus distinct from the NN intervals, which have a "bubbling" effect due to their temporal nature—a sinusoidal envelope for each data point.

While localization was employed in some of the earlier sonification models, it does not appear in the general and sleep apnea models. It is likely, however, that localization would be an effective supporting cue for comparisons of more than one data set. It would be possible, for example, to compare the mean and standard deviation of two data sets by listening to them simultaneously, panned to separate stereo channels. Differences in pitch or tremolo rate would be easily distinguished when separated and heard binaurally. If more than two data sets were to be compared, they could all be panned to different locations to remain autonomous from each other (the same principle employed by Wenzel, described in Chapter 2, in which multiple voices heard over headphones have a higher degree of intelligibility when panned to different locations).

The strength of the multi-track model is in its flexibility. It is comparative in nature, with an open ended structure that allows any number of processing operations to be heard in tandem.

5.2 Auditory Display in Cardiology

The advantage of using a multi-track sonification to analyze complex data sets is that the number of layers is potentially unlimited. It is thus a highly inclusive method of representation, as opposed to many processing techniques that result in certain elements of the original signal being lost. Heart rate variability data sets are highly complex. The analytical methods for heart rate variability discussed in Chapter 2 all attempt to describe factors of the complexity through filtering operations. While the filtering produces valuable results, these signal processing operations by their very nature eliminate other elements. A simple example is in Fourier transforms. Besides the loss of time resolution that results from a Fourier transform, the phase components of HRV spectra are often so complex that they are simply discarded, and focus is instead given to the amplitudes. Thus, all such operations are a tradeoff, in which certain parts of the data set must be deemed expendable. Correlations among these operations are therefore problematic. Even if such correlations are attempted, they are often difficult to display visually, as it is difficult to create visualizations containing more than four dimensions. The auditory system, with its suitability towards multiple stream intelligibility, is the preferable sensory means for comprehending data in higher dimensions.

A tenet of the models presented here is that the original data set always remains intact. Different levels of processing may be applied to it and added or removed from the model at will, but the basis of this processing is always available for comparison. The number of parameters is limited only by the power of the computer platform and the distinctiveness of the synthesis algorithms employed. Listeners may learn to relate processing operations gradually, listening only to a few of them initially, and adding layers to the sonification when it becomes useful to listen to them. Thus, the listening environment may be made as simple or as complex as is desired.

5.3 Future Work

The analyses presented here represent a subset of the methods currently explored in the study of heart rate variability. Given the open-endedness of the model and its suitability for comparing different types of analytical data, different approaches could be consolidated for a more comprehensive sonification model.

Many researchers examine the spectrum of variability over varying timescales. There could be value in mapping a sliding spectral window to a sonification. As was done with the mean in the apnea model, various timescales/window sizes could be represented simultaneously. Other researchers prefer a time-based approach, in which a function is interpolated resulting in peaks at times that correspond to the NN intervals as identified by beat recognition algorithms. While this approach was ruled out for this work, it may be valuable to incorporate research in this direction into future sonifications.

Finally, nonlinear dynamics present a new range of analytical possibilities. As the role of nonlinear analysis in heart rate variability remains speculative, its results did not play a large part in the more recent sonifications. It was decided that more straightforward statistical measurements should be employed and proven effective before incorporating more complex operations. Having established a general model and an apnea diagnostic model, it would be interesting, both artistically and analytically, to focus on the implementation of nonlinear operations through sonification. As was pointed out in Chapter 2, chaos theory remains a compelling and largely uncharted area of music composition. The same can be said about chaos theory in auditory display analysis. The methods explored in heart rate variability could provide the basis for intriguing and informative sonifications.

5.4 General Guidelines for the Creation of Auditory Displays

The work presented here may be summarized by the following general guidelines for the creation of effective sonification models:

- Analysis of the dynamics underlying complex data sets may benefit from a number of signal processing operations that highlight different characteristics of the set. Correlations among these operations may be perceived through a simultaneous auditory display, as the auditory system is well suited for following changes in multiple sound streams.
- To ensure the integrity of signal processing operations in representing aspects of the data, the untreated data set should be included as a basis for comparison.
- A flexible listening environment may allow the listener independent control over each display stream. Control parameters may include relative volume

among streams, rate of playback (number of data points sonified per time unit), and the ability to selectively listen to chosen portions of the data set.

- For a single display stream to be perceived in a complex sound field, some parameter must be changing at any given moment. Constant and unchanging sounds will recede into the background and be difficult to perceive individually.
- Pitch and volume pulse rate are particularly effective as primary cues to reflect moment to moment changes in the data. This effectiveness is due to the auditory system's high sensitivity to changes in periodicity. These two parameters may change separately without interfering with each other's distinctiveness.
- The most effective mapping of data to pitch is to use data points as an exponent. This will ensure that changes in the data are reflected in equal changes of musical pitch interval. The mapping can be performed according to the range of data values in order to control the resulting pitch range. The mapping used in the HRV sonifications is one of many possible methods. The sonification model can easily be altered by changing the mapping equation.
- Overtone content, envelope shape and localization are less effective as primary cues. They are well suited to function as parameters that are not affected by changes in the data, but which may serve as distinguishing factors to allow different streams to remain distinct from each other. The precise settings of these parameters is not trivial, and a good deal of trial and error may be necessary to create a suitable blend of auditory data tracks.

5.5 Concluding Thoughts

When R.T.H. Laënnec invented the stethoscope in 1819, his innovation was not as much in the introduction of a new piece of hardware as it was that he learned to listen through it and make diagnostic judgments based on what he heard. Today, listening training is an essential component of a physician's education. The potential of sonification lies in the fact that it relies on a skill that physicians have already spent a significant amount of time developing: learning to hear diagnostically significant nuances in a changing sound pattern. The only

adjustment is in the type of information that is being presented. Furthermore, the flexibility of sonifications such as those presented here allow physicians to pause and replay segments of a data set at any chosen speed, an ability to "zoom in" on a portion of the data set at will.

The nature of research is incremental. No one project advances any field of knowledge to any great degree without corroborating work done by other researchers. Heart rate variability analysis remains an exploratory avenue of cardiology with few absolute answers to date. Based on the results reported here, it can be stated that auditory display represents a potentially valuable diagnostic component and is a compelling avenue for further development.

Appendix 1
Fundamental Auditory Concepts and Terms

Sound and Time

The origin of a sound event is a disturbance of molecules in the air, which might result from clapping one's hands, plucking a string, blowing into a pipe, striking a membrane, or using electricity to activate the diaphragm of a speaker. The displacement of molecules in the area of this disturbance causes collisions with neighboring molecules, followed by ricochets back towards the original position. The struck molecules in turn collide with their neighboring molecules. Thus a sound wave is a series of compressions and rarefactions of air molecules, traveling outward from the initial point of disturbance. Eventually these oscillations reach our eardrum, which transduces this oscillating motion into mechanical energy and then into electrical current that is interpreted by the auditory system as sound.

The pattern of a sound wave is often plotted as in Figure A1_1, which represents a simple sine wave. The horizontal axis represents time, and the vertical axis represents changes in pressure. The zero point of the vertical axis represents the normal, undisturbed acoustic pressure level.

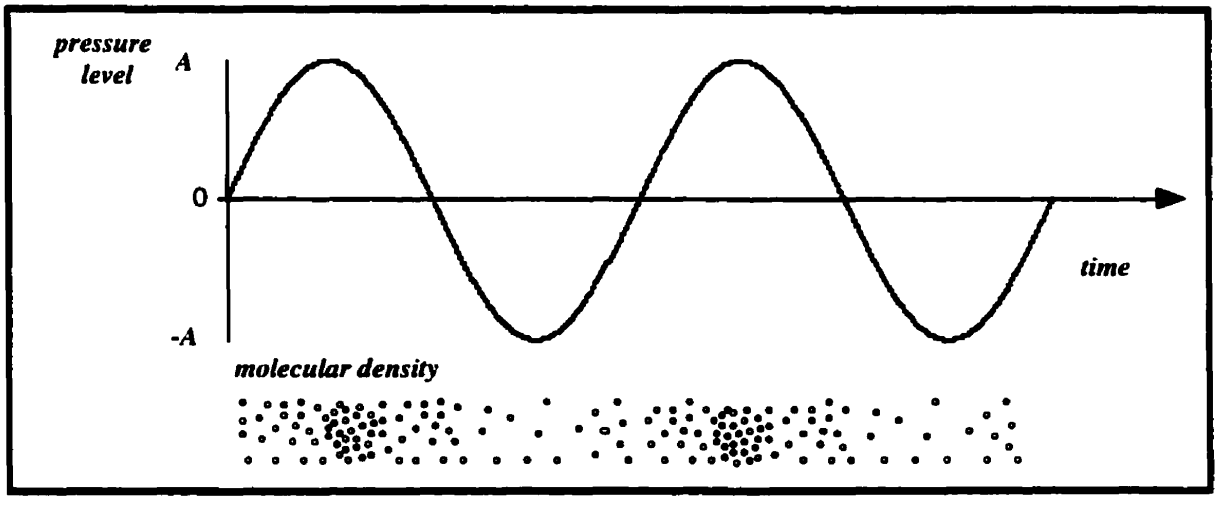


Figure A1_1: Sound wave plot

The back-and-forth motion of these molecules is an example of a *longitudinal wave*. A longitudinal wave is defined by the wave oscillations that move in the same direction that the wave is traveling. The other common wave type is a *transverse* wave, such as that seen in water. A transverse wave is defined by

oscillations that are perpendicular to the direction of the wave's travel. Common to both types, however, is the principle that while the wave moves outward from its starting point, the displaced molecules are not themselves moving along with the wave. They simply collide with neighboring molecules and then reflect back to their original positions. What moves outward is the energy from the initial disturbance. In the case of a sound wave, air molecules are initially at an undisturbed level of pressure. When disturbed, they alternate between pressure levels that are higher and lower than normal. Eventually, the energy from the disturbance diffuses, with the molecules returning to their equilibrium spacing, and the sound ceases. The elapsed time from the initial disturbance to the end of the sound makes up the sound's *duration*.

While the differences between auditory and visual perception are many, the idea of duration is the most significant among them. An image may or may not change over time. A viewer may decide how long to view it, and on which parts of it to focus the attention. Sound, in contrast, exists inherently in time. Sound events have a beginning, middle and end. A sound event can never be perceived simultaneously in its entirety.

Prior to the invention of sound recording technologies in the late nineteenth century, there was no way to control sound events explicitly. It is now commonplace to manipulate sound recordings by changing their speed, playing them in reverse or manipulating the output wave in various ways. Still, the comparatively short history of sound event storage is no doubt part of the reason for the "visual bias" in representing information noted by the International Committee on Auditory Display (Kramer, *et. al.*, 1999).

Pitch

The sine wave plotted in Figure A1_1 is an example of a *periodic* wave. While the majority of natural sounds, such as speech, waves crashing, traffic, etc., produce erratic, non-repeating wave forms, the category of sounds commonly described as *musical* can be quantitatively defined as those wave forms that are repeating, or periodic, in nature.

The musical pitch of a sound is correlated with the frequency of its waveform. The human auditory system is able to perceive pitches roughly in the range of 20 – 20,000 cycles per second. Higher frequencies produce higher pitches. The assignment of frequencies to musical pitches, however, is somewhat arbitrary. While the pitch called middle A is commonly defined the frequency of 440 cycles per second (also referred to as *Hertz*, abbreviated Hz), in reality many orchestras tune to a frequency of 444. An appendix of Helmholtz (1885) contains a table of frequencies assigned to the pitch A in cathedral bells throughout Europe. The lowest "A" is below 400, while the highest is in the range of 480.

When more than two pitches are sounded simultaneously, the blending of tones takes on varying degrees of *consonance* or *dissonance*, depending on the frequency relationship of the two tones. The most fundamental pitch relationship in music is that of frequencies having a 2:1 frequency ratio. Two pitches with this relationship will blend to the degree that they sound very much like a single tone. The similarity is such that musicians will refer to these two pitches as virtually identical, belonging to the same *pitch class*. Western classical music consists of twelve pitch classes. The layout of a piano keyboard is simplified once students learn the repeating pattern of white and black keys, and that corresponding keys of the pattern belong to the same pitch class. The convention in familiar Western music of seven-tone sets of pitch classes, *scales*, means that pitch classes will repeat after each progression of eight scale tones. This 2:1 (or the inverse, 1:2) relationship, then, is commonly referred to as the *octave*. The repetition of pitch classes from octave to octave is the source of psychologist Roger Shepard's illustration of musical tones on a helix (Figure A1_2). The helix is a spiral shape oriented vertically. Moving up the spiral is visualized as moving up in frequency, with a doubling of frequency with each full circle. Thus, a full circle represents the span of an octave, lines adjoining corresponding points along subsequent traversals indicating repetitions of pitch class.

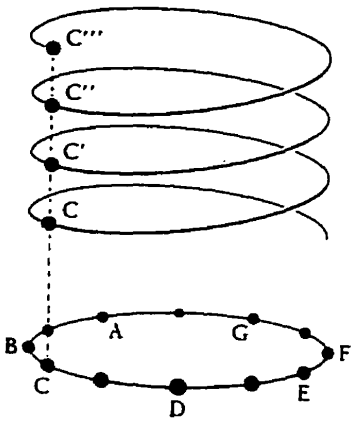


Figure A1_2: Pitch helix
(From:* The Science of Musical Sound *by John R. Pierce
© 1983 by W.H. Freeman and Company. Used with permission.)

The repetition of pitch classes with every doubling of frequency means that the correspondence of musical pitches to frequencies is not linear, but logarithmic. Western music is based on a pitch system in which each tone is equally spaced within the octave. An octave sequence of musical tones, then, starting from a frequency F, can be expressed mathematically as

$$F * 2^{\frac{n}{12}} \qquad n = 0, 1, 2, \ldots, 12$$

Similarly, any pitch of A can be expressed mathematically as $55*2^n$, with n as some integer.

The wavelength, commonly notated λ, of a pitch varies according to the inverse of its frequency. It can be calculated by dividing the speed of sound ($\approx$ 330 meters/second, depending on air temperature) by the frequency. Thus, the wavelength of A440 would be 330/440 $\approx$ 0.75m. Wavelengths of lower pitches can be several meters in length.

The wavelengths of sound waves are many orders of magnitude larger than those of light, which accounts for our ability to hear events that occur behind obstacles, where we cannot see them. Waves are reflected when they strike a surface that is larger than the wavelength, and diffracted around the object if the wavelength is larger. Sound wave fronts, particularly of low frequencies, can easily travel around obstacles, while light waves cannot.

When two tones close in frequency are sounded simultaneously, the two waveforms create constructive and destructive interference patterns that are heard as *beats*. If the two frequencies are within a difference of 10 Hz, the perceived frequency will be the average of the two with loudness oscillations at a rate of the difference between the two. For example, playing tones of 440 and 444 Hz will result in a perceived tone of 442 Hz, with a tremolo at a rate of 4 Hz. This type of oscillation can be seen in Figure A1_3.

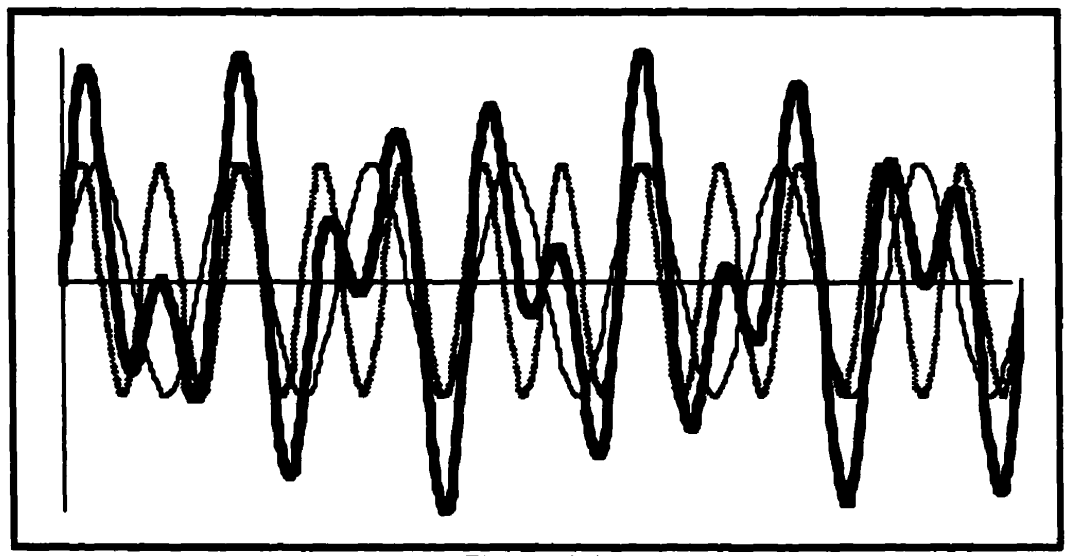


Figure A1_3:
Oscillations due to interference patterns between two pure tones close in frequency

If the two frequencies are moved farther apart, the oscillations quicken until the perception is more one of roughness than tremolo. As the frequencies fall outside of the *critical band*, the width of which varies with the frequency range, the roughness ceases and the perception is of two different tones.

The concept of beating plus the logarithmic nature of the auditory system's pitch perception is the reason why chords played in lower registers will often sound "muddy," while the same interval set played a few octaves higher will sound consonant. The frequency differences at the lower registers are much less than in the higher registers, and the resultant beating creates the "muddy" sensation.

Timbre

The simple sine wave shown in Figure A1_1 exists as sound only in synthetic environments. In everyday life, they can be heard late at night on television,

accompanying a test pattern after a station has stopped broadcasting. Tuning forks also produce a sine-like tone. What is unusual about the sinusoidal wave is that it consists of only one frequency, and is thus also called a *pure tone*.

Natural sounds are composed of multiple frequencies, and thus the shape of the wave will be more complex. The shape of the wave determines the sound's *timbre*, which is the quality of sound that allows us to differentiate between two different instruments playing the same pitch.

A primary component of timbre has to do with a phenomenon that occurs when vibrations occur within a bounded area. A clear illustration can be taken from a plucked string, which is secured at both ends. The wavelength of the resultant wave will be twice the length of the string, and the perceived pitch, also called the *fundamental pitch (f)*, will be speed of sound divided by the wavelength.

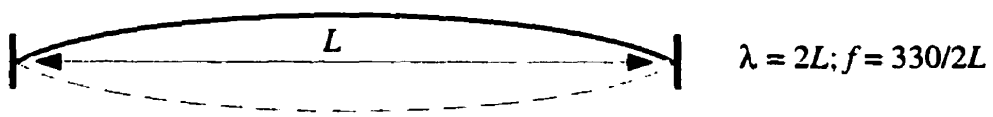


Figure A1_4: Fundamental pitch of a plucked string

The curved shape of Figure A1_4 actually represents the maximum deviation traversed by the vibrating string. The string's actual shape at any given moment is angular, with a point resulting from the string being stretched and plucked that moves along the length of the string to an endpoint, is reflected in the opposite direction, and continues to move back and forth.

The bounded nature of the string confines the propagation of the wave, and the range of frequencies it can support. Only frequency components that remain at the same phase following one motion back and forth along the string's length will continue to propagate within the string's bounded space. Other frequency components will cancel each other out, with the result that only wavelengths that have an integer relationship to the length of the string will continue to propagate. People can learn to "hear out" these frequencies added frequencies above the fundamental, a process called *analytic listening*.

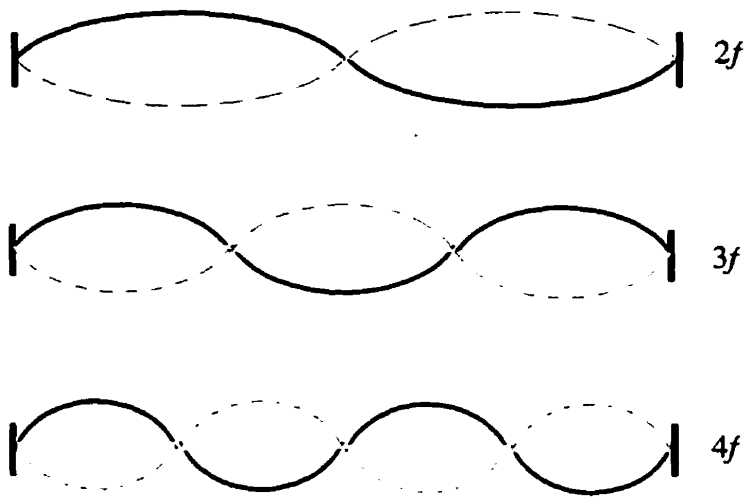


Figure A1_5: Harmonics of a plucked string

Instruments and other natural sounds may contain many frequency components above the fundamental frequency. As these additional components play a *part* in the overall sound produced, they are termed *partials*. The first partial is the fundamental frequency. The term *overtones* is also used to describe all partials excluding the fundamental. The term *harmonics* refers to the frequency components that are at integer multiples of the fundamental. String players are taught to produce harmonic tones by placing a finger lightly at the mid-point of a string, thus producing frequencies higher than the string's fundamental. The term is also used in mathematics. A *harmonic series* is a succession of inverse integers: 1, 1/2, 1/3, 1/4 ... The first harmonic is equivalent to the fundamental.

The difference in timbre between a violin and a flute playing the same pitch is threefold. One has to do with the initial attack portion of each instrument's sound, which may be a breathy *chiff* from a flute or a scraping sound from a violin. The other has to do with the different composition of the instrument bodies that produces different *resonances*, a concept that does not fall into the scope of this work. The final difference has to do with the overtone content of each instrument. Each instrument has a characteristic set of overtones at different relative volumes to each other. This fact is the basis of *additive sound synthesis*, in which pure tones at various frequencies are relative amplitudes are combined. A trumpet, for example, may be emulated by combining harmonics of a fundamental frequency, with volumes at the inverse of the harmonic number. A clarinet may be emulated

in a similar fashion, using only odd harmonics. A flute-like sound may be synthesized by using only odd harmonics with amplitudes at the inverse square of the harmonic number.

The mathematician Joseph Fourier (1768-1830) demonstrated a vital theorem of spectral analysis, which is that all periodic vibrations are composed of a series of sinusoidal vibrations, each of which are harmonics of the fundamental vibration frequency, each at a particular amplitude and phase (these two terms will be discussed presently). The decomposition of a complex waveform into its harmonic components is called a *Fourier analysis*.

Examination of harmonics gives insight as to the consonance or dissonance of various intervals. Figure A1_5 shows the harmonics of two tones at 100 and 200 Hz. We can see that all of the harmonics coincide, which is why the two fuse into a sound that can be mistaken for just one tone.

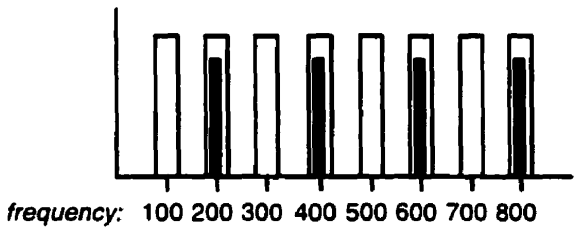


Figure A1_6: Harmonics of two tones an octave apart

Figure A1_7 shows the harmonics of two tones spaced at a *perfect fifth*, a frequency ratio of 3:2. We can see significant overlap among the harmonics, which explains why the perfect fifth is considered the most consonant interval after the octave.

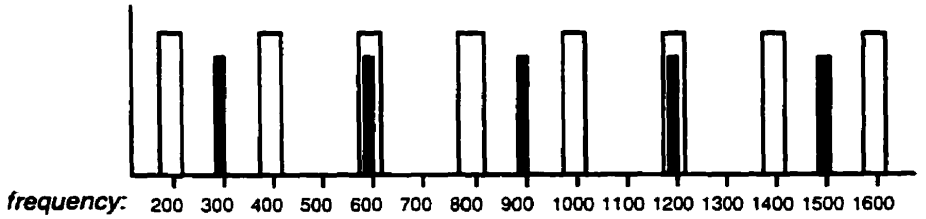


Figure A1_7: Harmonics of two tones a perfect fifth apart

Figure A1_8 shows the harmonics of two tones spaced at a *major second*, a fundamental at 200 Hz and another tone at 200 x 9/8 Hz. It is clear from the graph why this interval is considered a dissonance: there is little overlap of harmonics; furthermore, the close proximity of the partials is likely to produce beating or roughness among many of them.

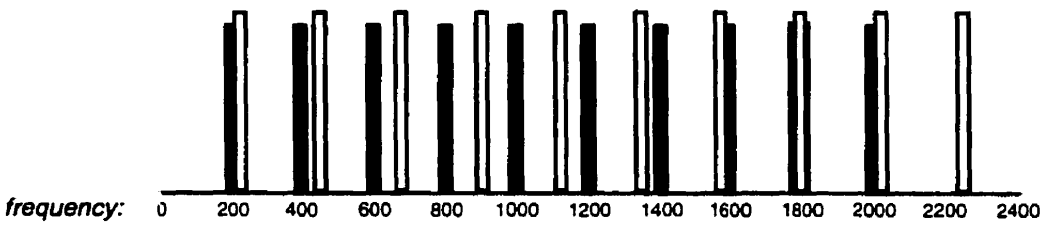


Figure A1_8: Harmonics of two tones a major second apart

Volume

The magnitude of maximum molecular disturbance determines the overall degree of change in pressure level. In Figure A1_1, the pressure level oscillates between $\pm A$. The maximum level of pressure change is the sound's *amplitude*.

The loudness of a sound is based on changes in atmospheric pressure, measured in Newtons per square meter (N/m^2). The smallest perceptible change in pressure at 1000 Hz is 2×10^{-5} N/m^2. The threshold of pain is approximately a million times greater than this threshold of hearing. Given the wide range of audible pressure levels, pressure changes are usually expressed on a logarithmic scale, in *decibels* (dB). The decibel scale is a comparison of a given sound's pressure level with the threshold level, which is abbreviated p_0 and assigned a *sound pressure level* (L_p) of 0. The pressure level L_p of a sound, measured in decibels, is

$$L_p \text{ (dB)} = 10 \log_{10}(p/p_0) \qquad \text{(A1-1)}$$

A closer description of loudness is based on the sound's *power* level, measured in Watts (W). Watts are also measured in decibels with the equation (A1-1):

$$L_W \text{ (dB)} = 10 \log_{10}(W/W_0), \qquad \text{(A1-2)}$$

with W_0 equal to 10^{-12} watts, corresponding to p_0.

A change in power is proportional to the change in pressure squared. Therefore, to use change of pressure to express changes in power, equations (A1-1) and (A1-2) may be combined:

$$L_W\,(\mathrm{dB}) = 10 \log_{10}(W/W_0) = 10 \log_{10}(p/p_0)^2 = 20 \log_{10}(p/p_0) \quad \text{(A1-3)}$$

Thus, a doubling of power results in an increase of 3 dB of power level. A doubling of pressure results in an increase of 6 dB of power level.

Typical sound power levels range from soft rustling leaves of 10 dB, to normal conversation at 60 dB, to a construction site at 110 dB, to the threshold of pain at approximately 125 dB (Rossing, 1990).

The power of a sound remains constant, radiating outward from the sound source as an expanding sphere of energy. The power level remains constant, distributed evenly over the surface of the sphere. The perceived loudness, then, is dependent both on the sound's power level and the distance of the listener from the source. This value is quantified as *intensity (I)*, measured in Watts per square meter (W/m^2). Intensity is also measured in decibels, with I_0 equal to 10^{-12} W/m^2:

$$L_I\,(\mathrm{dB}) = 10 \log_{10}(I/I_0), \quad \text{(A1-4)}$$

The perception of loudness, however, is a complex phenomenon, determined by a number of factors other than an objective measurement of intensity. Some researchers have tried to create measurement scales that reflect perceived volume. The *phon* is a subjective measurement that uses a pure tone at 1000 Hz as a reference. At 1 kHz, the phon level matches the dB level. Sounds that are perceived as matching this loudness are considered to be at the same phon level. Fletcher and Munson in the 1930s studied the ear's sensitivity to volume at different frequencies. They used a phon scale to determine the different sound pressure levels of different frequencies that create the same perceived volume. They created a set of equal loudness curves that are recognized by the International Standards Organization that illustrate the changes in pressure level necessary to maintain a constant phon level at a given frequency (Figure A1_9).

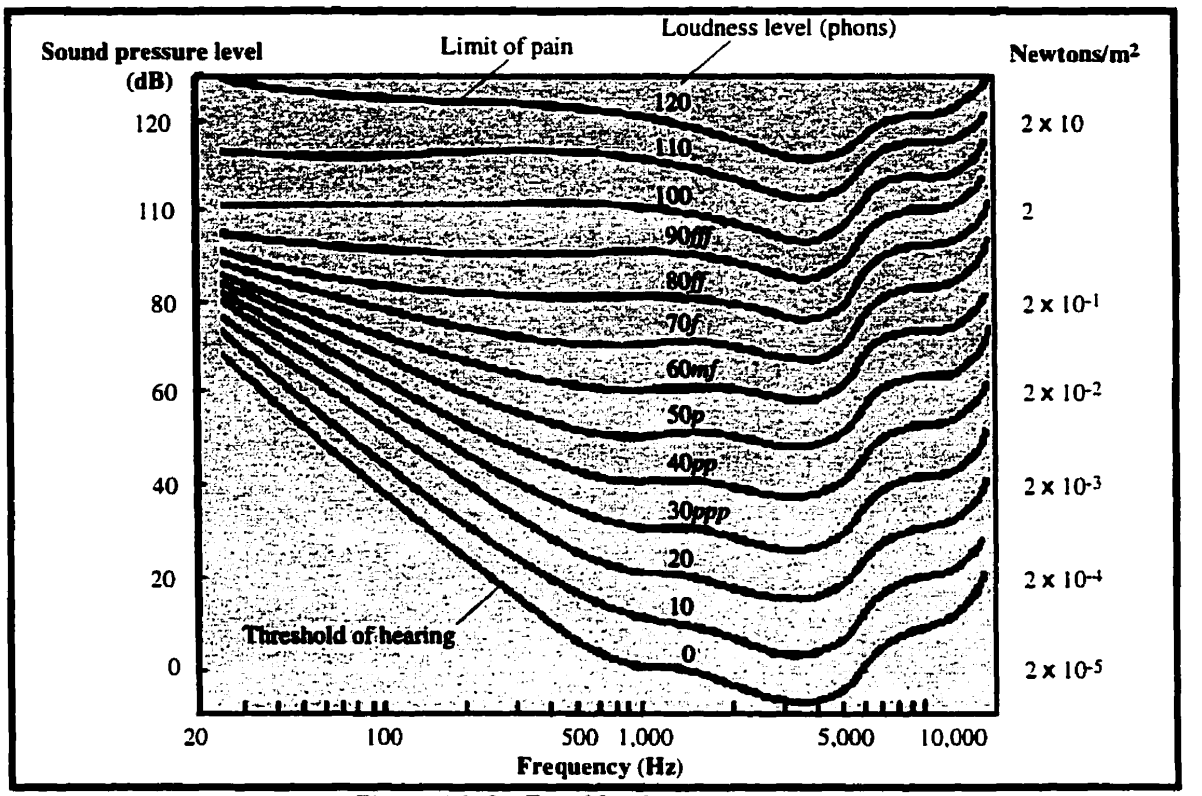


Figure A1_9: Equal loudness curves
(From:* The Science of Musical Sound *by John R. Pierce
© 1983 by W.H. Freeman and Company. Used with permission.)

The phon measurement reflects only the perception of extended steady-state tones. Equal loudness measurements need to be modified to account for the transients of sound levels changes over time, which occur in natural sounds. To account for transients, which typically contain a greater degree of higher frequencies than the steady state portion of a sound, a bias is given to the measurements to give greater weight to the higher frequencies in the final loudness determination (B.C.J. Moore, 1989).

As is further reported by Brian C.J. Moore (1989), the *sone* was a measurement proposed by S.S. Stevens in 1957. At pure tone at 1000 Hz at 40 dB is assigned a level of one sone. Stevens found that a tone at 50 dB was generally perceived as being twice as loud, and assigned an increase of 10 dB to be an increase of one sone. Within the critical band of frequencies discussed above, Stevens found that loudness was proportional to the cube root of the intensity. Thus, if one instrumentalist plays a certain pitch, and is then joined by a second instrumentalist who plays the same pitch, the intensity will be doubled but the perceived loudness

will not. Eight players would be required for a doubling of volume (Moore, 1990).

Further studies (B.C.J. Moore, 1989) have attempted to quantify loudness perception by breaking a complex sound into frequency bands (usually one-third octave), assigning the loudness of each band according to the power law described above, and then summing the loudness of each power band to determine the total loudness of the sound. However, the bandwidth of a sound also adds to the perceived volume level. Noises at a fixed intensity but variable bandwidth increase in perceived loudness once the bandwidth exceeds 175 Hz or so (B.C.J. Moore, 1989).

Definitive loudness scaling remains elusive. Numerous tests have produced varied results, depending on factors such as the range of stimuli, first stimulus presented, instructions given to the subject, etc. It cannot be said with any certainty that any perceptual scale measures loudness more effectively than does a measurement of intensity. It has also been argued that the perception of loudness in everyday life is due to a number of higher-level processes that estimate the distance, context and import of a sound event. B.C.J. Moore (1989) cites the summation of Helmholtz (1885):

> . . . we are exceedingly well trained in finding out by our sensations the objective nature of the objects around us, but we are completely unskilled in observing these sensations per se; and the practice of associating them with things outside of us actually prevents us from being distinctly conscious of the pure sensations.

Localization

The ability to localize auditory objects is based on numerous cues, some physical, some learned. There are three primary physical cues: *interaural time difference* (ITD), *interaural intensity difference* (IID) and spectral difference.

Interaural time difference is due to an off-center sound object's wave front reaching the nearer ear before it reaches the farther ear. This is the most powerful localization cue. It is also called the *precedence effect* or the *Haas effect* in sound reproduction contexts. With an identical sound stimulus emanating from multiple loudspeakers, all of which are at different distances from the listener, listeners will tend to localize the sound at the nearest loudspeaker, which produces the wave

front that reaches the ear first. In localization tests involving pure tones, ITD is the strongest perceptual cue for frequencies under 1500 Hz.

Frequencies above 1500 Hz have a wavelength under 21 cm, the average diameter of the human head. These higher frequencies tend to reflect off of the head, resulting an acoustic shadow in the region of the farther ear. Therefore, the strongest localization cue for these higher frequencies is IID.

The perception of elevation is due to reflections of the wave front off of the shoulders, as well as filtering carried out by the pinnae. This filtering provides the spectral cues that give information about elevation.

In describing the perceptual system's treatment of location, Blauert (1997) quantifies its tendencies with the term "localization blur," which is a measure in degrees of the average margin of error present in a given region. In the sanitized conditions of a laboratory, where stimuli are tightly controlled and limited to pure tones, clicks, noise and speech samples, the minimum localization blur in any direction has an average near 1°. In this regard, the auditory perceptual system demonstrates less resolution than does the visual system, with which changes in position have been perceived at less than one minute of an arc.

Perception of direction is most sensitive in a forward, horizontal direction (also known as the lateral field), with 0° being the direction in which the listener's nose points. Localization blur increases as sound sources move away from this area. At ±90°, localization blur is three to ten times as great as at 0°. Sideways localization accuracy decreases due to the *cone of confusion*, which refers to the fact objects toward the front by a given number of degrees are difficult to differentiate from objects that are rearward by the same degree factor. Imaging re-consolidates towards the rear, where the localization blur of objects directly behind averages twice that of the forward perception.

Elevation perception is less certain. Elevation tests involving continuous speech of an unfamiliar voice have shown a localization blur of 17°, a blur of 9° when the speech is that of a familiar voice, and 4° for wideband noise. With the stimulus of narrowband noise, there is virtually no perceptibility in elevation; instead, the

perception of height becomes associated with the pitch of the sound. The higher the pitch, the higher in height the sound's location is perceived.

Audio engineers simulate localization via loudspeakers by creating *phantom images*. A sound source from two equidistant loudspeakers will be localized in space, directly between them. Changing the intensity of one speaker will "pull" the phantom image toward the louder source; placing a delay on one speaker will "pull" the phantom image towards the loudspeaker that produces the first wave front to reach the listener. In comparing the effects of ITD and IID, it has been found that a difference of approximately 18 dB in amplitude (9 dB in intensity) is necessary to overcome the precedence effect. While ITD is by far the stronger cue, its effectiveness is dependent on the listener being in a central "sweet spot," equidistant from each speaker. The effectiveness of intensity panning, on the other hand, can be appreciated within a much wider listening area. It is only the rare audiophile who sits stationary in a central listening position when listening to music at home. For this reason intensity panning, rather than delay panning, is employed in the vast majority of commercial recordings.

More specific localization images can be obtained by simulating the filtering done by the pinnae. Attempting to create such effects is problematic for two reasons. One is that each individual's pinnae produce a different filtering operation. Researchers have had some success through the use of *head related transfer functions* (HRTFs), which are a general model of a typical ear's response. Effects through HRTFs are very dependent on listener location, however, and are usually only effective is played over headphones, or with close listening environments such as personal computer speakers.

Phase

Phase concerns the time relationship of two sinusoidal waveforms that do not have simultaneous zero-crossings. Figure A1_10 shows two sinusoidal waves of the same amplitude and frequency, but of different phases, and the resultant combination wave.

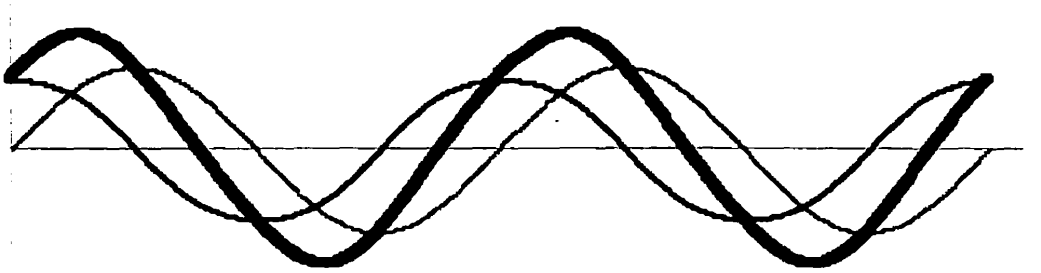


Figure A1_10: Sum of two sine waves with the same frequency and different phases

Curiously, the phase relationship of sound waves is at times critical and at other times not perceivable at all. Analysis of timbres considers the volume changes over time (the *envelope*) of a sound's overtones. The envelope is often broken into two coarse segments: the attack portion and the steady state portion. Timbral research has shown that the attack portion of a tone is its defining characteristic. The overtone content of a synthesized instrument has far less to do with its perceived simulation of an acoustic instrument than does the envelope shape (Chowning, 1974). The phases of the partials can be critical in defining the attack of a sound. Audio engineers frequently need to employ phase-correcting filters to avoid blurring due to amplification systems in which the phase relationship of the partials has been altered. In a concert setting, the sound of a solo performer is very different from the sound of multiple performers, even playing the same material on the same instruments. No two human beings will ever be in perfect synchronization, so there will be a less distinct attack on ensemble playing than there will be on solo playing.

On the other hand, the ear is completely insensitive to phase in steady-state tones, a discovery that dates back to Helmholtz (1885). This phenomenon is in some ways counter-intuitive, as the sum wave of many harmonics can have a drastically different shape, depending on the phases of the harmonics. Yet, the sound of a steady-state complex tone with uniform phases will be indistinguishable from a tone containing the same set of harmonics with different phase relationships to each other. This auditory insensitivity is likely an evolutionary development. The reason for it becomes clear with the example of solo versus ensemble performance. If, during the performance of a duet, one player should take a step toward or away from the listener, the phase relationship of the two instruments

will be changed. If such phase changes produced drastic changes in sound quality, the sound would be altered by any movement the performers make, resulting in auditory disarray. The absence of significant qualitative changes in sound due to phase is a vital element in our ability to make sense of our environment through sound.

Appendix 2

Fundamentals of Nonlinear Dynamics

Iterative Functions, Asymptotic States and Chaos

A linear equation is one that has only one variable, to the power of 1. The generic linear equation is $y = mx + b$. Plotted on a Cartesian plane, this equation will produce a straight line, with a *slope* of m that intersects the y-axis at value b. Examples of linear equations include Ohm's law, $V = IR$, in which electrical voltage (V) increases proportionally to increases in current (I) provided the resistance (R) remains constant. Another example is the equation *Distance* = *Rate*x*Time*, in which the distance traveled is a simple function of time, provided the rate remains constant. Linear systems are "well behaved," in that the output of such a system is proportional to the sum of its inputs, and the entire system can be understood by looking at each component separately (Goldberger, 1996).

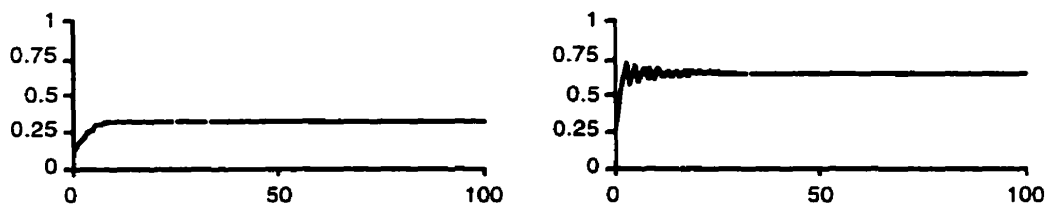
Biological systems, however, are usually not so easily described. They are often described by equations with more than one variable or equations with one or more variables at higher powers than 1. Such equations, when graphed, do not produce a straight line, and are thus termed *nonlinear*. Nonlinear systems are not as easily decomposed as linear systems. Minute changes in input elements can produce large-scale changes in the output, and the interactions of the elements rule out explanation by simple examination of each element separately. In cardiology, as was noted in the Introduction, it is thought that fluctuations in the heart rate are due to nonlinear interactions among the sinus node and the sympathetic and parasympathetic nervous systems.

While biological systems are in a continual state of change, they must be measured at discrete time increments, notated t to mean "at time t." The condition of the system at the present time is dependent on its condition in the past, just as the present condition determines future states of the system. Discrete measurements of deterministic systems are described by *iterative equations*. An iterative equation is one that takes the form: $x_{t+1} = f(x_t)$, which states that the condition of the system at time $t + 1$ is dependent on the state of the system at time t.

The *logistic difference equation* is often employed to describe such biological systems: $x_{t+1} = Rx_t(1 - x_t)$. This equation is useful for modeling dynamics such as population levels in an environment with a finite amount of space and a limited food supply. For values of R greater than 1, the output will grow steadily when x_t is small, and diminish as x_t approaches a value of 1 (that is, 100% of the population capacity in a given environment). The output of such equations, like audio signals, often begins with a short, highly active state (the *transient*), before a stable state or cycle is emerges that does not change significantly even as the number of iterations approaches infinity (the *steady*, or *asymptotic* state).

The asymptotic state is determined by the initial value, x_0, and the value of the scalar R. Choosing both of these values and plotting a number of iterations shows that the asymptotic state of the logistic equation can take a number of forms. At low values of R, a *fixed point* will emerge. The approach to the fixed point may be *monotonic* (a steady approach) or *alternating* (alternating above and below the fixed point value). Figure A2_1 illustrates both approaches with an initial value of 0.25, and scalar values of 1.5 and 2.9.

Figure A2_1:



$x_{t+1} = 1.5\,(1 - x_t)x_t$ $\qquad$ $x_{t+1} = 2.9\,(1 - x_t)x_t$

from (Kaplan and Glass, 1995)

As R increases, different types of asymptotic behavior result. One type of behavior is a periodic cycle. Figure A2_2 shows a cycle of 2 that emerges when $R = 3.3$.

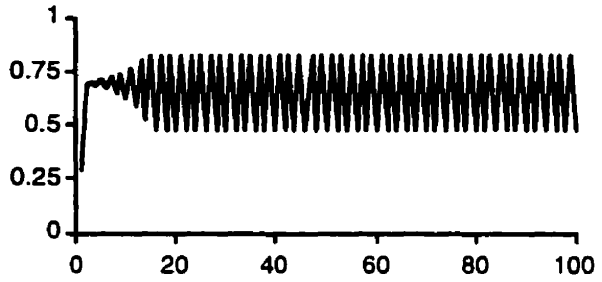


Figure A2_2: $x_{t+1}=3.3\,(1-x_t)\,x_t$ from (Kaplan and Glass, 1995)

A fixed point may be *globally stable* if all initial conditions iterate to it, *locally stable* if initial conditions near the fixed point will iterate to it, or *unstable* if only a precise initial condition will iterate to that fixed point. In the same way, cycles may be globally stable, locally stable if initial conditions near points on the cycle will iterate to that cycle, or unstable if only a very precise set of initial conditions lead to then. The set of initial conditions that lead to a particular fixed point of cycle is called the *basin of attraction* for the fixed point or cycle.

A different type of asymptotic condition occurs for a scalar value of 4. The condition is called *chaos*, or *deterministic chaos*, to differentiate the condition from the colloquial definition of random, catastrophic disorder (Goldberger, 1996).

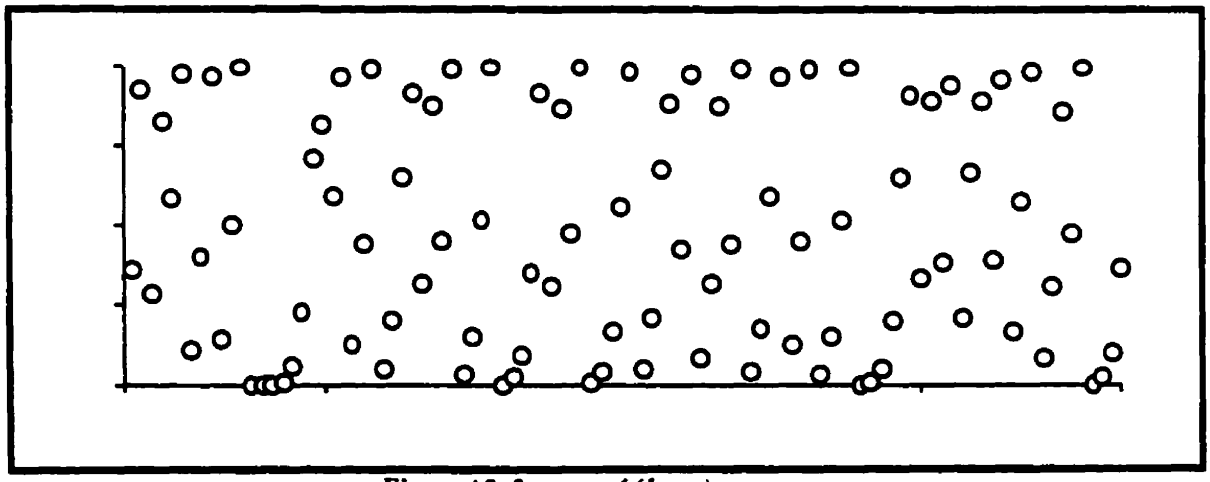


Figure A2_3: $x_{t+1}=4\,(1-x_t)\,x_t$ from (Kaplan and Glass, 1995)

Deterministic chaos displays four characteristics:

- *Aperiodic* – the system never repeats itself. (Although in practice, long cycles may be difficult to distinguish from aperiodicity.)

- *Bounded* – the system will remain in a finite range, and not approach infinity. The logistic difference equation will always produce values between 0 and 1 for initial conditions within that range.

- *Deterministic* – each value is entirely dependent on previous value, with no random elements. For any x_t there is only one value of x_{t+1}, and all future points can be determined given x_0. In practice, it can be difficult to determine whether a natural system, in which the initial value is not known, is completely deterministic or contains random elements.

- *Sensitive dependence on initial conditions* – the iterated points will depend on the value of x_0. Given two initial conditions, even two values very close to each other, their iterations will soon diverge and iterate to very different sets of values.

As the value of R changes in the logistic difference equation, the asymptotic state may be fixed points, cycles of varying lengths, or chaos. These changes in asymptotic behavior as a result of a change of one parameter are called *bifurcations*. Starting with values of R between 1 and 3, logistic difference equation's asymptotic behavior goes through a series of cycles, the period of which double as the value of R increases. This type of behavior is called a *period-doubling bifurcation*. When R is increased to the range above 3.57, the asymptotic behavior takes on a variety of periodic and aperiodic behaviors. A *bifurcation diagram* is often employed to illustrate the changes of asymptotic behavior as a function of R.

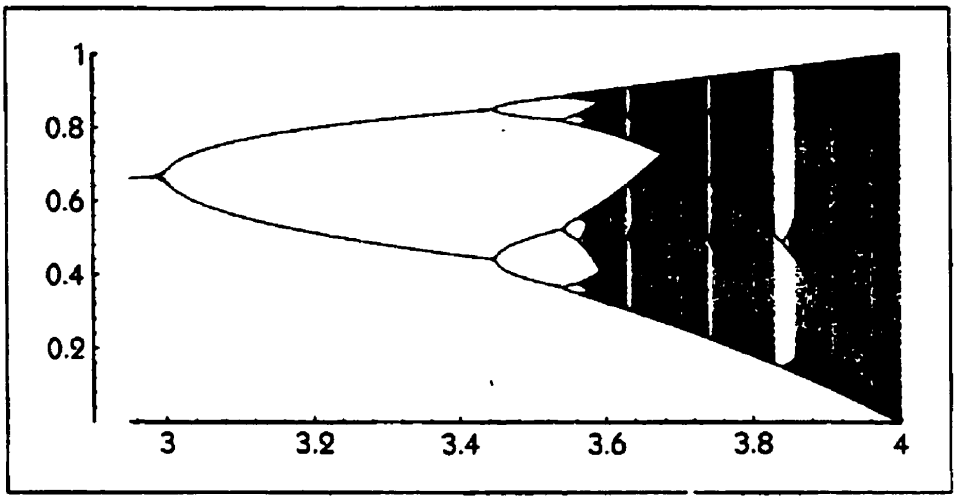


Figure A2_4: Bifurcation diagram for the logistic difference equation.
From (Kaplan and Glass, 1995). Re-printed with permission of Springer-Verlag New York, Inc.

Fractals

This branch of nonlinear dynamics is an exploration of *self-similarity* over multiple scales. Magnifications of a fractal object reveal that it is composed of smaller versions of its whole. The creation of two self-similar images is shown below.

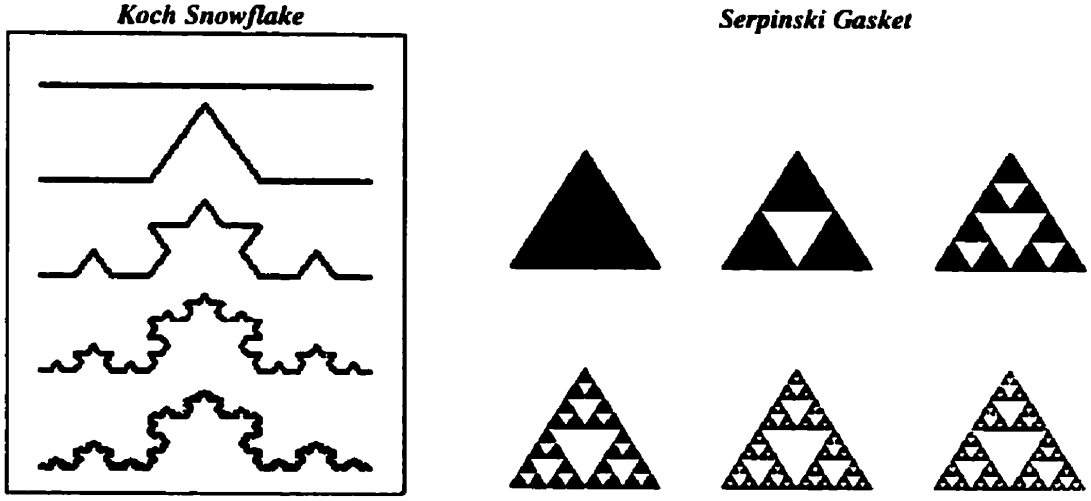


Figure A2_5: Creation of two self-similar figures
From (Kaplan and Glass, 1995). Re-printed with permission of Springer-Verlag New York, Inc.

In the two figures above, successive iterations create smaller versions of the original drawing on successively smaller scales. To describe the structure of such objects, mathematician Benoit Mandelbrot created a variant of the term

"dimension." (Mandelbrot, 1983; Kaplan and Glass, 1995). For fractal images such as these, the dimension D is characterized by the number of self-similar copies present N and the edge length of the original image relative to the edge length of each successive copy, ε, according to the formula

$$D \equiv \frac{\log_{10} N}{\log_{10} \varepsilon}$$

The resulting dimension for objects represented on a flat surface, such as the Koch Snowflake of the Serpinski Gasket, will be a number between 1 and 2. Since the dimension is a fractional number, Mandelbrot coined the term *fractal* to apply to such objects.

These images, in which each iteration produces an identical version of the same image, are an example of *geometric* self-similarity. Musically, this kind of relationship might be illustrated by equivalent intervals over different time scales, as illustrated below. This type of hierarchical self-similarity has been observed in Balinese Gamelan music (Chou, 1971).

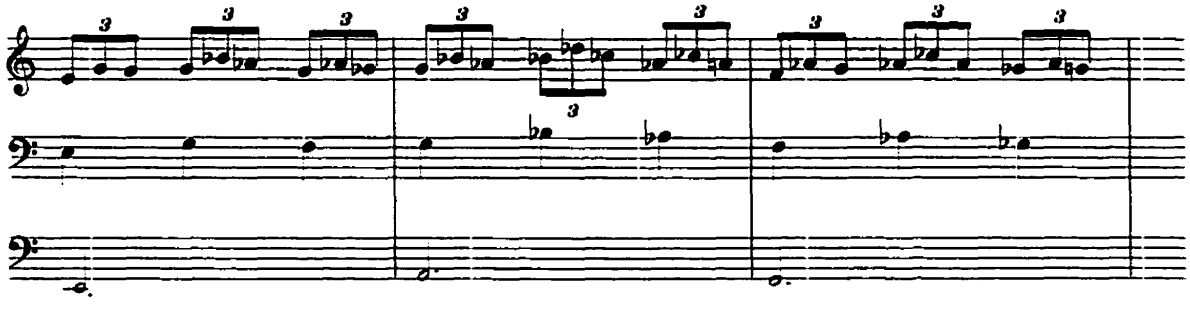


Figure A2_6: "Self-similar" music

These images, however, are a simplified introduction to Mandelbrot's attempts to characterize many of the apparently irregular forms present in the natural world. Classic geometry is concerned with "perfect" forms such as the sphere or the cube. These forms, however, do not exist in nature. Mountains are not cones and trees are not cylinders. Mountains, however, often contain smaller outcroppings that resemble the shape of the larger mountain. Coastlines often have inlets or bays, which themselves contain smaller inlets or bays. Self-similar branching structures can often be found in trees, the vascular system, the bronchi of the

lungs, or the system of deltas that branch from a river as it approaches an ocean or sea. These structures, which contain apparent, rather than exact, copies, are examples of *statistical* self-similarity.

As described by Gleick (1987), Mandelbrot's exploration of self-similarity arose from his work at the IBM research center in Tarrytown, NY. Researchers were encountering problems in transferring data among the computers, which were connected by telephone lines. Data flow was interrupted by intermittent noise bursts, which appeared to occur randomly. Mandelbrot found that the electrical noise that appeared unpredictably was consistent over different time scales. The ratio of silence to noise averaged to the same value over scales of a minute, hour, day, etc. As the theory of self-similarity became popularized, statistical self-similarity was subsequently found to exist in floodings of the Nile river, rainfall in Amazon rainforests, traffic flow at intersections, and many other phenomena (Mandelbrot, 1983). An intuitive sense of statistical self-similarity can be observed in objects such as coastlines, mountains or clouds, whose apparent degree of roughness does not change with magnification. While such objects may not always appear geometrically self-similar, they may display statistical self-similarity and their fractal dimension, notated D, can be estimated by the "box counting" method (Kaplan and Glass, 1995). This is a variation on the fractal dimension equation shown above:

1. Cover all points in the object with boxes with edge-length $\in_0$. Count the number of boxes, and call the result $N(\in_0)$.
2. Repeat step 1, each time halving the edge length of the box, so that
   $\in_1 = \in_0/2$
   $\in_2 = \in_1/2$
   $\in_3 = \in_0/2$, and so on.

   The appropriate number of repetitions will depend on the object.
3. The fractal dimension can be estimated as

$$D \equiv \frac{\log \frac{N(\in_{i+1})}{N(\in_i)}}{\log \frac{\in_i}{\in_{i+1}}}$$

When this formula is applied to a familiar geometric shape such as a square or a cube, it will give the familiar Euclidean dimensional values of 2 and 3, respectively. When applied to a self-similar figure, the fractional value that results provides a quantitative means of describing an object in terms of its degree of self-similarity.

Scaled Noise

Statistical scale invariance over time is the basis of *noise*. Generalized from the common definition of "unpleasant sound," scientists use the term to refer to data values with varying degrees of randomness and correlation. The most extreme form of noise is a complete absence of correlation, each value being completely random and unrelated to those that preceded it. A Fourier transform performed on a series of uncorrelated numbers will produce a spectrum in which each frequency has an equal probability of occurring. If a continuous signal is windowed and a Fourier transform is performed on each window, the result will be an averaged spectrum in which all frequencies are present at equal magnitudes. This type of signal referred to as *white noise*, in an analogy to white light, which contains all frequencies of the visible spectrum in equal proportion.

White noise is one class of what Mandelbrot termed *scaled noise*, a special class of sounds that, when played on a variable speed tape recorder, do not change in character as the speed of playback changed (Gardner, 1978). Scaled noises are also referred to as $1/f^{\alpha}$ noises, referring to a spectral plot, with power on the vertical axis as a function of frequency on the horizontal axis. The value of the exponent α defines the nature of the noise. For white noise, the magnitude of each frequency will be at maximum, 1, so it may be described as $1/f^{0}$ noise.

A second important class of scaled noise has an exponent of 1, *1/f noise*. $1/f$ noise is often used interchangeably with the word "fractal," due to an interesting feature of the function $f(x) = 1/x$. Examining the positive range of this function, the integral $\int 1/x\,dx$ will produce the same result for exponentially equivalent ranges of dx, as shown below:

$$\int_{ab^n}^{ab^{n+1}} \frac{dx}{x} = \ln x\, dx \qquad n = 0, 1, 2 \ldots$$

$$= \ln(ab^{n+1}) - \ln(ab^n)$$

$$= (n+1)\ln(ab) - n\ln(ab)$$

$$= (n+1-n)\ln(ab)$$

$$= \ln(ab)$$

As an example, if we let a equal 55 and b equal 2, then incrementing n will produce ranges of 55-110, 110-220, 220-440, 440-880, 880-1760, etc. The value of the integral, $\ln x\, dx$, in all cases will be 0.6931472. Regardless of the values we choose for a and b, the value of the integral will be scale invariant in that there will always be the same area under exponentially related segments of the curve.

Gardner (1978) presents an algorithm for creating a number series that contains a $1/f$ distribution. Some number n of random number generators, perhaps dice, is used. Each die is associated with a bit in a binary number, which is considered to increment from 0 to 2^n. An initial roll of all the dice sets an initial sum. The binary number is then incremented. Each time a bit changes from 1 to 0 or from 0 to 1, a new value on its associated die is generated. At each increment, the appropriate dice are rolled, and the sum of all the dice is taken. Thus, some numbers will change more rapidly than others will. The die associated with the 1 bit will change with every increment, the die associated with the 2 bit will change every two increments, the die associated with the 4 bit will change every four iterations, and so on. Numbers produced in this fashion "have a memory," due to the less frequent changes of the random generators representing the larger bits.

If the numbers generated represent audio samples, the resulting signal will contain a spectrum that follows the curve $1/f$. The values for a, b and n above represent successive octaves starting on the pitch class A. The power is constant over every octave. Since the spectrum contains higher magnitudes in the lower frequency ranges, this type of noise known as *pink noise*. The name is derived from another analogy to visible light, in which the lower frequencies are at the red end of the spectrum.

The third important scaled noise is *Brown noise*, named not in an analogy to light, but after Robert Brown, who observed the erratic motion of pollen grains in a

glass of water. His observations led to conclusive proof of molecular diffusion when Einstein proved that the movements were the result of pollen grains interacting with the water molecules.

This *Brownian motion* is often modeled with the "*drunken walk*," or *random walk*, analogy. We imagine an inebriate's impaired sense of equilibrium results in a series of steps, each of which goes in a random direction. The distance traveled will be proportional to the square root of the number of steps taken. Gardner (1978) also presents an algorithm to generate a "Brownian" series of numbers. Starting from an initial value, a random number in the range ± 1 is added to the total with each iteration. The spectrum of this Brown noise is $1/f^2$.

Appendix 3

The Poisson Distribution

The Poisson Distribution describes uncorrelated events that happen one at a time over a continuous time or space. If the average number of events to occur within a given timespan is known, the Poisson distribution attaches a quantitative value to the probability that an event will occur at a given instant, or how many events are likely to occur within a subset of that timespan. With low means, the distribution resembles an exponential curve; as the mean increases, the probability curve resembles a bell curve that is centered about the mean. A detailed overview of the history and applications of the Poisson Distribution is contained in (Haight, 1967).

Simeon Denis Poisson (1781-1840), a prominent mathematician and physicist who held many positions in the French academic and scientific community, derived the Poisson distribution in 1837. Late in his life he concentrated on probability theory in sociology, specifically in the administration of justice. His introduction of the distribution was to quantify how often juries were likely to come up with correct verdicts. As probability distributions became linked with statistics in the nineteenth century, the formula was generalized to describe many types of discrete events. The earliest application described how often death occurred by horse kicking in the Prussian army. It is now used in a variety of descriptions in sociology, industry and science. Many of the phenomena it describes are similar to those described by Mandelbrot, such as traffic activity over a period of time or over a stretch of road. Other implementations include ecological models to determine distribution of animals within an area of terrain, scientific models to determine events such as how likely an unstable atomic nucleus is to emit energy, and in industry for problems such as how busy telephone switchboards are likely to be at a given time, how many defective items are likely to be found in a given shipment, or what the demand is likely to be for retail goods. It is also a facet of risk theory, used by insurance companies to determine the numbers of deaths due to transportation and other accidents.

The shape of the distribution is determined by the mean of the series, notated λ. With the mean given, the possibility of generating the integer j is given by the formula:

$$P\{X = J\} = e^{-\lambda}(\frac{\lambda^J}{J!})$$

An algorithm to generate numbers according to the Poisson Distribution is given in (Dodge and Jerse, 1995), shown in the following program:

```
#define X some_positive_integer;
#define mean some_positive_integer;
int main(void)
{
float reference;
int i, poissonNumber;
int poissonArray[X];
int poisson(void);

reference = exp(mean);
for (i=0; i<X; i++) {
   poissonNumber = poisson();
   poissonArray[i] = poissonNumber;
   }
return 0;
}

int poisson(void)  {
int n=0; r=1; lessThan=0;
while (lessThan == 0) {
  r *= rand();
  if (r < reference)
    { lessThan = 1;
      return n }
  else
    { n += 1 }
}
```

Appendix 4

Csound Code for Encoding Instrument Orchestra File

```
;-----------------------
     sr = 44100
     kr = 441
  ksmps = 100
 nchnls = 4
;-----------------------


        instr 1
;-----------------------
   idur = p3
   ihrv = p3*100
   iamp = ampdb(ihrv)
 ipitch = (1/ihrv)*440
   ivol = iamp*3000

   kone = ihrv*.7854
   ktwo = 0

   kenv linen   ivol, idur*.01, idur, idur*.15

        if (p3 < .008) goto wave2
        if ((p3 >= .008) && (p3 < .0095)) goto wave3
        if ((p3 >= .0095) && (p3 < .011)) goto wave1
        if (p3 >= .011) goto wave4

wave2:
     a5 oscili   kenv, ipitch, 2
        goto contin

wave3:
     a5 oscili   kenv, ipitch, 3
        goto contin

wave1:
     a5 oscili   kenv, ipitch, 1
        goto contin

wave4:
     a5 oscili   kenv, ipitch, 4
        goto contin

contin:
    kca = cos(kone)
    ksa = sin(kone)
    kcb = cos(ktwo)
    ksb = sin(ktwo)

     ax = a5*kca*kcb
     ay = a5*ksa*kcb
     az = a5*ksb
     aw = a5*.707

        outq ax, ay, az, aw

        endin
```

```
        instr 2
;------------------------
   idur = p3
   ihrv = p3*50
 ipitch = (1/(ihrv))*440
   iamp = ampdb(ihrv)
   ivol = iamp*3000

   kone = ihrv*2.3562
   ktwo = 0

   kenv linen   ivol, idur*.01, idur, idur*.15

        if (p3 < .017) goto wave2
        if ((p3 >= .017) && (p3 < .019)) goto wave3
        if ((p3 >= .019) && (p3 < .022)) goto wave1
        if (p3 >= .022) goto wave4

wave2:
     a5 oscili kenv, ipitch, 2
        goto contin

wave3:
     a5 oscili kenv, ipitch, 3
        goto contin

wave1:
     a5 oscili kenv, ipitch, 1
        goto contin

wave4:
     a5 oscili kenv, ipitch, 4
        goto contin

contin:
    kca = cos(kone)
    ksa = sin(kone)
    kcb = cos(ktwo)
    ksb = sin(ktwo)

     ax = a5*kca*kcb
     ay = a5*ksa*kcb
     az = a5*ksb
     aw = a5*.707

        outq ax,ay,az,aw

        endin
```

```
        instr 3
;-----------------------
   idur = p3
   ihrv = p3*25
 ipitch = (1/ihrv)*440
   iamp = ampdb(ihrv)
   ivol = iamp*3000

   kone = ihrv*3.927
   ktwo = 0

   kenv linen   ivol, idur*.01, idur, idur*.15

        if (p3 < .037) goto wave2
        if ((p3 >= .037) && (p3 < .042)) goto wave3
        if ((p3 >= .042) && (p3 < .046)) goto wave1
        if (p3 >= .046) goto wave4

wave2:
     a5 oscili kenv, ipitch, 2
        goto contin

wave3:
     a5 oscili kenv, ipitch, 3
        goto contin

wave1:
     a5 oscili kenv, ipitch, 1
        goto contin

wave4:
     a5 oscili kenv, ipitch, 4
        goto contin

contin:
    kca = cos(kone)
    ksa = sin(kone)
    kcb = cos(ktwo)
    ksb = sin(ktwo)

     ax = a5*kca*kcb
     ay = a5*ksa*kcb
     az = a5*ksb
     aw = a5*.707

        outq ax,ay,az,aw

        endin
```

```
        instr 4
;------------------------
   idur = p3
   ihrv = p3*12.5
 ipitch = (1/ihrv)*440
   iamp = ampdb(ihrv)
   ivol = iamp*3000

   kone = ihrv*5.4978
   ktwo = 0

   kenv linen   ivol, idur*.01, idur, idur*.15

        if (p3 < .07) goto wave2
        if ((p3 >= .07) && (p3 < .08)) goto wave3
        if ((p3 >= .08) && (p3 < .09)) goto wave1
        if (p3 >= .09) goto wave4

wave2:
     a5 oscili kenv, ipitch, 2
        goto contin

wave3:
     a5 oscili kenv, ipitch, 3
        goto contin

wave1:
     a5 oscili kenv, ipitch, 1
        goto contin

wave4:
     a5 oscili kenv, ipitch, 4
        goto contin

contin:
    kca = cos(kone)
    ksa = sin(kone)
    kcb = cos(ktwo)
    ksb = sin(ktwo)

     ax = a5*kca*kcb
     ay = a5*ksa*kcb
     az = a5*ksb
     aw = a5*.707

        outq ax,ay,az,aw

        endin
```

Appendix 5

SuperCollider Code for Sonification Models

1. CVAA Sonification Model

```
NN INTERVALS, NN50 INTERVALS, WAVELET CONVOLUTION VALUES, HILBERT TRANSFORM VALUES,
MEDIAN FILTERED VALUES, RUNNING WINDOW WITH CURRENT NN POINT IN THE MIDDLE AND THE
MEDIAN VALUE OF THE WINDOW

(
// GLOBALS
var timedelta;
var glasstable, metaltable, sawtable, f3456table, fplushi, squaretable;

// NN INTERVAL RELATED STUFF
var nnlist, nnpitches;
var nnvol;
var nndisplay;
var nn_medVol;

// NN50 STUFF
var nn50vol;

// WAVELET CONVOLUTION STUFF
var wavelist, wavepitches;
var waveVol;

// HILBERT TRANSFORMED STUFF
var hilblist, hilbpitches;
var hilbvol;

// MEDIAN FILTERED HILBERT TRANSFORM RELATED STUFF
var base;
var thresh1, thresh2, thresh3, thresh4, thresh5;
var medianlist, medianthreshold, medarraylength, medamps, backtrack, currentPoints;
var medianvol;
var timbrevol;
var mediandisplay;
var medianratios;

// STUFF FOR THE MEDIAN PITCH
var medpitcharraylength;
var medpitch;
var midlength;
var iMax;
var medWinVol;

// CLOCK
timedelta = 0.016;

// WAVETABLE DEFINITIONS
glasstable=Wavetable.sineFill(512, [1, 0, 0, 0.2, 0, 0, 0, 0, 0.1, 0, 0, 0.1, 0, 0, 0.1, 0, 0,
0, 0, 0, 0.1]);
metaltable = Wavetable.sineFill(512, [1,0.75, 0.5, 0.25, 0.1, 1, 0.75, 0.5, 0.25, 0.1]);
sawtable = Wavetable.sineFill(512, 1/[1, 2, 3, 4, 5, 6, 7]);
f3456table = Wavetable.sineFill(512, [1, 0, 1, 1, 1, 1]);
fplushi = Wavetable.sineFill(512, [0.3, 0, 0, 0, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]);
squaretable = Wavetable.sineFill(512, [1, 0, 0.3, 0, 0.2, 0, 0.143, 0, 0.111]);

// DEFINE NN INTERVAL VARIABLES
nnlist = thisProcess.interpreter.executeFile(":wavelets:slp66_nnlist");
nnpitches=Array.fill(nnlist.size, 0);
nnlist.do({ arg item, i; nnpitches.put(i, 128*(2**(1/item))) });

// DEFINE WAVELET CONVOLUTION VARIABLES
wavelist = thisProcess.interpreter.executeFile(":wavelets:slp66_wavelist");
wavepitches=Array.fill(wavelist.size, 0);
wavelist.do({ arg item, i; wavepitches.put(i, 512*(2**item)) });

// DEFINE HILBERT TRANSFORM VARIABLES
hilblist = thisProcess.interpreter.executeFile(":wavelets:slp66_hilblist");
hilbpitches=Array.fill(hilblist.size, 0);
hilblist.do({ arg item, i; hilbpitches.put(i, 512*(2**item)) });

// DEFINE MEDIAN FILTER VARIABLES
medianlist = thisProcess.interpreter.executeFile(":wavelets:slp66_medlist");
medianthreshold=0.6244;
medarraylength=32;
thresh1=0.35;
thresh2=0.5;
thresh3=0.6;
thresh4=medianthreshold;
thresh5=1-((1-medianthreshold)*0.5);
base=128;
medamps=Array.fill(medarraylength, { arg item; 0.1-((item/medarraylength)/10)  });
```

```
currentPoints=Array.fill(medarraylength, 0);
medianratios=Array.fill(medianlist.size, 0);
medianlist.do({ arg item, i; medianratios.put(i, 512*(2**(item)) ) });

backtrack = { arg index, sourcelist, destlist, multiple;
      medarraylength.do({ arg item; destlist.put(item, multiple*(2**sourcelist.at(index-
item))); })
      };

// MEDIAN PITCH VALUE STUFF
medpitcharraylength=32;
midlength=(medpitcharraylength*0.5).asInt;
iMax=nnlist.size-midlength;
medpitch= { arg index;
     var temparray, sortedlist, medianPoint, otherMedianPoint, theMedian, thePitch;
     temparray=Array.fill(medpitcharraylength, 0);
     medpitcharraylength.do(
       { arg i; temparray.put(i, nnlist.at((index+i)-(midlength))) });
     sortedlist=temparray.sort;
     medianPoint=sortedlist.at(((medpitcharraylength)*0.5).asInt);
     otherMedianPoint=sortedlist.at(((medpitcharraylength)*0.5).asInt-1);
      theMedian=medianPoint-((medianPoint-otherMedianPoint)*0.5);
      thePitch=128*(2**(1/theMedian));
     thePitch;
     };

// GUI
w = GUIWindow.new("panel", Rect.newBy( 176, 77, 313, 339 ));
  StringView.new( w, Rect.newBy( 11, 8, 71, 18 ), "NN Int");
  StringView.new( w, Rect.newBy( 86, 8, 71, 18 ), "Median filt");
  nndisplay=NumericalView.new( w, Rect.newBy( 13, 29, 64, 20 ), "NumericalView", 0.908, -
1e+10, 1e+10, 0, 'linear');
  mediandisplay=NumericalView.new( w, Rect.newBy( 86, 29, 128, 20 ), "NumericalView",
0.817987, -1e+10, 1e+10, 0, 'linear');
  nnvol=SliderView.new( w, Rect.newBy( 13, 62, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 62, 128, 20 ), "Beat-to-beat");
  nn_medVol=SliderView.new( w, Rect.newBy( 13, 86, 128, 20 ), "SliderView", 0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 86, 128, 20 ), "NN/Median filt");
  nn50vol=SliderView.new( w, Rect.newBy( 13, 117, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 117, 128, 20 ), "NN50");
  waveVol=SliderView.new( w, Rect.newBy( 13, 155, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 155, 128, 20 ), "Wavelet");
  hilbvol=SliderView.new( w, Rect.newBy( 13, 183, 128, 20 ), "SliderView", 0.0, 0, 3, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 183, 128, 20 ), "Hilbert");
  medianvol=SliderView.new( w, Rect.newBy( 13, 223, 128, 20 ), "SliderView", 0.0, 0, 3, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 223, 128, 20 ), "Median Filtered");
  timbrevol=SliderView.new( w, Rect.newBy( 13, 247, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 247, 128, 20 ), "Timbres");
  medWinVol=SliderView.new( w, Rect.newBy( 13, 285, 128, 20 ), "SliderView", 0, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 285, 140, 20 ), "Median Running Window");

Synth.play({ arg synth;
  var sineenv, percenv, medarraypercenv;
  var ourosc, ourfreq, oscupdate;
  var glassosc, metalosc, sawosc, f3456osc, fplushiosc, squareosc, higlass;

  sineenv=Env.sine(timedelta*3);
  medarraypercenv=Env.perc(timedelta*0.1, timedelta*2.9, 1, -4);
  percenv=Env.perc(timedelta*0.1, timedelta*4.9, 1, -4);
  ourfreq=Plug.kr(base);

  glassosc=Osc.ar(glasstable, ourfreq, 0, timbrevol.kr);
  metalosc=Osc.ar(metaltable, ourfreq, 0, timbrevol.kr);
  sawosc=Osc.ar(sawtable, ourfreq, 0, timbrevol.kr);
  f3456osc=Osc.ar(f3456table, ourfreq, 0, timbrevol.kr);
  fplushiosc=Osc.ar(fplushi, ourfreq, 0, timbrevol.kr);
  squareosc=Osc.ar(squaretable, ourfreq*1.5, 0, timbrevol.kr);
  higlass=Osc.ar(glasstable, ourfreq*2, 0, timbrevol.kr);

  // USE OUROSC AS THE DRONE, DEPENDING ON THRESHOLDS OF THE MEDIAN FILTERED DATA
  ourosc=Plug.ar(glassosc, 0);

  // UPDATE OUROSC EVERY [TIMEDELTA] SECONDS
  synth.repeatN(0, timedelta, medianlist.size-1,
         { arg synth, now, count;
           var osclist;
           osclist=[f3456osc,metalosc,fplushiosc,glassosc,squareosc,higlass];
           if ( (medianlist.at(count) < thresh1),
               { ourosc.source=osclist.at(0); ourfreq=base; },
               { if ( (medianlist.at(count) < thresh2 ),
                   { ourosc.source=osclist.at(1); ourfreq=base*(2**thresh1); },
                   { if ( (medianlist.at(count) < thresh3),
                         { ourosc.source=osclist.at(2); ourfreq=base*(2**thresh2); },
```

```
                        { if ( (medianlist.at(count) < thresh4),
                             { ourosc.source=osclist.at(3); ourfreq=base*(2**thresh3); },
                             { if ( (medianlist.at(count) < thresh5),
                                { ourosc.source=osclist.at(4); ourfreq=base*(2**thresh4); },
                                { ourosc.source=osclist.at(5); ourfreq=base*2; } )} )} )} )}
) } };
  ourosc
  +
  // KLANG MAPS THE LAST [MEDARRAYLENGTH] MEDIAN FILTER VALUES TO PITCHES, EACH AT A LOWER
AMPLITUDE THAN THE LAST
  Spawn.ar({ arg spawn, i, synth;

    mediandisplay.value = medianlist.at(i);

    if ( (i > medarraylength),
      {
      backtrack.value(i, medianlist, currentPoints, 256);
      EnvGen.ar(medarraypercenv,
        Klang.ar( `[ currentPoints, medamps, nil ], 1, 0, medianvol.kr ) )
      },
      { nil } );
      },  1, timedelta, medianlist.size-1)
  +
  // NN INTERVALS, WAVELET AND HILBERT TRANSFORMS
  Spawn.ar({ arg spawn, i, synth;

    nndisplay.value = nnlist.at(i);

    // WAVELET: PHASEMOD PAIR
    Pan2.ar(
      EnvGen.ar(percenv, PMOsc.ar(wavepitches.at(i), wavepitches.at(i)*5, 3, 0, waveVol.kr)),
      wavelist.at(i)*2)
    +
    // HILBERT:  WAVETABLE
    EnvGen.ar(medarraypercenv, Osc.ar(squaretable, hilbpitches.at(i), 0, hilbvol.kr))
    +
    // NN INTERVALS 1:  WAVETABLE
    EnvGen.ar(sineenv, Osc.ar(glasstable, nnpitches.at(i), 0, nnvol.kr))
    +
    // NN INTERVALS + MEDIAN FILTER:  PHASE MOD PAIR, C SET BY NNPITCH, M:C RATIO SET BY
MEDIAN
    EnvGen.ar(sineenv,
      PMOsc.ar(nnpitches.at(i), medianratios.at(i), 1, 0, nn_medVol.kr))

    }, 2, timedelta, nnpitches.size-1)
  +
  // NN50 VALUES ARE AUDIFIED BY A TINKLING SOUND, PHASE MOD PAIR WITH HIGH M:C
  Spawn.ar({ arg spawn, i, synth;
      var nndiff, pmvol;
      if ( i > 0,
        { nndiff = nnlist.at(i) - nnlist.at(i-1);
          if ( abs(nndiff) > 0.05,
               { pmvol=0.25 }, { pmvol=0 });
            EnvGen.ar(percenv,
              PMOsc.ar(nnpitches.at(i), nnpitches.at(i)*15, 3, 0, pmvol*nn50vol.kr));
          });
        }, 1, timedelta, nnlist.size-1 )
  +
  Spawn.ar({ arg spawn, i, synth;
      var oscfreq, oscvol;
      if ( (i >= midlength) && (i < iMax),
        { oscfreq=medpitch.value(i); oscvol=medWinVol.kr },
        { oscfreq=0; oscvol=0; } );
        EnvGen.ar(sineenv,
            Osc.ar(f3456table, oscfreq, 0, oscvol))
      }, 1, timedelta, nnlist.size-1 )

  },medianlist.size*timedelta+0.5);

w.close
)
```

2. General Model

```
HRV SONIFICATION:  GENERAL MODEL

VARIABLE RATE PLAYBACK
NN INTERVALS, NN50 INTERVALS,
SINE TONE SOUNDS WHEN MEDIAN VALUE IS OVER THE THRESHOLD
MEAN IS REPRESENTED BY THE PITCH OF A SQUARE WAVE
STANDARD DEVIATION IS REPRESENTED BY VIBRATO RATE AND
  # HARMONICS IN A BLIP

(
// GENERAL GLOBAL VARIABLES
var glasstable, metaltable, sawtable, f3456table, fplushi, squaretable, maxLength;
var windowLength, halfWindow;
var rateSlider, rateView;

// NN INTERVAL VARIABLES
var nnlist, nnpitches;
var nnvol;
var nndisplay;

// NN50 VARIABLES
var nn50vol;

// MEAN VARIABLES
var meanList, meanPitches, meanVol, meanSlider, meanView;

// STANDARD DEVIATION VARIABLES
var sdList, sdWorking, sdVol, sdView, sdosc;

// SET WINDOW SIZE
windowLength = 300;
halfWindow = (windowLength/2).asInt;

// SET WAVETABLES
glasstable=Wavetable.sineFill(512, [1, 0, 0, 0.2, 0, 0, 0, 0, 0.1, 0, 0, 0.1, 0, 0, 0.1, 0, 0, 0, 0, 0, 0.1]);
metaltable = Wavetable.sineFill(512, [1,0.75, 0.5, 0.25, 0.1, 1, 0.75, 0.5, 0.25, 0.1]);
sawtable = Wavetable.sineFill(512, 1/[1, 2, 3, 4, 5, 6, 7]);
f3456table = Wavetable.sineFill(512, [1, 0, 1, 1, 1, 1]);
fplushi = Wavetable.sineFill(512, [0.3, 0, 0, 0, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]);
squaretable = Wavetable.sineFill(512, [1, 0, 0.3, 0, 0.2, 0, 0.143, 0, 0.111]);

// SET NN INTERVAL VARIABLES
nnlist = thisProcess.interpreter.executeFile(":slp37:slp37_nn");
nnpitches=Array.fill(nnlist.size, 0);
nnlist.do({ arg item, i; nnpitches.put(i, 128*(2**(1/item))) });

// SET MEAN LIST
meanList = thisProcess.interpreter.executeFile(":slp37:slp37_mean40");
meanPitches=Array.fill(meanList.size, 0);
meanList.do({ arg item, i; meanPitches.put(i, 128*(2**(1/item))) });

// SET STANDARD DEVIATION LIST
sdList = thisProcess.interpreter.executeFile(":slp37:slp37_sd40");
sdWorking=Array.fill(sdList.size, 0);
sdList.do({ arg item, i; sdWorking.put(i, item*40) });

// FIND THE LONGEST LIST
maxLength=max(nnlist.size, max(meanList.size, sdList.size));

// SET GUI
w = GUIWindow.new("panel", Rect.newBy( 176, 77, 313, 339 ));
  StringView.new( w, Rect.newBy( 11, 8, 71, 18 ), "NN Int");
  nndisplay=NumericalView.new( w, Rect.newBy( 13, 29, 64, 20 ), "NumericalView", 0.908, -1e+10, 1e+10, 0,
'linear');
  nnvol=SliderView.new( w, Rect.newBy( 13, 62, 128, 20 ), "SliderView", 0.24, 0, 0.5, 0, 'linear');
  StringView.new( w, Rect.newBy( 149, 62, 128, 20 ), "Beat-to-beat");
  meanSlider=SliderView.new( w, Rect.newBy( 13, 86, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0, 'linear'); //
0.035
  StringView.new( w, Rect.newBy( 149, 86, 128, 20 ), "Mean");
  nn50vol=SliderView.new( w, Rect.newBy( 13, 117, 128, 20 ), "SliderView", 0.096, 0, 0.5, 0, 'linear');
  StringView.new( w, Rect.newBy( 149, 117, 128, 20 ), "NN50");
  sdVol=SliderView.new( w, Rect.newBy( 13, 165, 128, 20 ), "SliderView", 0.0, 0, 0.5, 0, 'linear'); // 0.026
  StringView.new( w, Rect.newBy( 149, 165, 128, 20 ), "SD");
  StringView.new( w, Rect.newBy( 161, 8, 55, 18 ), "Mean");
  meanView=NumericalView.new( w, Rect.newBy( 160, 28, 58, 21 ), "NumericalView", 0.086976, -1e+10, 1e+10, 0,
'linear');
  StringView.new( w, Rect.newBy( 227, 7, 71, 18 ), "Std Dev");
  sdView=NumericalView.new( w, Rect.newBy( 227, 28, 73, 21 ), "NumericalView", 0.365631, -1e+10, 1e+10, 0,
'linear');
  rateSlider=SliderView.new( w, Rect.newBy( 12, 204, 128, 20 ), "SliderView", 60, 1, 80, 1, 'linear');
  rateView=NumericalView.new( w, Rect.newBy( 146, 204, 64, 20 ), "NumericalView", 60, -1e+10, 1e+10, 0,
'linear');
  StringView.new( w, Rect.newBy( 12, 227, 128, 20 ), "Beats per Second");

  rateView.action = { rateSlider.value = rateView.value };
  rateSlider.action = { rateView.value = rateSlider.value };
```

```
Synth.play({ arg synth;
  var percenv;
  var meanOsc, meanFreq;
  var glassosc, metalosc, sawosc, f3456osc, fplushiosc;
  var sdLevel, blipVol, medMult, oscPlay;

  // VOLUME AND FREQUENCY PLUGS FOR MEAN, STANDARD DEVIATION:
  meanFreq=Plug.kr(50);
  meanVol=Plug.kr(0);
  medMult=Plug.kr(0);
  sdLevel=Plug.kr(100);
  blipVol=Plug.kr(0);
  // LEVELFUNC PLUG, USED IN THE PAUSE.AR TO SHUT THESE OFF
  oscPlay=Plug.kr(1);

  // LIBRARY OF WAVETABLES
  glassosc=Osc.ar(glasstable, meanFreq, 0, meanVol);
  metalosc=Osc.ar(metaltable, meanFreq, 0, meanVol);
  sawosc=Osc.ar(sawtable, meanFreq, 0, meanVol);
  f3456osc=Osc.ar(f3456table, meanFreq, 0, meanVol);
  fplushiosc=Osc.ar(fplushi, meanFreq, 0, meanVol);

  meanOsc=glassosc;

  // REPEAT FUNCTION FOR MEAN, STANDARD DEVIATION
  // MEAN IS A PITCH, REPRESENTING MEAN OF THE LAST 300 VALUES
  // THE STANDARD DEVIATION IS MAPPED TO A BLIP: TO ITS # OF HARMONICS AND TO ITS
  // VIBRATO RATE
  // CURRENT NN INTERVAL IS IN THE MIDDLE OF THIS WINDOW
  // SO THE MEAN PITCH IS THE CURRENT COUNT + 150
  // THE BLIP IS ALSO SILENT UNTIL 150 VALUES HAVE BEEN READ
  // THE MEDIAN IS AN UNDULATING SET OF SINE OSCILLATORS WHICH SOUND WHEN THE THRESHOLD IS EXCEEDED
  // WHEN THE MEDIAN OSC SOUNDS, THE MEAN AND SD COME UP IN LEVEL A BIT (IF THEIR LEVEL ISN'T ZERO)
  synth.trepeatN(0, { 1/(rateSlider.poll) }, maxLength-1,
         { arg synth, now, count;
           var theMeanVol, standDevvol;

           theMeanVol=meanSlider.poll;
           standDevvol=sdVol.poll;
               if ( ( count > halfWindow),
               ( meanFreq.source = meanPitches.clipAt(count + halfWindow);
                 meanVol.source = theMeanVol;
                 sdLevel.source=sdWorking.clipAt(count + halfWindow);
                 blipVol.source = standDevvol },
                ( meanVol.source = 0;
                  sdLevel.source=1;
                  blipVol.source = 0 ) );
           ), { oscPlay.source = 0; Synth.stop } );
  Pause.ar((
    Blip.ar(meanFreq, sdLevel*10.asInt, SinOsc.kr(sdLevel, 0, blipVol))
    +
    meanOsc
    ), oscPlay)
  +
  // NN INTERVALS
  Spawn.ar({ arg spawn, i, synth;
    var dur, nndiff, pmvol;

    // SET NEXTTIME AND ENVELOPE BY RATESLIDER POSITION
    dur = 1/(rateSlider.poll);
    spawn.nextTime = dur;
    percenv=Env.perc(dur*0.1, dur*4.9, 1, -4);

    // DISPLAY VALUES FOR NN INTERVAL, MEDIAN VALUE, STANDARD DEVIATION
    nndisplay.value = nnlist.at(i);
    sdView.value = sdList.clipAt(i + halfWindow);
    meanView.value = meanList.clipAt(i + halfWindow);

    // TEST FOR NN50
    if ( i > 0,
      { nndiff = nnlist.at(i) - nnlist.at(i-1);
        if ( abs(nndiff) > 0.05,
          { pmvol=0.25 }, { pmvol=0 });
       }, { pmvol=0 });

    // NN INTERVAL MAPPED TO SINGRAIN FREQUENCY
    PSinGrain.ar(nnpitches.at(i), dur*2, nnvol.kr)
    +
    // NN50 VALUES ARE AUDIFIED BY A TINKLING SOUND, PHASE MOD PAIR WITH HIGH M:C
    EnvGen.ar(percenv,
        PMOsc.ar(nnpitches.at(i), nnpitches.at(i)*15, 3, 0, pmvol*nn50vol.kr));
      }, 1, nil, nnpitches.size-1)
  });
 w.close
)
```

3. Apnea Diagnosis Model

```
VARIABLE RATE PLAYBACK
NN INTERVALS, NN50 INTERVALS,
SINE TONE SOUNDS WHEN MEDIAN VALUE IS OVER THE THRESHOLD
MEAN IS REPRESENTED BY THE PITCH OF A SQUARE WAVE .
STANDARD DEVIATION IS REPRESENTED BY VIBRATO RATE AND
  # HARMONICS IN A BLIP

(
// GENERAL GLOBAL VARIABLES
var glasstable, metaltable, sawtable, f3456table, fplushi, squaretable, twentytable, newtable,
maxLength;
var windowLength, halfWindow;
var rateSlider, rateView, rateinit;
var filename;

// NN INTERVAL VARIABLES
var nnlist, nnfilename, nnpitches, nninit;
var nnvol;
var nndisplay;

// NN50 VARIABLES
var nn50vol, nn50init;

// MEDIAN FILTERED HILBERT TRANSFORM VARIABLES
var medianlist, medianfilename, medianthreshold, medianinit;
var mediandisplay;
var medVol;
var medianOsc;
var medSlider;

// var derivativeSlider, derivativeVol, derivativeinit;

// MEAN VARIABLES
var mean5List, mean5filename, mean5Pitches, mean5Vol, mean5Slider, mean5View, mean5init;
// var mean10List, mean10filename, mean10Pitches, mean10Vol, mean10Slider, mean10View,
mean10init;
var mean15List, mean15filename, mean15Pitches, mean15Vol, mean15Slider, mean15View,
mean15init;
// var mean20List, mean20filename, mean20Pitches, mean20Vol, mean20Slider, mean20View,
mean20init;

// STANDARD DEVIATION VARIABLES
var sdList, sdfilename, sdWorking, sdVol, sdView, sdosc, sdinit, sdhalfwindow;

//TIME VARIABLES
var timefile, timefilename, hrdisp, mindisp, secdisp, timeSlider, currentTime, sonStop;

// SET SOURCE FILE
filename="slp04";

// SET WINDOW SIZES
windowLength = 15;
halfWindow = (windowLength/2).asInt;
sdhalfwindow = 150;

// SET WAVETABLES
glasstable=Wavetable.sineFill(512, [1, 0, 0, 0.2, 0, 0, 0, 0, 0.1, 0, 0, 0.1, 0, 0, 0.1, 0, 0,
0, 0, 0, 0.1]);
metaltable = Wavetable.sineFill(512, [1,0.75, 0.5, 0.25, 0.1, 1, 0.75, 0.5, 0.25, 0.1]);
sawtable = Wavetable.sineFill(512, 1/[1, 2, 3, 4, 5, 6, 7]);
f3456table = Wavetable.sineFill(512, [1, 0, 1, 1, 1, 1]);
fplushi = Wavetable.sineFill(512, [0.3, 0, 0, 0, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]);
squaretable = Wavetable.sineFill(512, [1, 0, 0.3, 0, 0.2, 0, 0.143, 0, 0.111]);
twentytable = Wavetable.sineFill(512, [ 0.447368, 0.25, 0.111, 0.0625, 0, 0, 0.166, 0, 0, 0,
  0, 0, 0, 0, 0, 0.05, 0.1]);
newtable = Wavetable.sineFill(512, [ 0.447368, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0.122807, 0, 0.0614035, 0, 0.1]);


// SET TIME FILE
timefilename=":"++filename++":"++filename++"_time";
timefile = thisProcess.interpreter.executeFile(timefilename);

// SET NN INTERVAL VARIABLES
nnfilename=":"++filename++":"++filename++"_nn";
nnlist = thisProcess.interpreter.executeFile(nnfilename);
nnpitches=Array.fill(nnlist.size, 0);
nnlist.do({ arg item, i; nnpitches.put(i, 128*(2**(1/item))) });

// SET MEDIAN FILTER VARIABLES
//medianlist=#[0];
medianfilename=":"++filename++":"++filename++"_median";
medianlist = thisProcess.interpreter.executeFile(medianfilename);
medianthreshold=0.5139;

// SET MEAN LIST
mean5filename=":"++filename++":"++filename++"_mean5";
mean5List = thisProcess.interpreter.executeFile(mean5filename);
```

```
mean5Pitches=Array.fill(mean5List.size, 0);
mean5List.do({ arg item, i; mean5Pitches.put(i, 128*(2**(1/(item.round(0.21)) ))) });
//.round(0.03)

mean15filename=":"++filename++":"++filename++"_mean15";
mean15List = thisProcess.interpreter.executeFile(mean15filename);
mean15Pitches=Array.fill(mean15List.size, 0);
mean15List.do({ arg item, i; mean15Pitches.put(i, 128*(2**(1/(item.round(0.01)) ))) });
//.round(0.03)


// SET STANDARD DEVIATION LIST
sdfilename=":"++filename++":"++filename++"_sd"++300;
sdList = thisProcess.interpreter.executeFile(sdfilename);
sdWorking=Array.fill(sdList.size, 0);
sdList.do({ arg item, i; sdWorking.put(i, item*40) });

// FIND THE LONGEST LIST
maxLength=max(max(nnlist.size, medianlist.size), max(mean5List.size, sdList.size));

nninit=0.0; //0.24;
mean5init=0.2; //0.0;
//derivativeinit=0.0; //0.0;
mean15init=0.1; //0.0;
//mean20init=0.0; //0.0;
nn50init=0.0; //0.096;
medianinit=0.3; //0.89;
sdinit=0.0;
rateinit=30;

// SET GUI
w = GUIWindow.new(filename, Rect.newBy(176, 77, 312, 449));
  StringView.new( w, Rect.newBy( 11, 8, 71, 18 ), "NN Int");
  StringView.new( w, Rect.newBy( 82, 8, 71, 18 ), "Median filt");
  StringView.new( w, Rect.newBy( 161, 8, 55, 18 ), "Mean");
  StringView.new( w, Rect.newBy( 227, 7, 71, 18 ), "Std Dev");
  nndisplay=NumericalView.new( w, Rect.newBy( 13, 29, 64, 20 ), "NumericalView", 0.908, -
1e+10, 1e+10, 0, 'linear');
  mediandisplay=NumericalView.new( w, Rect.newBy( 82, 29, 73, 21 ), "NumericalView", 0.817987,
-1e+10, 1e+10, 0, 'linear');
  nnvol=SliderView.new( w, Rect.newBy( 13, 62, 128, 20 ), "SliderView", nninit, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy( 149, 62, 128, 20 ), "Beat-to-beat");
  nn50vol=SliderView.new( w, Rect.newBy(13, 93, 128, 20), "SliderView", nn50init, 0, 0.5, 0,
'linear');
  StringView.new( w, Rect.newBy(149, 93, 128, 20), "NN50");
  mean5Slider=SliderView.new( w, Rect.newBy(13, 127, 128, 20), "SliderView", mean5init, 0,
0.5, 0, 'linear'); // 0.035
  StringView.new( w, Rect.newBy(149, 127, 128, 20), "Mean5");
  mean15Slider=SliderView.new( w, Rect.newBy(13, 175, 128, 20), "SliderView", mean15init, 0,
0.5, 0, 'linear');
  StringView.new( w, Rect.newBy(149, 175, 128, 20), "Mean15");
  sdVol=SliderView.new( w, Rect.newBy(13, 232, 128, 20), "SliderView", sdinit, 0, 0.5, 0,
'linear'); // 0.026
  StringView.new( w, Rect.newBy(149, 232, 128, 20), "SD");
  medSlider=SliderView.new( w, Rect.newBy(13, 259, 128, 20), "SliderView", medianinit, 0, 2.0,
0, 'linear');
  StringView.new( w, Rect.newBy(149, 259, 128, 20), "Median");
  mean15View=NumericalView.new( w, Rect.newBy( 160, 28, 58, 21 ), "NumericalView", 0.086976, -
1e+10, 1e+10, 0, 'linear');
  sdView=NumericalView.new( w, Rect.newBy( 227, 28, 73, 21 ), "NumericalView", 0.365631, -
1e+10, 1e+10, 0, 'linear');
  rateSlider=SliderView.new( w, Rect.newBy(12, 300, 128, 20), "SliderView", rateinit, 1, 120,
1, 'linear');
  rateView=NumericalView.new( w, Rect.newBy(146, 300, 64, 20), "NumericalView", rateinit, -
1e+10, 1e+10, 0, 'linear');
  StringView.new( w, Rect.newBy(12, 323, 128, 20), "Beats per Second");
  StringView.new( w, Rect.newBy(121, 387, 80, 19), "Elapsed time");
  hrdisp=NumericalView.new( w, Rect.newBy(203, 386, 31, 21), "NumericalView", 0, -1e+10,
1e+10, 0, 'linear');
  StringView.new( w, Rect.newBy(201, 412, 27, 16), "Hrs");
  mindisp=NumericalView.new( w, Rect.newBy(236, 386, 31, 21), "NumericalView", 0, -1e+10,
1e+10, 0, 'linear');
  StringView.new( w, Rect.newBy(235, 411, 27, 18), "Min");
  secdisp=NumericalView.new( w, Rect.newBy(269, 386, 31, 21), "NumericalView", 0, -1e+10,
1e+10, 0, 'linear');
  StringView.new( w, Rect.newBy(268, 411, 26, 18), "Sec");
  timeSlider=SliderView.new( w, Rect.newBy(76, 360, 224, 18), "SliderView", 0, 0, nnlist.size,
1, 'linear');
  sonStop=CheckBoxView.new( w, Rect.newBy(13, 362, 55, 15), "", 1, 0, 1, 0, 'linear');
  StringView.new( w, Rect.newBy(12, 381, 69, 16), "Un-check");
  StringView.new( w, Rect.newBy(12, 398, 70, 16), "to pause");

  rateView.action = { rateSlider.value = rateView.value };
  rateSlider.action = { rateView.value = rateSlider.value };

  // IF THE CHECKBOX IS UNCHECKED, THE TIME WINDOWS TRACK THE USER'S SLIDER MOVEMENTS
  timeSlider.action = { if ( (sonStop.value == 0),
            { hrdisp.value = timefile.at(timeSlider.value.asInt).at(0);
              mindisp.value = timefile.at(timeSlider.value.asInt).at(1);
              secdisp.value = timefile.at(timeSlider.value.asInt).at(2); }) };
```

```
Synth.play({ arg synth;
  var percenv;
  var mean50sc, mean5Freq;
  //var derivative;
  var mean150sc, mean15Freq;
  //var mean200sc, mean20Freq;
  var glassosc, metalosc, sawosc, f3456osc, fplushiosc;
  var sdLevel, blipVol, medMult;

  // VOLUME AND FREQUENCY PLUGS FOR MEAN, STANDARD DEVIATION AND MEDIAN:
  mean5Freq=Plug.kr(50);
  mean5Vol=Plug.kr(0);
  //derivative=Plug.kr(10);
  //derivativeVol=Plug.kr(0);
  mean15Freq=Plug.kr(50);
  mean15Vol=Plug.kr(0);
  //mean20Freq=Plug.kr(50);
  //mean20Vol=Plug.kr(0);
  medVol=Plug.kr(0);
  medMult=Plug.kr(0);
  sdLevel=Plug.kr(100);
  blipVol=Plug.kr(0);

  // LIBRARY OF WAVETABLES
  glassosc=Osc.ar(glasstable, mean15Freq, 0, mean15Vol);
  metalosc=Osc.ar(squaretable, mean5Freq, 0, mean5Vol);
  //sawosc=Osc.ar(sawtable, meanFreq, 0, meanVol);
  //f3456osc=Osc.ar(newtable, mean10Freq, 0, mean10Vol);
  //fplushiosc=Osc.ar(twentytable, mean20Freq, 0, mean20Vol);

  mean50sc=metalosc;
  //mean100sc=f3456osc;
  mean150sc=glassosc;
  //mean200sc=fplushiosc;

  medianOsc=Mix.ar(
    SinOsc.ar([400, 1100, 600], 0, SinOsc.kr([ 0.3, 0.4, 0.25 ],
                                [ 0, 3pi/5, 6pi/11],
                              0.05, 0.1)*medVol)
            );

  // REPEAT FUNCTION FOR TIME UPDATE,
  // MEAN, STANDARD DEVIATION AND MEDIAN
  // TIMESLIDER IS POLLED, ITS CURRENT POSITION IS THE CURRENT INDEX FOR ALL LISTS.
  // MEAN IS A PITCH, REPRESENTING MEAN OF THE LAST 300 VALUES
  // THE STANDARD DEVIATION IS MAPPED TO A BLIP: TO ITS # OF HARMONICS AND TO ITS VIBRATO RATE
  // CURRENT NN INTERVAL IS IN THE MIDDLE OF THIS WINDOW
  // SO THE MEAN PITCH IS THE CURRENT COUNT + 150
  // THE BLIP IS ALSO SILENT UNTIL 150 VALUES HAVE BEEN READ
  // THE MEDIAN IS AN UNDULATING SET OF SINE OSCILLATORS WHICH SOUND WHEN THE THRESHOLD IS
EXCEEDED
  // WHEN THE MEDIAN OSC SOUNDS, THE MEAN AND SD COME UP IN LEVEL A BIT (IF THEIR LEVEL ISN'T
ZERO)
  synth.trepeat(0, { 1/(rateSlider.poll) },
          { arg synth, now, count;
            var theMean5Vol, theDerivativeVol, theMean15Vol,theMean20Vol, theMedVol,
standDevvol;
            currentTime = timeSlider.poll.asInt;
            if ( (currentTime < (maxLength-1)),
            {
            theMean5Vol=mean5Slider.poll;
           // theDerivativeVol=derivativeSlider.poll;
            theMean15Vol=mean15Slider.poll;
           // theMean20Vol=mean20Slider.poll;
            theMedVol=medSlider.poll;
            standDevvol=sdVol.poll;
            if (( sonStop.value == 1),
               { timeSlider.value = timeSlider.value + 1 });
            if ( ( currentTime > 2),
                { mean5Freq.source = mean5Pitches.clipAt(currentTime + 2);
                  mean5Vol.source = theMean5Vol;

                    },
                { mean5Vol.source = 0; } );
               if ( ( currentTime > 7),
                { mean15Freq.source = mean15Pitches.clipAt(currentTime + 7);
                  mean15Vol.source = theMean15Vol;
                      },
                { mean15Vol.source = 0; } );
           if ( ( currentTime > sdhalfwindow),
                { sdLevel.source=sdWorking.clipAt(currentTime + sdhalfwindow);
                  blipVol.source = standDevvol },
                { sdLevel.source=1;
                  blipVol.source = 0 } );
            if ( ( medianlist.clipAt(currentTime) > medianthreshold),
                { medVol.source = theMedVol;
                if ( ( mean5Slider.value > 0 ) && ( medSlider.value > 0) && ( currentTime > 2),
                      { mean5Vol.source = theMean5Vol+0.02; });
                  if ( ( mean15Slider.value > 0 ) && ( medSlider.value > 0) && ( currentTime > 7),
                      { mean15Vol.source = theMean15Vol+0.02; });
```

```
            if ( ( sdVol.value > 0 ) && ( medSlider.value > 0) &&
                                                ( currentTime > sdhalfwindow),
                    { blipVol.source = standDevvol*0.02; }) },
                { medVol.source = 0; } );
          }, { sonStop.value = 0; Synth.stop }) } );
  Pause.ar({
    Blip.ar(mean15Freq, sdLevel*10.asInt, SinOsc.kr(sdLevel, 0, blipVol))
    //+
    //Blip.ar(mean15Freq, derivative, derivativeVol)
    +
    mean50sc + mean150sc //+ mean200sc
    +
    medianOsc
  +
  // NN INTERVALS
  Spawn.ar({ arg spawn, i, synth;
    var dur, nndiff, currentnn, pmvol;
    // SET NEXTTIME AND ENVELOPE BY RATESLIDER POSITION
    dur = 1/(rateSlider.poll);
    spawn.nextTime = dur;
    percenv=Env.perc(dur*0.1, dur*4.9, 1, -4);
    currentnn = nnpitches.clipAt(currentTime);
    // DISPLAY VALUES FOR NN INTERVAL, MEDIAN VALUE, STANDARD DEVIATION, TIME
    nndisplay.value = nnlist.clipAt(currentTime);
    mediandisplay.value = medianlist.clipAt(currentTime); //(i+1).value;
    sdView.value = sdList.clipAt(currentTime + sdhalfwindow);
    mean15View.value = mean15List.clipAt(currentTime + halfWindow);
    hrdisp.value = timefile.clipAt(currentTime).at(0);
    mindisp.value = timefile.clipAt(currentTime).at(1);
    secdisp.value = timefile.clipAt(currentTime).at(2);

    // TEST FOR NN50
    if ( currentTime > 0,
      { nndiff = nnlist.clipAt(currentTime) - nnlist.clipAt(currentTime-1);
        if ( abs(nndiff) > 0.05,
          { pmvol=0.25 }, { pmvol=0 });
       }, { pmvol=0 });

    // NN INTERVAL MAPPED TO SINGRAIN FREQUENCY
    PSinGrain.ar(currentnn, dur*2, nnvol.kr)
    +
    // NN50 VALUES ARE AUDIFIED BY A TINKLING SOUND, PHASE MOD PAIR WITH HIGH M:C
    EnvGen.ar(percenv,
        PMOsc.ar(currentnn, currentnn*15, 3, 0, pmvol*nn50vol.kr));
      }, 1, nil)

  }, sonStop.kr)
  });
w.close
)
)
```

Appendix 6

Listening Perception Test Materials

1. Training Session

The sounds I am going to play for you today are from a research project involving the illustration of data sets with sound, rather than a visual graph. It's a new field of research called auditory display. The question is whether there are patterns in the data that are perceived just as well, if not better, by the ears than by the eyes.

The data explored by these displays represents heart rate variability. It is taken from a branch of cardiology that studies the changes in inter-heartbeat intervals, that is, how the speed at which the heart beats changes over time. The data is obtained by the patient wearing an ambulatory holter monitor that records the heart's electrical activity. After the recording, a beat recognition algorithm pinpoints the times of the QRS complex, which corresponds to the muscular contraction we call the heartbeat. The times of these events are retained, and the rest of the data discarded. What is left is a series of numbers representing each NN (normal to normal) interval, all within the range of one second, plus or minus a half second or so, each signifying the amount of time between each heart beat. Many cardiologists now feel that a great deal can be determined about a patient's condition by the changes the heart rate undergoes over time. There is not, however, general agreement about the best methods for interpreting this data, and many different methods are employed and interpretations proposed. I am developing an auditory display methodology for heart rate variability.

In the samples you will hear today, each inter-beat interval has been mapped to a pitch, which is played by a high-pitched humming timbre. Higher sound frequencies (pitches), are associated with faster heart rates, lower pitches are associated with a slower heart rate. The playback rate is sixty beats per second, so each second corresponds roughly to one minute of heart rate activity. So an auditory display that sounded like [vocalize glissando up] would indicate a heart that is beating faster and faster, while a sound like [vocalize gliss down] would indicate a heartbeat that is getting slower and slower. In addition, the larger interbeat increments, those exceeding 50ms, are given additional annotation. These intervals are audifies by a tinkling sound. So a display that consists only of a sound like [whistling] means that all of the changes are happening gradually,

and not in sudden jumps. A display that contains [pinging] indicates that the heart rate is changing in large bursts.

The object of today's test is to get a baseline idea of how successful the work is to date. We would like to find out whether four conditions are as clearly defined by an auditory representation as they are by a visual one. A series of examples will be played, each lasting ten seconds, representing approximately ten minutes of heart activity. Each example will illustrate one of four conditions. I will ask you to indicate which of the four conditions you think each sample represents. I will give you a brief explanation of each condition and play a sample of each in a moment. After the auditory displays, I will then show a series of visual graphs on the overhead projector, and ask you to try to classify them in the same manner.

First, it is important to stress that I am in no way testing your intelligence, your ears or your eyes. This test is designed to tell me how effective my work is to date, and that is all. There is no deception of any kind involved. The test will contain samples that correspond to the examples I will play for you, and nothing else. The test has been reviewed and approved by the Faculty of Music Ethics Review Committee. The results of the test will be entirely confidential. I would ask that you do not mark your papers in any way other than to fill in the selection boxes, in order to ensure that no identifying characteristics are present. Your participation is also completely optional. Anyone who is uncomfortable participating for any reason may leave at any time. The results of the test will be reported in this class within a week or two. I will happily answer any questions about the work or this procedure following the test.

Now let me explain what you will be listening to. The heart rate is determined the interactions of three components. The sinus node is the pacemaker, which produces a steady pulse at roughly 70 beats per minute. The pacemaker interacts with the autonomic nervous system, which has two components. The sympathetic nervous system produces a chemical that tends to speed up the heart rate, while the parasympathetic nerves produce a chemical that tends to slow it down. The result is that the heart rate is changing constantly. All of these examples were recorded at night, during sleep, when external factors are presumably minimized.

A normal, healthy heart rate fluctuates in a complex fashion, even in a person at rest. On your sheets there is an illustration of a graph of 600 NN intervals. Here

is an auditory display of a healthy subject. Notice that the heart rate is constantly in flux, with irregular tinkling sounds, representing higher NN intervals.

Condition two is congestive heart failure, which describes a condition when the ventricle is not pumping properly. This unhealthy condition is characterized by an extremely regular heartbeat. Notice in this example that the pitch hardly changes at all, and that the higher interval tinkling sound is virtually non-existent.

Condition three is atrial fibrillation, which occurs when the pacemaker no longer sets the rhythm of the heart. This is characterized by extreme irregularity. Notice the extremely erratic character of this sample, and the high number of large interbeat intervals.

Condition four is obstructive sleep apnea, which occurs in people whose breathing stops during sleep. Apneic episodes can occur off and on throughout the night, during which people repeatedly gasp for breath. The condition can be observed in the heart rate as the heart slows down while breathing stops, then speeds up again when breathing resumes, displaying a cycling between high and low heart rates. Here is an example of an apneic episode. Notice that in addition to the alternating high and low pitch, there are clumps of tinkles as well.

I will now play all four examples again.

I will now play twenty-four examples, each of which represents one of these four conditions. The examples are taken from different subjects. Please mark on your page which of the four conditions you feel each sample represents. Each sample will last ten seconds. You will have eight seconds between samples in which to make your selection.

[DO TEST]

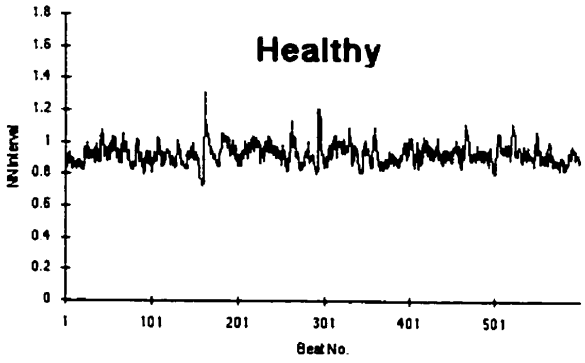
Thank you. Please pass your papers forward.

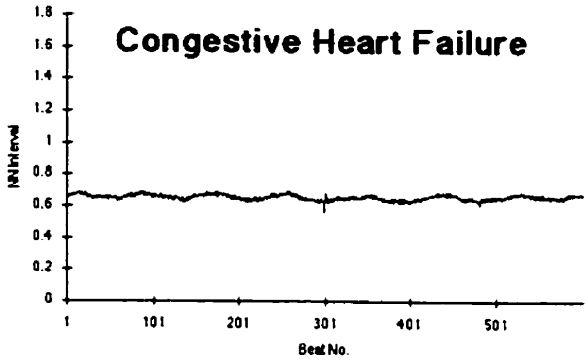
I will now distribute response sheets for the visual identifications.

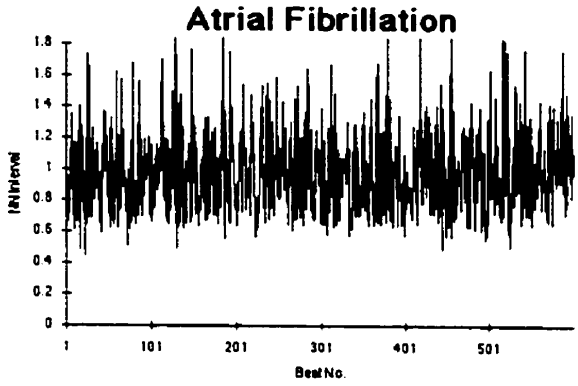
I will now ask you to do the same identification with visual graphs. The visual graphs are taken from the same subjects as the auditory displays were. I will project each graph for ten seconds, allowing you 8-10 seconds between projections to make your selection.

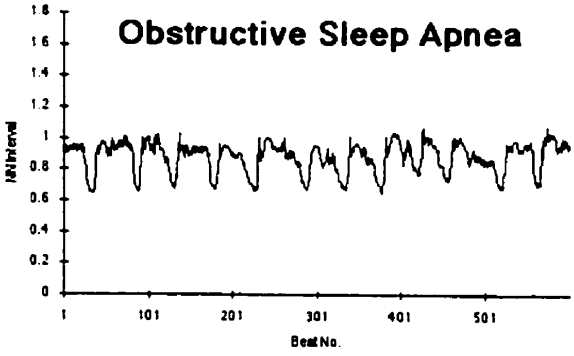
2. Response Forms

A. Auditory Condition response form





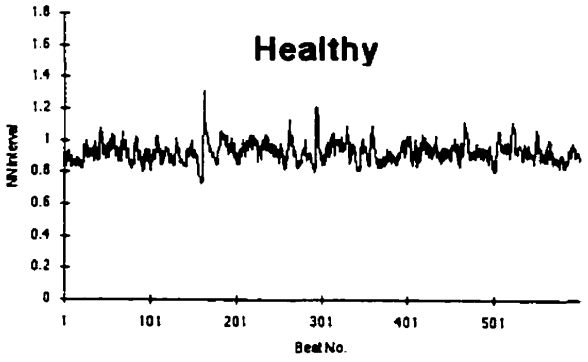


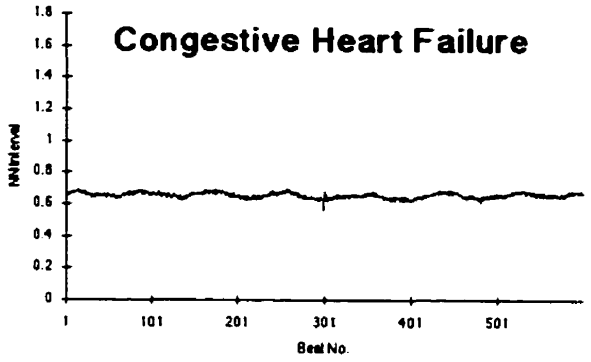


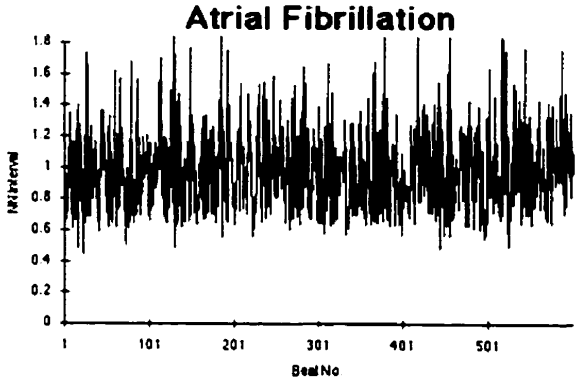
Twenty-four auditory displays of heart rate variability data will be played.
Each will represent one of the above four data types.
Please mark which type you think each selection represents.

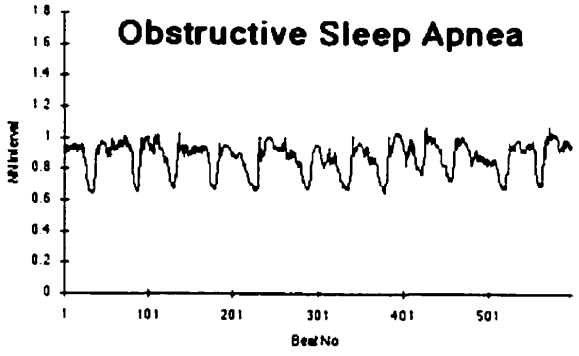
1.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
2.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
3.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
4.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
5.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
6.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
7.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
8.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
9.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
10.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
11.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
12.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
13.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
14.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
15.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
16.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
17.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
18.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
19.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
20.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
21.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
22.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
23.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
24.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea

B. Visual Condition response form





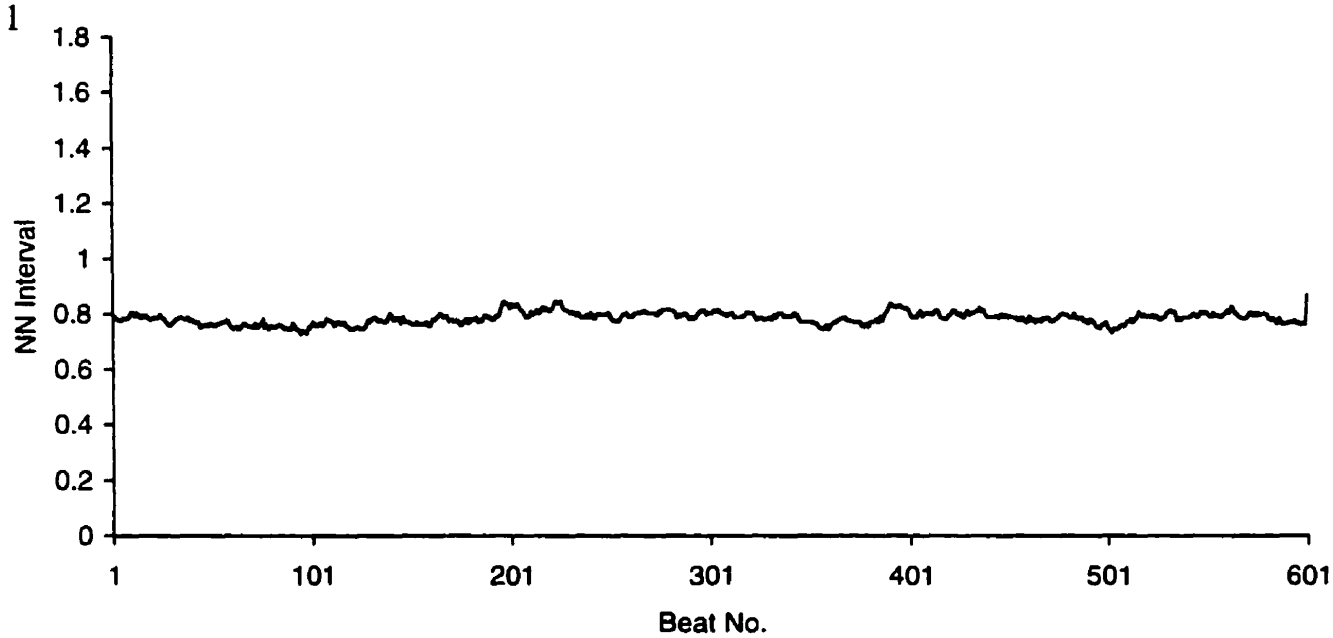




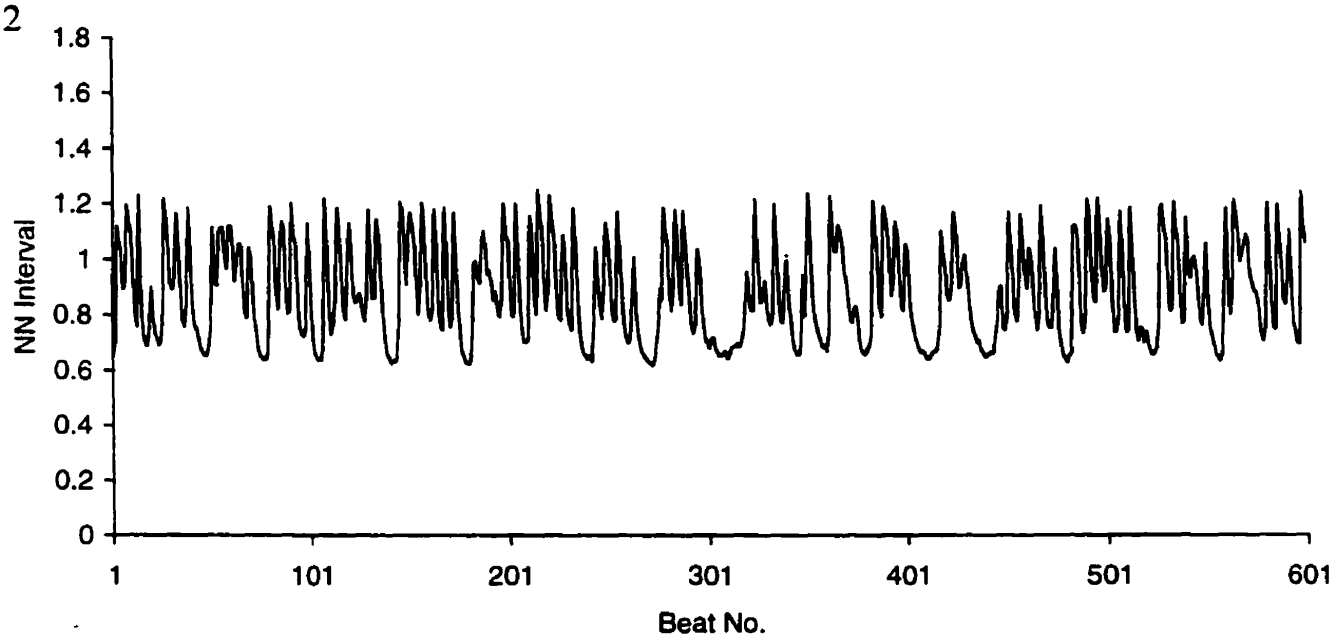
Twenty-four graphs of heart rate variability data will be shown.
Each will represent one of the above four data types.
Please mark which type you think each selection represents.

1.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
2.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
3.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
4.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
5.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
6.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
7.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
8.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
9.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
10.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
11.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
12.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
13.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
14.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
15.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
16.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
17.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
18.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
19.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
20.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
21.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
22.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
23.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea
24.	☐ Healthy	☐ Congestive Heart Failure	☐ Atrial Fibrillation	☐ Sleep Apnea

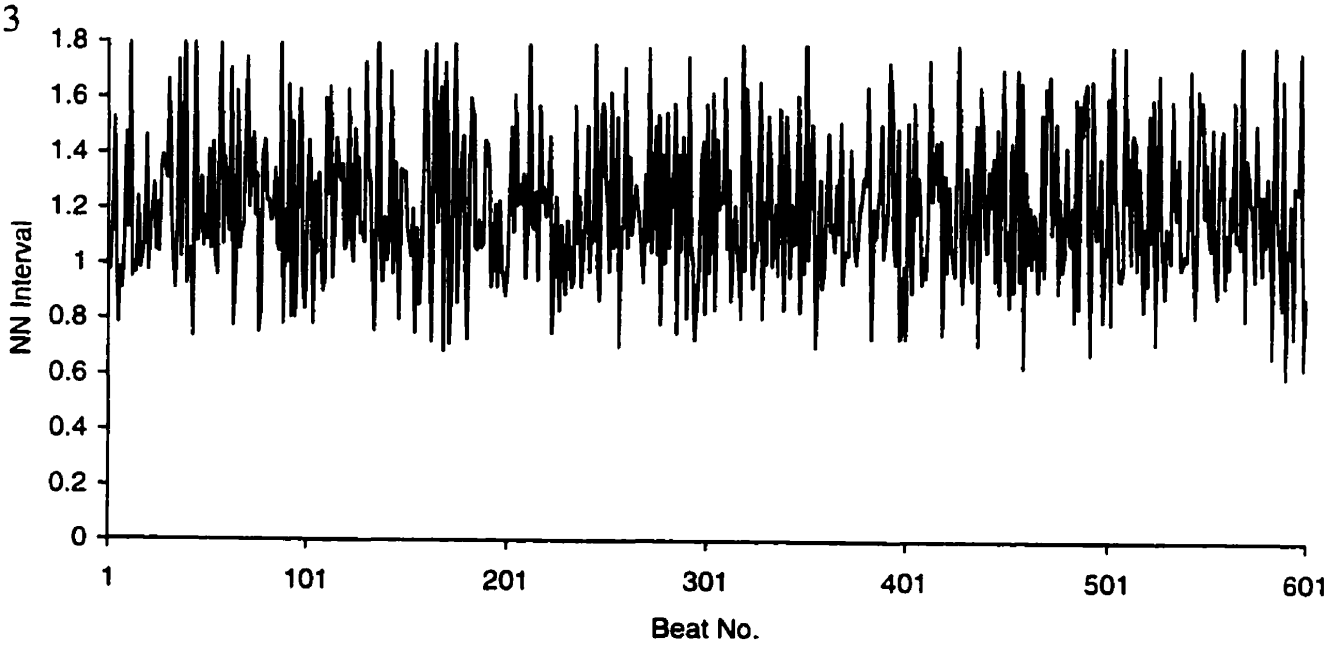
1


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

2


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

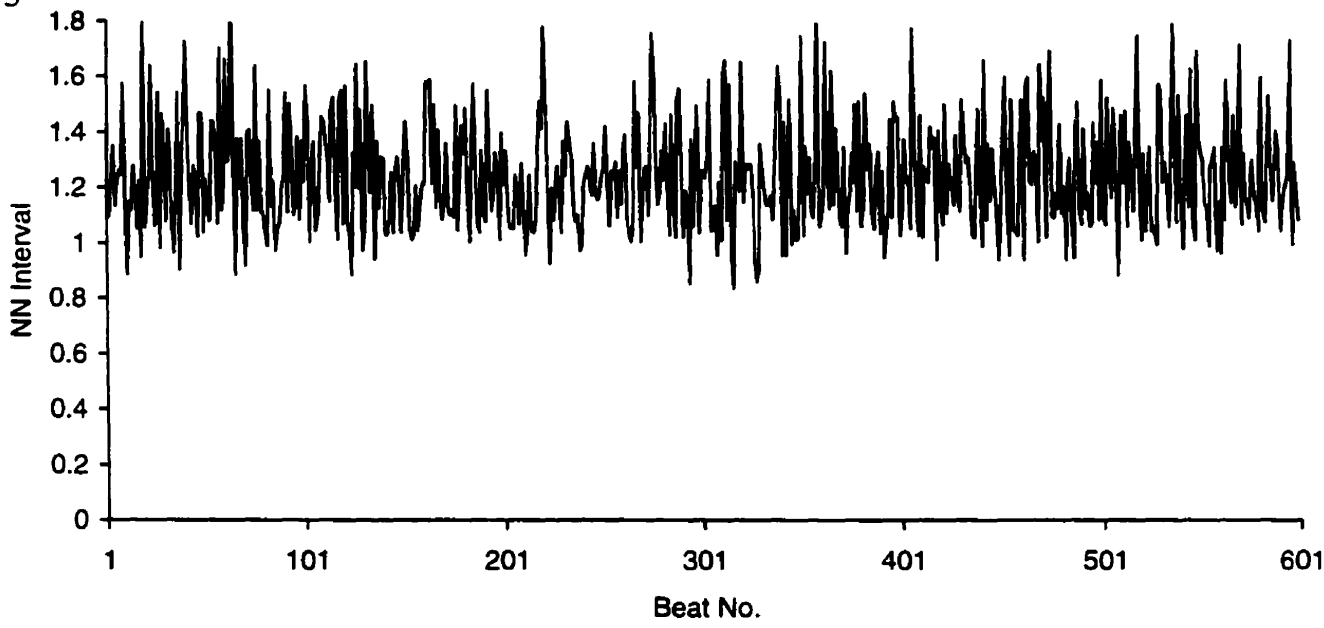
3


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

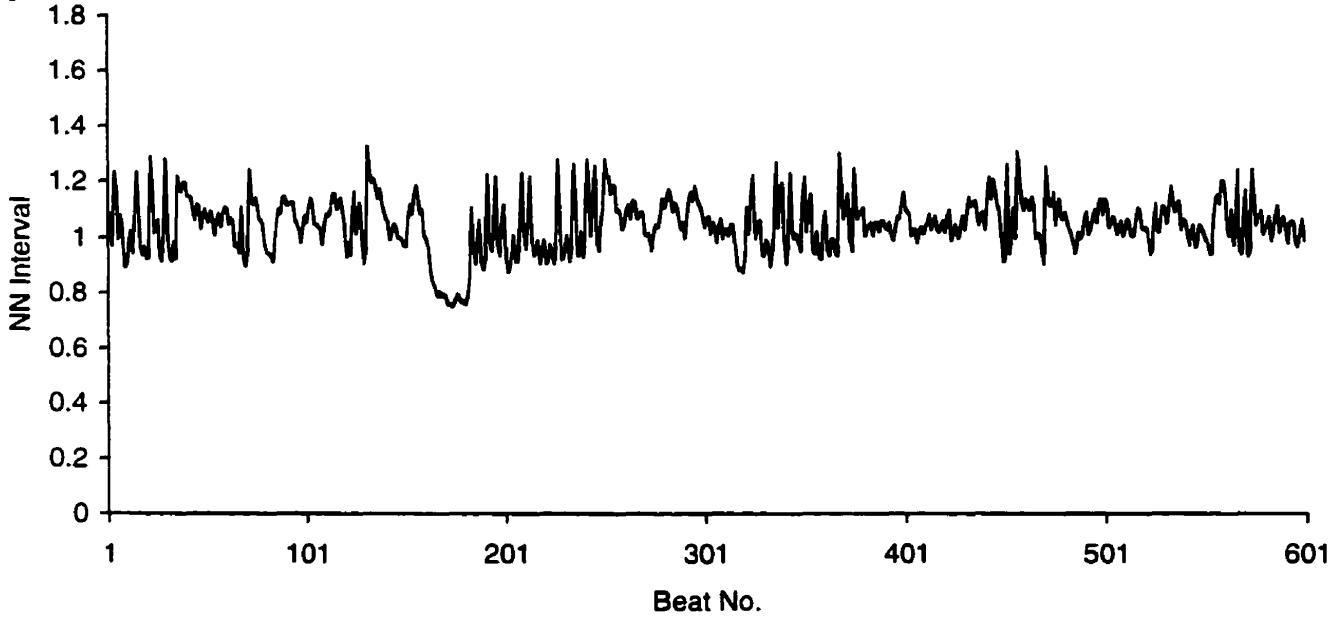
4


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

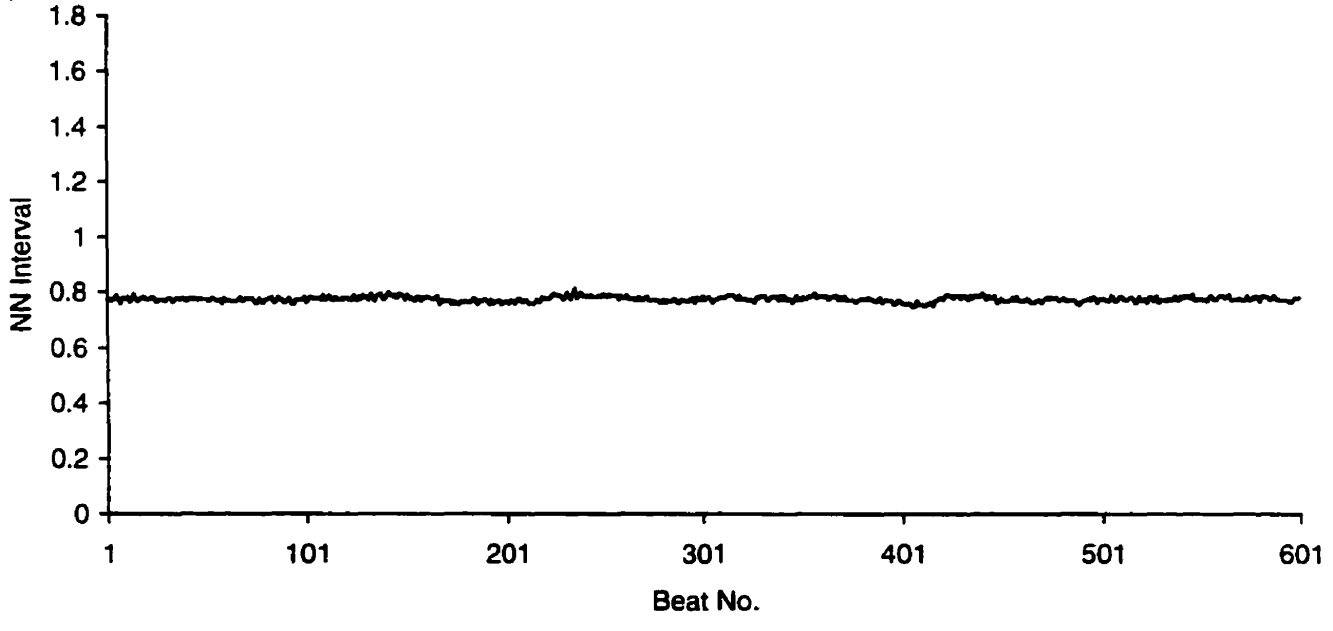
5


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

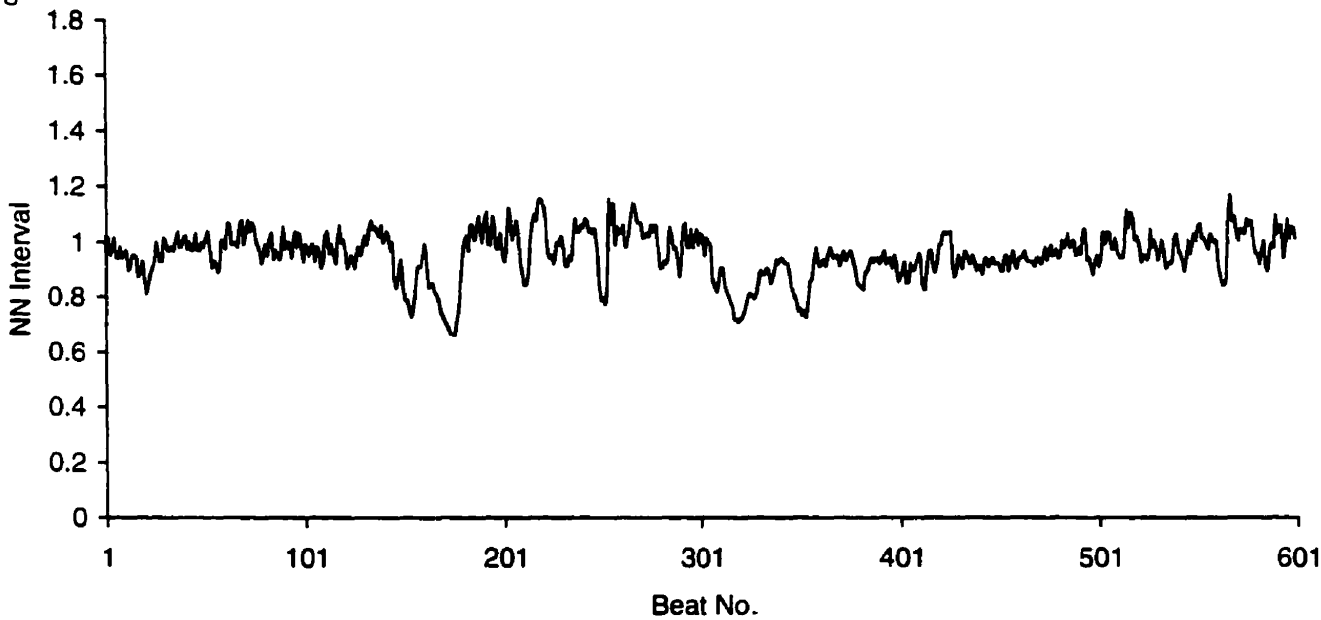
6


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

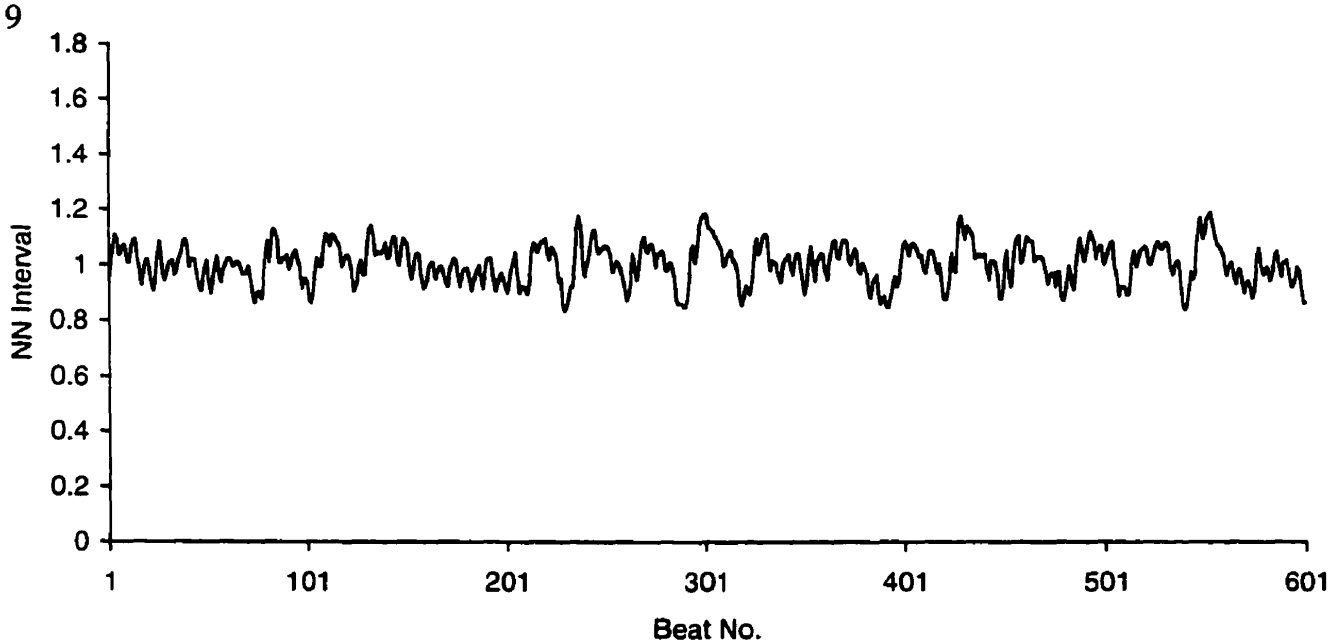
7

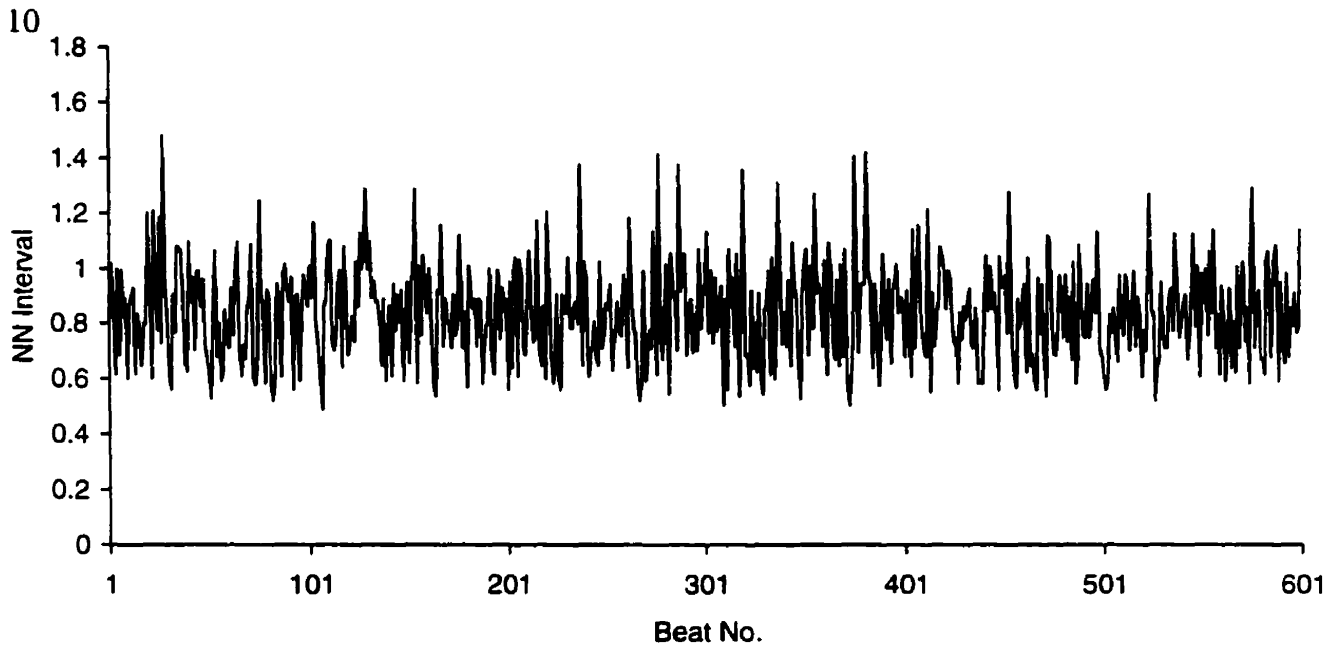

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

8

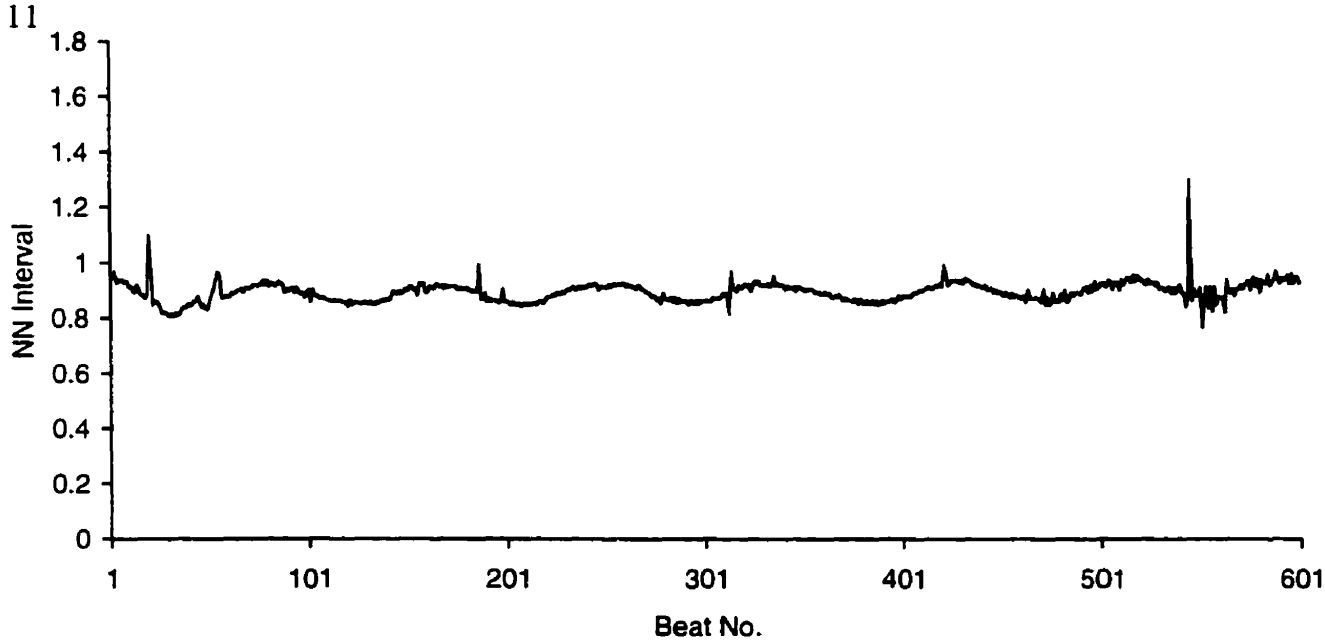

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

9

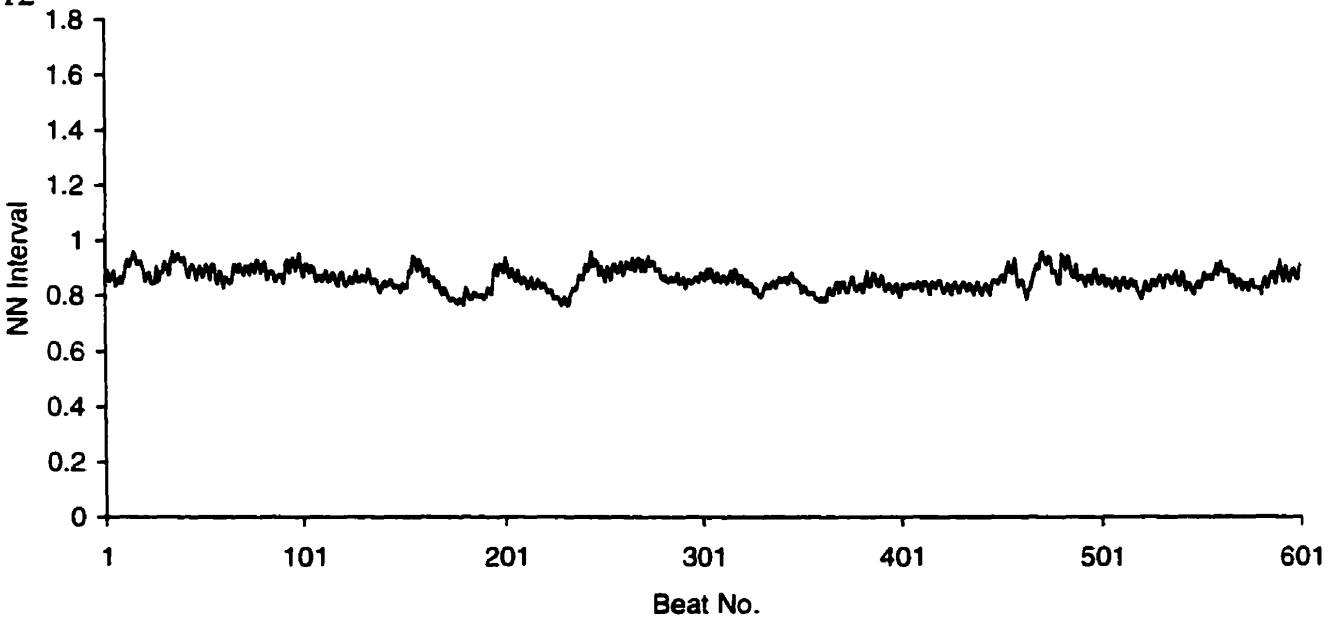

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.


10
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

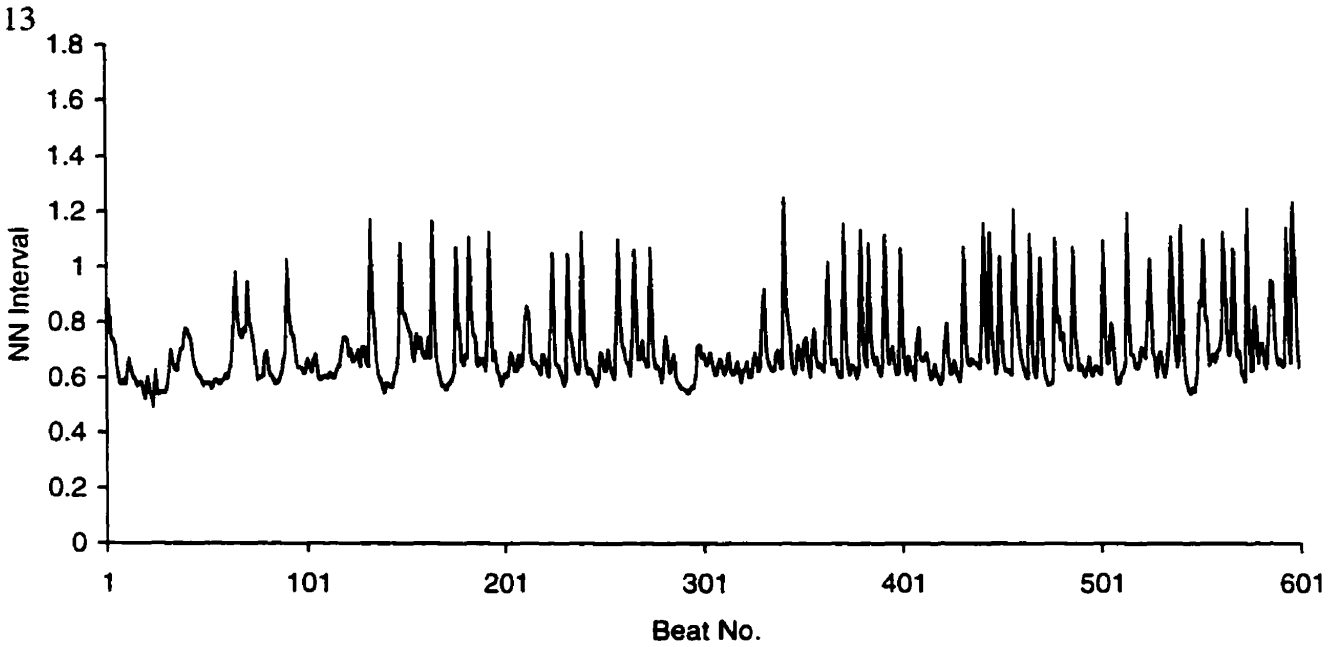
11


NN Interval
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
101
201
301
401
501
601
Beat No.

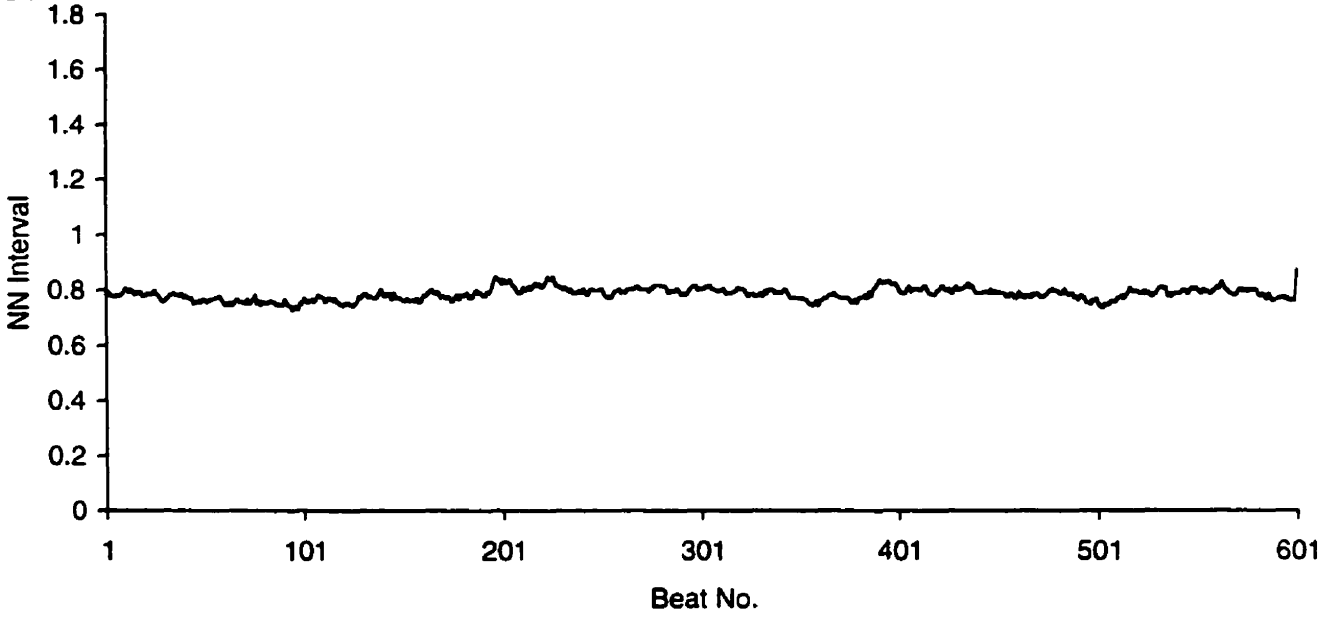
12


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

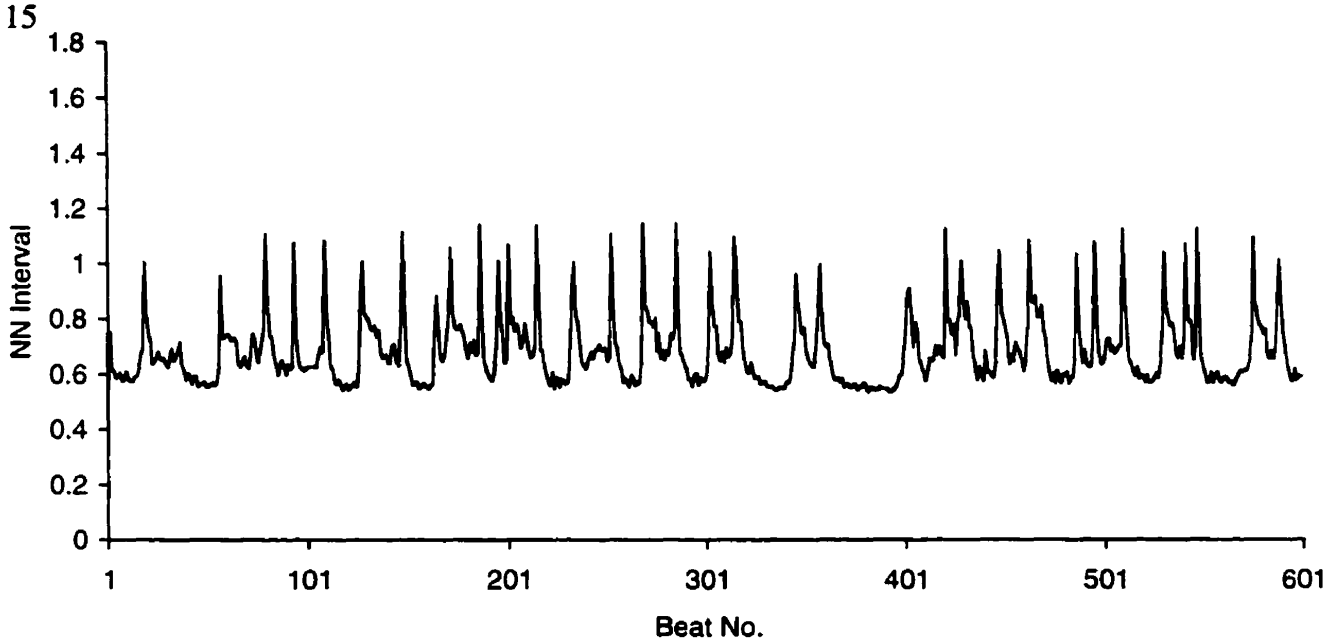
13


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

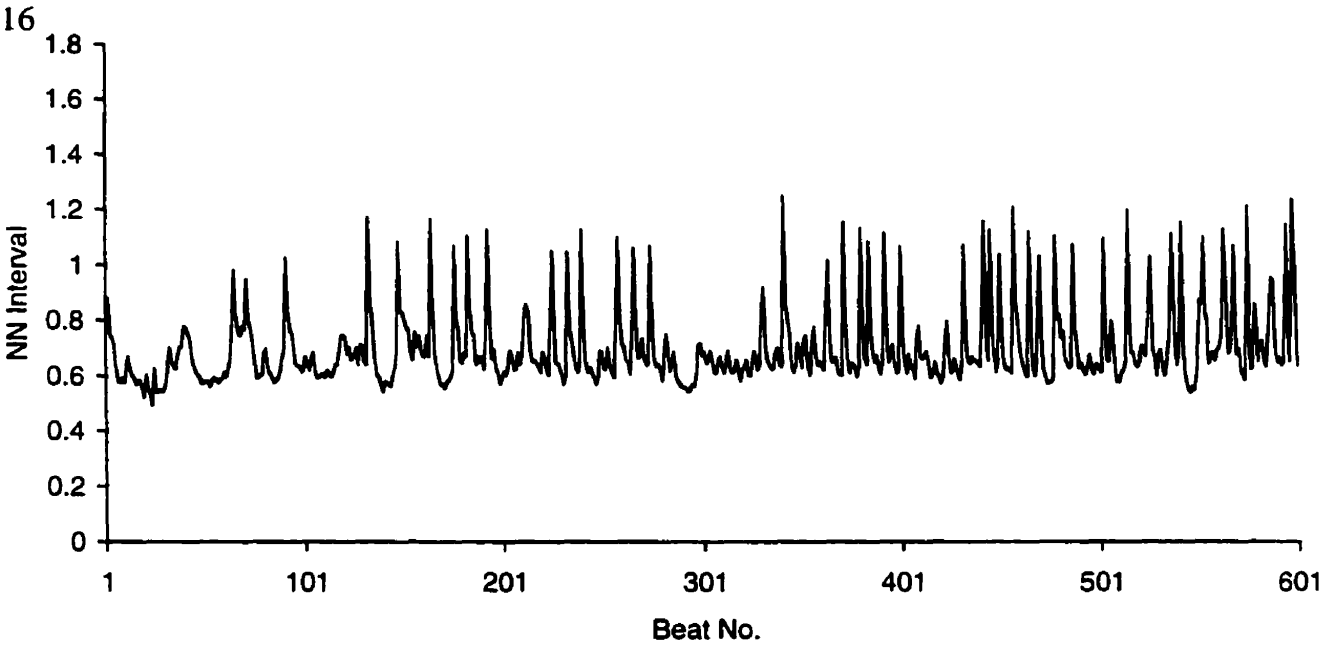
14


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

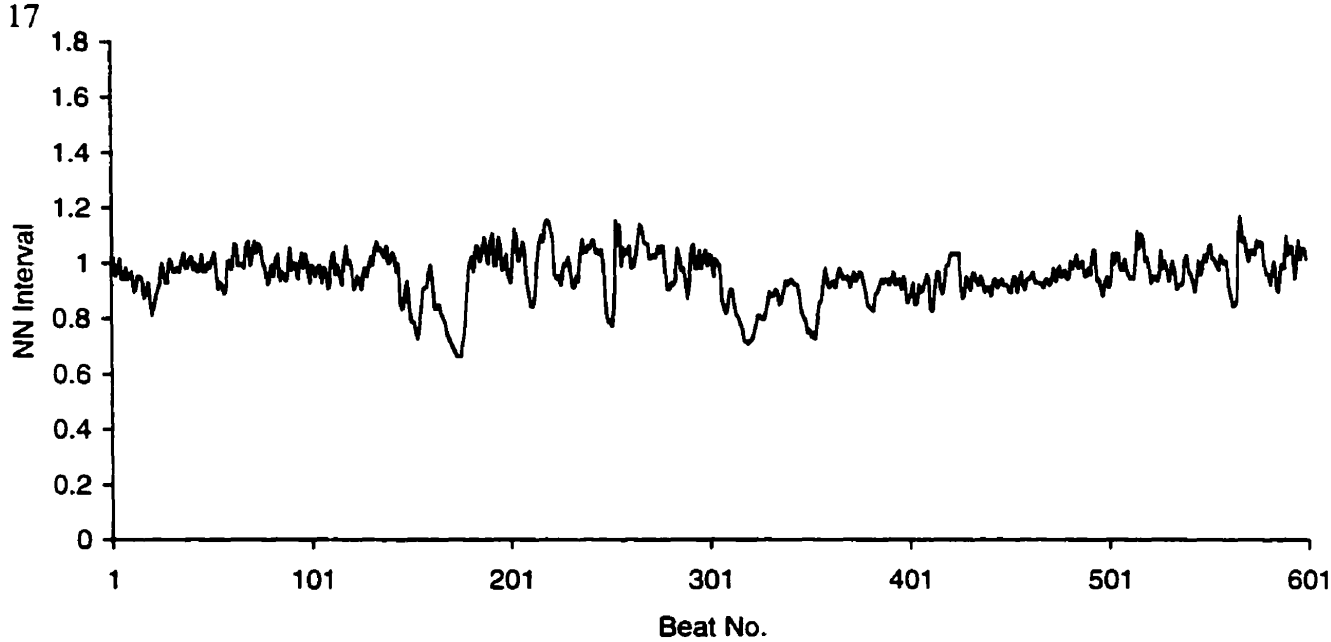
15


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

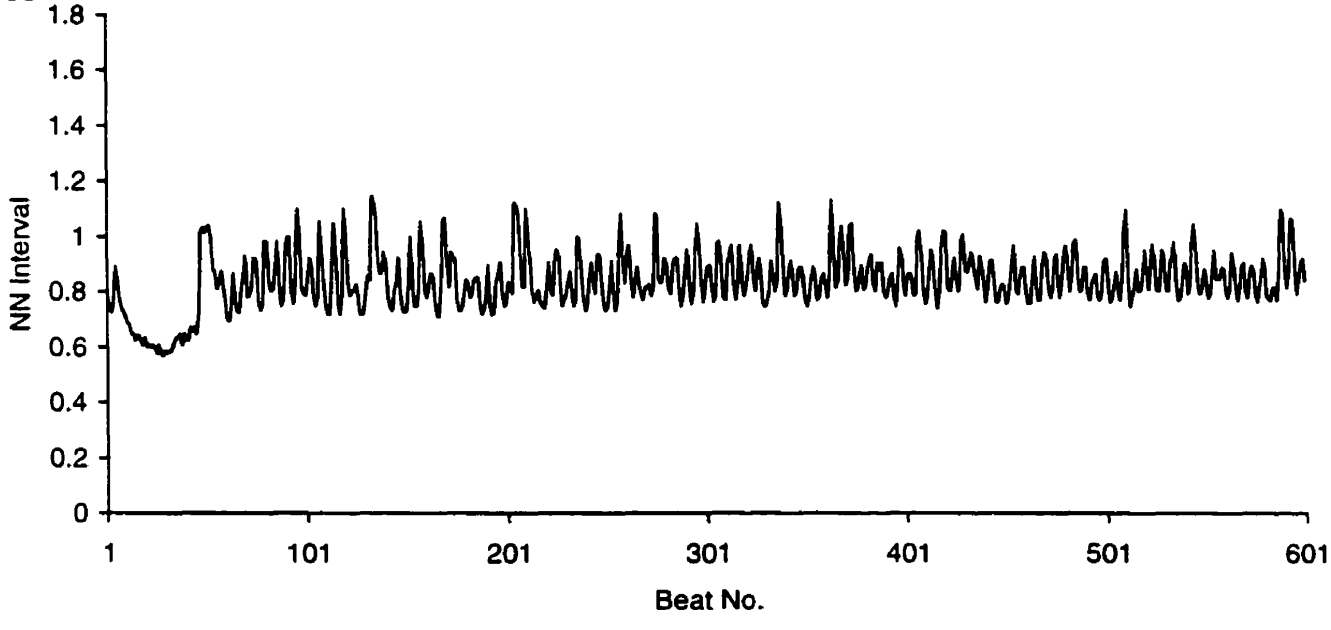
16


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

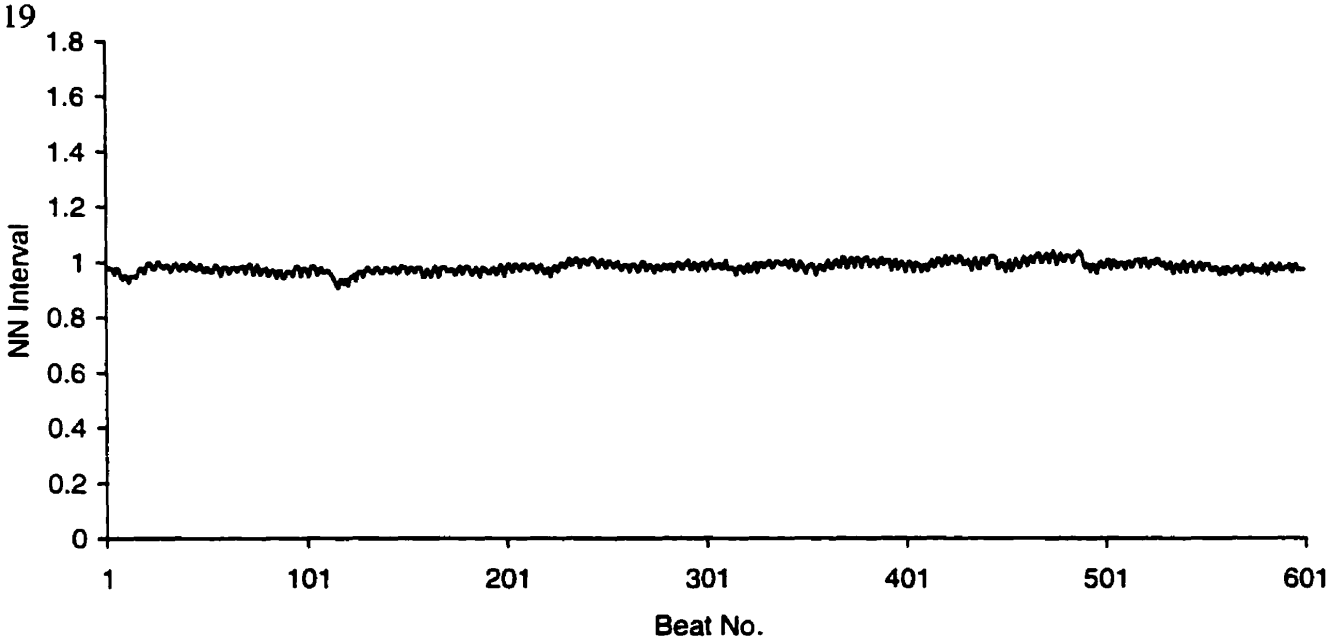
17

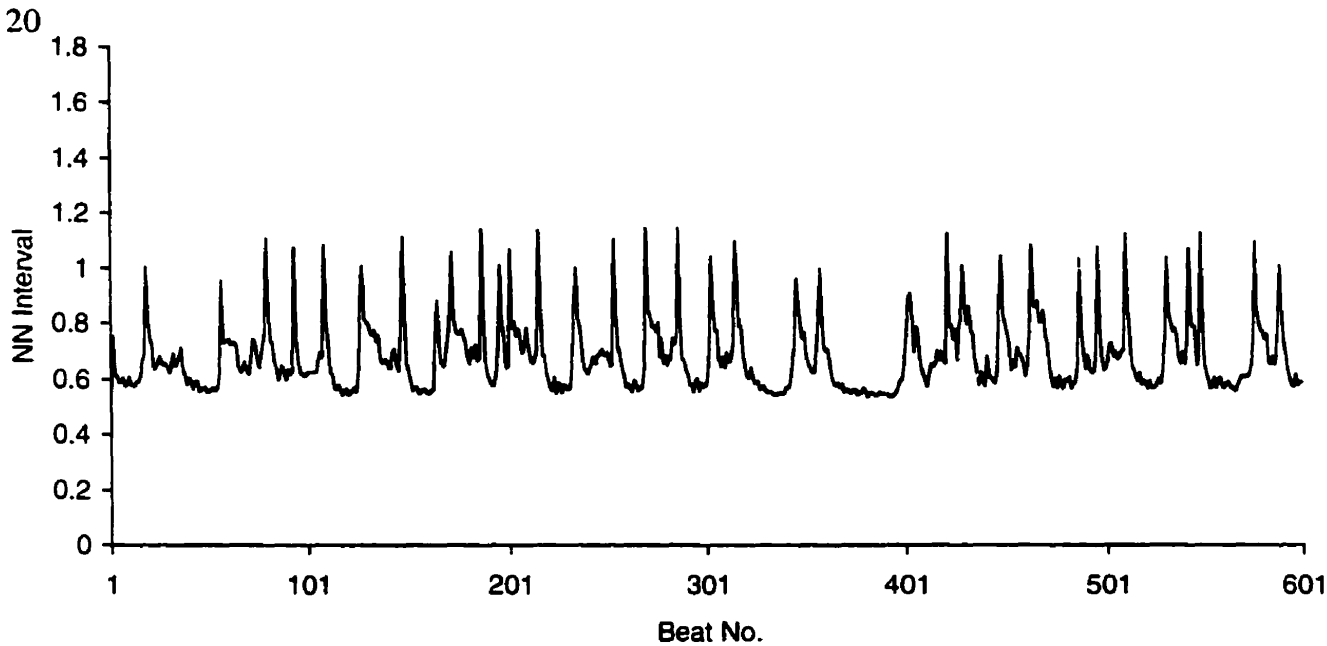

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

18

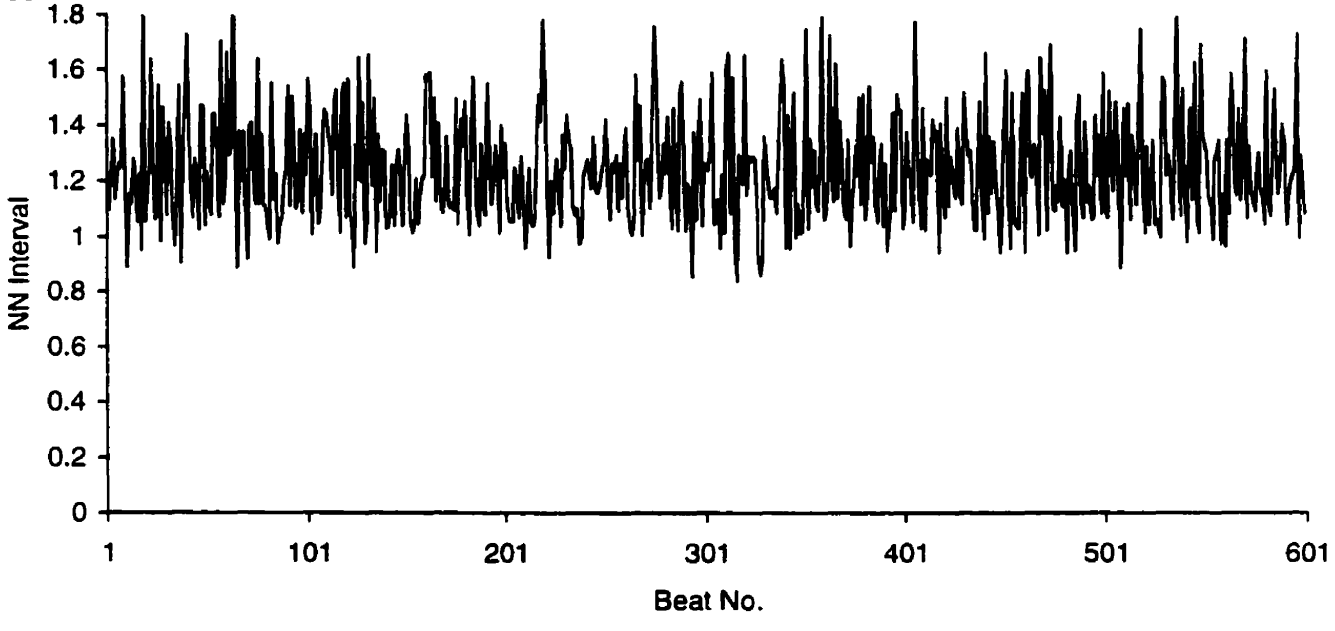

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

19

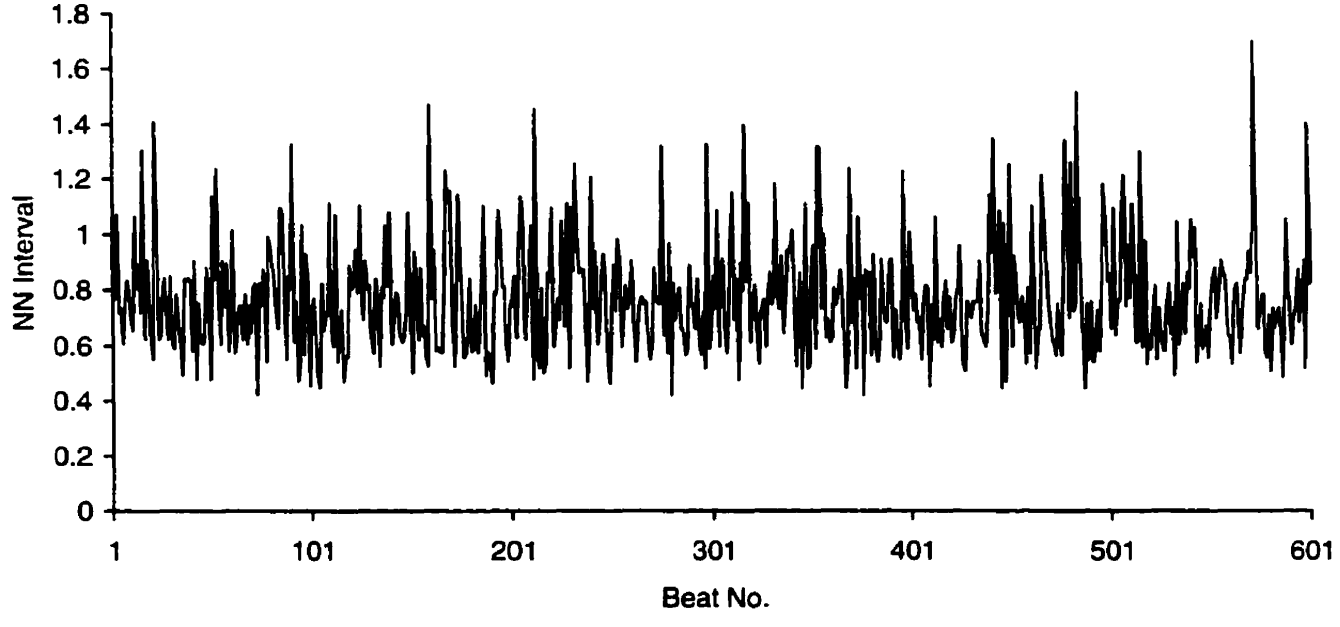

1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.


20
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

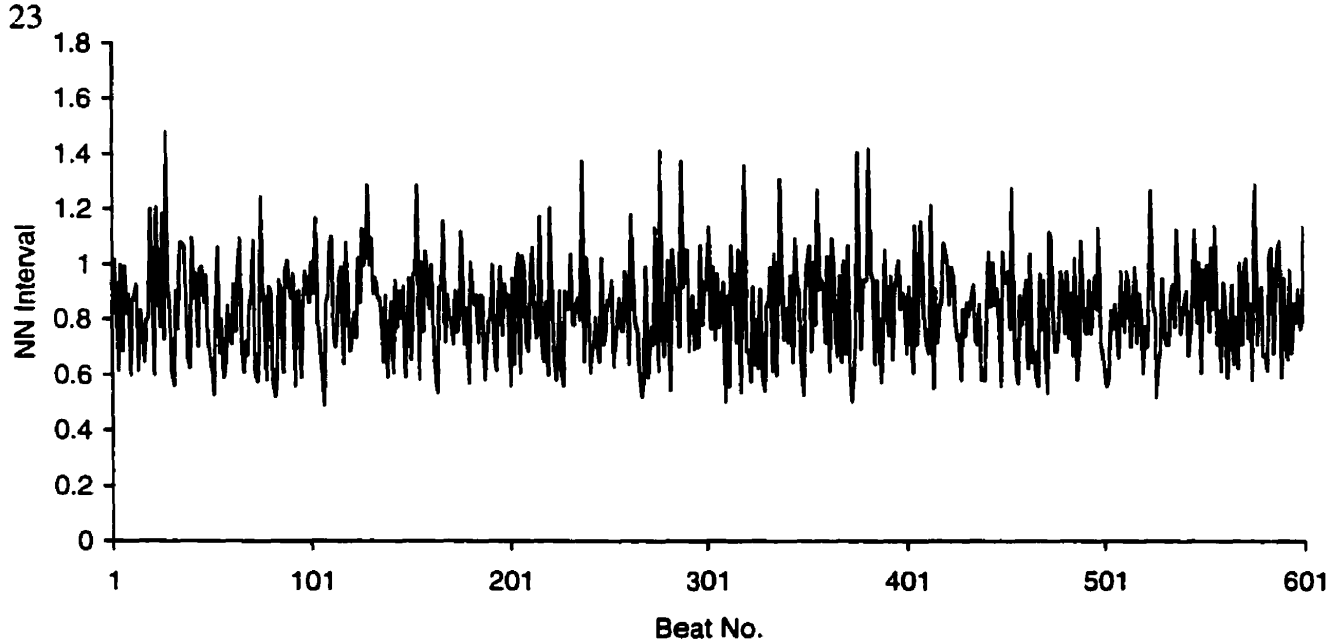
21

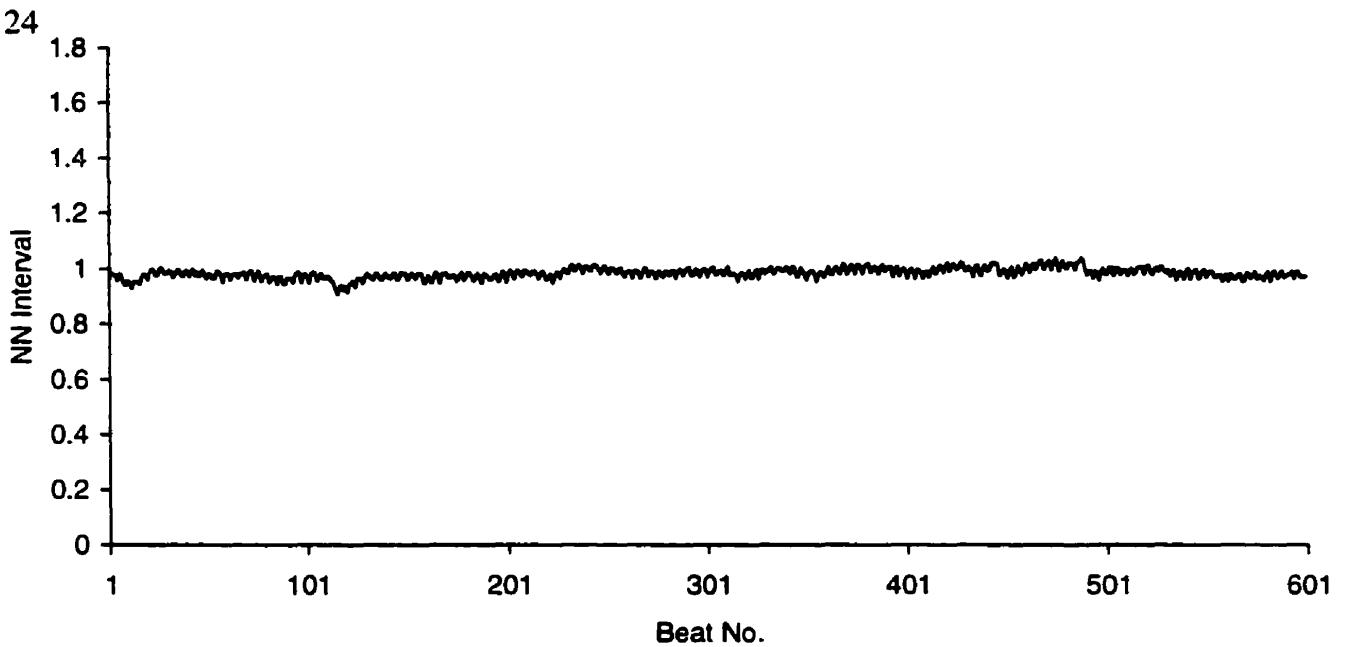

NN Interval
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
101
201
301
401
501
601
Beat No.

22


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

23


1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.


24
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
NN Interval
1
101
201
301
401
501
601
Beat No.

References

Anderson, John Davis. *The Influence of Scientific Concepts on the Music and Thought of Edgard Varèse*. D.A. Thesis, University of Northern Colorado, 1984.

Ballora, M., Pennycook, B., Glass, L. "Audification of Heart Rhythms in Csound." In *The Csound Book*, R. Boulanger, B. Vercoe, eds. Cambridge, MA: MIT Press, 2000.

Bate, John A. "The Effect of Modulator Phase on Timbres in FM Synthesis." *Computer Music Journal* 14(3) (Fall 1990): pp. 38-45.

Beauchamp, James. "Will the Real FM Equation Please Stand Up?" in Letters section of *Computer Music Journal* 16(4) (Winter 1992): pp. 6-7.

Bidlack, Rick. "Chaotic Systems as Simple (but Complex) Compositional Algorithms." *Computer Music Journal* 16(3) (Fall 1992): pp. 33-47.

Blauert, Jens. *Spatial Hearing: The Psychophysics of Human Sound Localization, revised edition.* Cambridge, MA: MIT Press, 1997.

Bly, Sara. "Multrivariate Data Mappings." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Bolognesi, Tommaso. "Automatic Composition: Experiments with Self-Similar Music." *Computer Music Journal* 7(1) (Spring 1983): pp. 25-36.

Bregman, A.S. *Auditory Scene Analysis*. Cambridge, MA: MIT Press, 1990.

Cannon, Walter. "Organization for Physiological Homeostatis." *Physiological Review* 9 (1929): pp. 399-431.

Chadabe, Joel. *Electronic Sound: The Past and Promise of Electronic Music.* Upper Saddle River, NJ: Prentice Hall, 1997.

Chou Wen-Chung. "Asian Concepts and Twentieth-Century Western Composers." *The Musical Quarterly*, 62(2) (April 1971): pp. 211-229.

Chowning, John. "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation." *Journal of the Audio Engineering Society* 21(7) (1974) (reprinted in *Computer Music Journal* 1(2) (April 1977): pp. 46-54).

Cosman, Madeleine Pelner. "Machaut's Medical Musical World." In *Machaut's World: Science and Art in the Fourteenth Century*, Madeleine Pelner Cosman, Bruce Chandler, eds. New York: New York Academy of Sciences, 1978.

Cutting, James E. "Auditory and linguistic processes in speech perception: Inferences from six fusions in dichotic listening." *Psychological Review* 83(2) (March 1976): pp. 114-40.

Dabby, Diana S. "Musical Variations from a Chaotic Mapping." *Chaos* 6(2) (Febuary 1996): pp. 95-106.

Dannenberg, Roger B. and Clifford W. Mercer. "Real-Time Software Synthesis on Superscalar Architectures." In A. Strange, ed. *Proceedings of the 1992 International Computer Music Conference*. San Francico: International Computer Music Association, 1992: pp. 174-177.

Davids, Zach. *Heartsongs: Musical Mappings of the Heartbeat*. Wellesley, MA: Ivory Moon Recordings, 1995.

de Campo, Alberto. *SuperCollider 2 Tutorial 0.8.5*. Bundled with SuperCollider software.

Dodge, Charles. "Profile: A Musical Fractal." *Computer Music Journal* 12(3) (Fall 1988): pp. 10-14.

Dodge, Charles and Thomas A. Jerse. *Computer Music: Synthesis, Composition and Performance*. New York: Schirmer Books, 1985; second edition 1995.

Fitch, Tecumseh and Gregory Kramer. "Sonifying the Body Electric: Superiority of an Auditory over a Visual Display in a Complex, Multivariate System." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Gardner, Martin. "Mathematical Games: White and brown music, fractal curves and one-over-f fluctuations." *Scientific American* 238(4) (April 1978): pp. 16-31.

Gillispie, Charles C. ed. *The Dictionary of Scientific Biography*, 16 vols. 2 supps. New York: Charles Scribner's Sons, 1970-1990. S.v. "Fibonacci, Leonardo" by Kurt Vogel.

Gleick, James. *Chaos: Making a New Science*. New York: Viking Penguin Inc., 1987.

Gogins, Michael. "Iterated Functions Systems Music." *Computer Music Journal* 15(1) (Spring 1991): pp. 40-48.

Goldberger, Ary L. and C.K. Peng, Zach Goldberger, Paul Trunfio. Liner notes to *Heartsongs: Musical Mappings of the Heartbeat*. Wellesley, MA: Ivory Moon Recordings, 1995.

Goldberger, Ary L. Memorandum to Malcolm W. Browne of the *New York Times* (private correspondence), October 16, 1995.

Goldberger Ary L. "Non-linear Dynamics for Clinicians: Chaos Theory, Fractals, and Complexity at the Bedside." *The Lancet* 347 (May 11, 1996): pp. 1312-1314.

Goldberger, Ary L. "Basic Concepts: Introduction to Chaos Theory, Fractals, and Complexity in Clinical Medicine." In *The Autonomic Nervous System*, Bolis C.L., Licinio J, eds. Geneva: World Health Organization, 1999.

Gordon, John W. and John M. Grey. "Perception of Spectral Modifications on Orchestral Instrument Tones." *Computer Music Journal* 2(1) (July 1978): pp. 24-31.

Graps, Amara. "An Introduction to Wavelets." *IEEE Computational Science and Engineering* 2(2) (Summer 1995): pp. 50-61

Griffiths, Paul. *Modern Music*. London: Thames and Hudson, 1986.

Grout, Donald J. and Claude V. Palisca. *A History of Western Music*. New York: W.W. Norton and Company, 1988.

_____. *The Grove Concise Dictionary of Music*. Stanley Sadie, ed. New York: W.W. Norton and Company, 1988.

Handel, Stephen. *Listening: An Introduction to the Perception of Auditory Events*. Cambridge, MA: MIT Press, 1989.

Haight, Frank A. *Handbook of the Poisson Distribution*. New York: John Wiley & Sons, Inc., 1967.

Harley, James. "Algorithms Adapted from Chaos Theory: Compositional Considerations." *Proceedings of the 1994 International Computer Music Conference* (ICMC): pp. 209-212. San Francisco: International Computer Music Association (ICMA), 1994.

Harley, James. *Analysis of "Cantico della Creature"*. D.M. Thesis in Composition, McGill University Faculty of Music, 1994.

_____. *The New Harvard Dictionary of Music*. Don Randel, ed. Cambridge, MA: The Belknap Press of Harvard University Press, 1986.

Hayward, Chris. "Listening to the Earth Sing." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Helmholtz, Hermann. *On the Sensation of Tone*. London: Longmans, 1885. (Original English translation by Alexander J. Ellis, reprinted by Dover, New York, 1954).

Holm, Frode. "Understanding FM Implementation: A Call for Common Standards." *Computer Music Journal* 16(1) (Spring 1992): pp. 34-42.

Ivanov, Plamen. *Scaling Features in Human Heartbeat Dynamics*. Ph.D dissertation, Boston University, Graduate School of Arts and Sciences, 1999.

Ivanov, Plamen and L.A.Nunes Amaral, Ary L. Goldberger, H. Eugene Stanley. "Stochastic Feedback and the Regulation of Biological Rhythms." *Europhysics Letters* 43(4) (August 15, 1998): pp. 363-68.

Ivanov, Plamen and Michael G. Rosenblum, C.-K. Peng, Joseph E. Mietus, Shlomo Havlin, H. Eugene Stanley, Ary L. Goldgerger. "Scaling Behaviour of Heartbeat Intervals Obtained by Wavelet-Based Time-Series Analysis. *Nature* 383 (September 26, 1996): pp. 323-327.

Ivanov, Plamen and Michael G. Rosenblum, C.-K. Peng, Joseph E. Mietus, Shlomo Havlin, H. Eugene Stanley, Ary L. Goldgerger. "Scaling and Universality in Heart Rate Variability Distributions. *Physica A* 249 (1988): pp. 587-593.

Kaplan, Daniel and Leon Glass. *Understanding Nonlinear Dynamics*. New York: Springer-Verlag Inc., 1995.

Karplus, Kevin and Alex Strong. "Digital Synthesis of Plucked Strong and Drum Timbres." *Computer Music Journal* 7(2) (Summer 1983): pp. 43-55.

Jaffe, David and Julius Smith. "Extensions of the Karplus-Strong Plucked Strong Algorithm." *Computer Music Journal* 7(2) (Summer 1983): pp. 56-69.

Jameson, David H. "Sonnet: Audio-Enhanced Monitoring and Debugging." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Kendall, Gary. "Composing from a Geometric Model: Five-Leaf Rose." *Computer Music Journal* 5(4) (Winter 1981): pp. 66-73.

Knapp, R. Benjamin and Hugh Lusted. "A Bioelectric Controller for Computer Music Applications." *Computer Music Journal* 14(1) (Spring 1990): pp. 42-47.

Kramer, Gregory. "An Introduction to Auditory Display." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Kramer, Gregory. "Some Organizing Principles for Representing Data with Sound." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Kramer, Gregory, et. al. *Sonification Report: Status of the Field and Research Agenda*. Prepared for the National Science Foundation by members of the International Community for Auditory Display Editorial Committee and Co-Authors. February, 1999.

Kramer, Jonathan. "The Fibonacci Series in Twentieth-Century Music." *Journal of Music Theory* 17(1) (Spring, 1973): pp. 110-149.

Kronland-Martinet, Richard. "The Wavelet Transform for Analysis, Synthesis, and Procesing of Speech and Music Sounds." *Computer Music Journal* 12(4) (Winter 1988): pp. 11-20.

Lambert, Kenneth A., and Martin Osborne. *Smalltalk in Brief: Introduction to Object-Oriented Software Development*. Boston: PWS Publishing Company, 1997.

Levarie, Siegmund and Ernst Levy. *Tone: A Study in Musical Acoustics*, 2d. ed. Kent State University Press, 1980.

Le Corbusier. *Modulor II*. London: Faber, 1958.

Lipsitz, Lewis A. and Ary L. Goldberger. "Loss of 'Complexity' and Aging: Potential Applications of Fractals and Chaos Theory to Senescence." *Journal of the American Medical Association* 267(13) (April 1, 1992): pp. 1806-9.

Lombreglia, Ralph. "Every Good Boy Deserves Favor." *The Atlantic Monthly* 272(6) (December 1993): pp. 90-100.

Madden, Charles. *Fractals in Music: Introductory Mathematics for Musical Analysis*. Salt Lake City, UT: High Art Press, 1999.

Malham, David G. and Andrew Myatt. "3-D Sound Spatialization Using Ambisonic Techniques." *Computer Music Journal* 19(4) (Spring 1995): pp. 58-70.

Mandelbrot, Benoit B. *The Fractal Geometry of Nature*. New York: W.H. Freeman and Company, 1983.

Mathews, Max V. *The Technology of Computer Music*. Cambridge, MA: MIT Press, 1969.

Matossian, Nouritza. *Xenakis*. New York: Taplinger Publishing Company, 1986.

McCabe, Kevin and Akil Rangwalla. "Auditory Display of Computational Fluid Dynamics Data." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

McMillan, Carolyn. "Sleep Apnea Has Police Worried. Sufferers Seeking Help." *Knight Ridder Newspapers*, August 9, 1999.

Mezrich, J.J. and S. Frysinger, R. Slivjanovski. "Dynamic Representation of Multivariate Time Series Data." *Journal of the American Statistical Association* 79(385) (March 1984): pp. 34-40.

Monro, Gordon. "Fractal Interpolation Waveforms." *Computer Music Journal* 19(1) (Spring 1995): pp. 88-98.

Moore, Brian C.J. *An Introduction to the Psychology of Hearing*, 3d ed. London: Harcourt Brace Jovanovich, 1989.

Moore, F. Richard. *Elements of Computer Music*. Englewood Cliffs, New Jersey: PTR Prentice Hall, 1990.

Moorer, James A. and John Grey. "Lexicon of Analyzed Tones (Part I; A Violin Tone)". *Computer Music Journal* 1(2) (April 1977): pp. 39-45.

Moorer, James A. and John Grey. "Lexicon of Analyzed Tones (Part II; Clarinet and Oboe Tones)". *Computer Music Journal* 1(3) (June 1977): pp. 12-29.

Moorer, James A. and John Grey. "Lexicon of Analyzed Tones (Part III; The Trumpet)". *Computer Music Journal* 2(2) (September 1978): pp. 23-31.

_____. "Facts about Sleep Apnea." NIH Publication No. 95-3798. U.S. Department of Health and Human Services, 1995.

Papoulis, Athanasios. *Signal Analysis*. New York: McGraw-Hill, 1977.

Peng, C.-K. and Joseph E. Mietus, Jeff M. Hausdorff, Shlomo Havlin, H. Eugene Stanley, Ary L. Goldberger. "Long-Range Anticorrelations and Non-Gaussian Behavior of the Heartbeat." *Physical Review Letters* 70(9) (March 1, 1993): pp. 1343-1346.

Peng, C.-K. and Shlomo Havlin, H. Eugene Stanley, Ary L. Goldberger. "Long-Range Anticorrelations and Non-Gaussian Behavior of the Heartbeat." *Chaos* 5(1) (1995): pp. 82-87.

Peng C.-K. and Jeff M. Hausdorff, Ary L. Goldberger. "Fractal Mechanisms in Neural Control: Human Heartbeat and Gait Dynamics in Health and Disease." In *Nonlinear Dynamics, Self-Organization, and Biomedicine*, J. Walleczek, editor. Cambridge University Press, 1999.

Pereverzev. S.V. and A. Loshak, S. Backhaus, J.C. Davis, R.E. Packard. "Quantum Oscillations Between Two Weakly Coupled Reservoirs of Superfluid 3He." *Nature* 388 (July 31, 1997): pp. 449-451.

Peterson, James L. *Computer Organization and Assembly Language Programming*. New York: Academic Press, 1978.

Pierce, John R. *The Science of Musical Sound*. New York: Scientific American Books, 1983.

Pilgram, Berndt and Daniel T. Kaplan. "A Comparison of Estimators for $1/f$ Noise." To appear in *Physica D*. 1997. Pre-print available at *http://www.math.macalester.edu/~kaplan/Preprints/Overfmethods/overfpaper.ps*

Pohlmann, Ken C. *Principles of Digital Audio*, third ed. New York: McGraw-Hill, Inc, 1995.

Pressing, Jeff. "Nonlinear Maps as Generators of Musical Design," *Computer Music Journal* 12(2) (Summer 1988): pp. 35-46.

Roach, Daniel. *Origin of Large-scale Temporal Structures in Heart Period Variability*. Cardiovascular Resarch Group, University of Calgary, 1996.

Roads, Curtis. *The Computer Music Tutorial*. Cambridge, MA: The MIT Press, 1996.

Rodet, Xavier. "Recent Developments in Computer Sound Analysis and Synthesis." *Computer Music Journal* 20(1) (Spring 1996): pp. 57-61.

Rosenboom, David. "The Performing Brain." *Computer Music Journal* 14(1) (Spring 1990): pp. 48-66.

Rossing, Thomas D. *The Science of Sound*, 2d ed. Reading, MA: Addison-Wesley Publishing Company, 1990.

Rowe, Robert. *Interactive Music Systems*. Cambridge, MA: MIT Press, 1996.

Scaletti, Carla. "Sound Synthesis Algorithms for Auditory Data Representations." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Standish, Thomas A. *Data Structures, Algorithms, and Software Principles.* Reading, MA: Addison-Wesley Publishing Company, 1994.

Steinberg, J.C. and W.B. Snow. "Auditory Perspective—Physical Factors." *Electrical Engineering* 53(1) (1934): pp. 12-15. 1934. Reprinted in *Stereophonic Techniques* by the Audio Engineering Society, 1986.

Strang, Gilbert. "Wavelets and Dilation Equations: A Brief Introduction." *SIAM Review* 31(4) (December 1989): pp. 614-627.

Task Force of the European Society of Cardiology and NASPE. "Heart Rate Variability, Standards of Measurement, Physiological Interpretation and Clinical Use." *Circulation* 93(5) (March 1, 1996): pp. 1043-1065.

Viswanathan, Gandhimohan M. and C.-K. Peng, H. Eugene Stanley, Ary L. Goldberger. "Deviations from Uniform Power Law Scaling in Nonstationary Time Series." *Physical Review E* 55(1) (January 1997): pp. 845-849.

von Baeyer, Hans Christian. "Wave of the Future." *Discover* (May 1995).

Voss, Richard F. and John Clarke. "'1/*f* noise' in music and speech." *Nature* 258 (November 27, 1975): pp. 317-318.

Voss, Richard F., Clarke, John. "'1/*f* noise' in music: Music from 1/*f* noise." *Journal of the Acoustic Society of America* 63(1) (January 1978) pp. 258-263.

Webster, J.H. Douglas. "Golden Mean Form in Music." *Music and Letters* 31 (July 1950): pp. 238-249.

Wenzel, Elizabeth M. "Spatial Sound and Sonification." In *Auditory Display: Sonification, Audification, and Auditory Interfaces*, edited by G. Kramer. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVIII. Reading, MA: Addison Wesley, 1994.

Wessel, David. "Timbre Space as a Musical Control Structure." *Computer Music Journal*, 3(2) (June 1979): pp. 45-52.

Wilkinson, Scott R. *Tuning In: Microtonality in Electronic Music*. Milwaukee: Hal Leonard Books, 1988.

Wilson, Paul R. "Uniprocessor Garbage Collection Techniques." *SpringVerlag Lecture Notes in Computer Science: 1992 International Workshop on Memory Management.*

Wilson, Paul R. and Mark S. Johnstone. "Real-Time Non-Copying Garbage Collection." *Position paper for the 1993 ACM OOPSLA Workshop on Memory Management and Garbage Collection.*

Xenakis, Iannis. "The Crisis of Serial Music." *Gravesaner Blätter*(1) (July 1956): pp. 2-4.

Xenakis, Iannis. *Formalized Music.* Indiana University Press, Bloomington Indiana. 1971. (Originally published as *Musique formelles*, by the Paris La Revue Musicale, 1963).

Xenakis, Iannis. *Metastaseis,* full score. London: Boosey & Hawkes, 1953-54.

Young, S.J. *Real Time Languages: Design and Development.* Chichester: Ellis Horwood Limited, 1982.

NOTE TO USERS

The CD is not included in this original manuscript. It is available for consultation at the author's graduate school library.

This reproduction is the best copy available.