Musical Sound Information

Musical Gestures and Embedding Synthesis

by

Eric Métois

Diplôme d'Ingénieur
Ecole Nationale Supérieure des Télécommunications de Paris - France
June 1991

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

at the
Massachusetts Institute of Technology
February 1997

Signature of author

Program in Media Arts and Sciences
November 30, 1996

Certified by

Tod Machover
Associate Professor of Music and Media
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by

Stephen A. Benton
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Musical Sound Information

Musical Gestures and Embedding Synthesis

Eric Métois

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning on November 30, 1996
in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Abstract

Computer music is not the artistic expression of an exclusive set of composers that it used to be. Musicians and composers have grown to expect much more from electronics and computers than their ability to create "out-of-this-world" sounds for a tape piece. Silicon has already made its way on stage, in real-time musical environments, and computer music has evolved from being an abstract layer of sound to a substitute for real instruments and musicians. Within the past three decades, an eclectic set of tools for sound analysis and synthesis has been developed without ever leading to a general scheme which would highlight the issues, the difficulties and the justifications associated with a specific approach. A rush in the direction of incremental improvement of existing techniques has traditionally distinguished analysis and synthesis.

Rather than confining ourselves to one of these arbitrarily exclusive tasks, we are pursuing an ambitious dream from a radically new perspective. The ultimate goal of this research is to infer virtual instruments from the observation of a real instrument without any strong pre-conception about the model's architecture. Ideally, the original observation should be a simple audio recording. We also want our inferred virtual instruments to exhibit physically realistic behaviors. Finally, we want the nature of the virtual instrument's control to be universal and perceptually meaningful.

For this purpose, our investigation falls naturally into three steps. We first identify a set of perceptually meaningful *musical gestures* which can be extracted from an audio stream. In the case of a monophonic sound, we discuss the definition and the estimation of loudness, pitch contour, noisiness and brightness. The second step is to investigate means by which a physically meaningful model can be inferred from observed data. While doing so, we introduce *embedding modeling* as our general philosophy and reduce modeling to the characterization of prediction surfaces. We also suggest some general purpose interpretations, including an original *cluster-weighted modeling* technique. Finally, our third and last step is to suggest a strategy for applying such modeling ideas to musical audio streams parametrized by the perceptually meaningful *musical gestures* that we previously identified. We present pitch synchronous embedding synthesis (or *Psymbesis*), a novel approach to the inference of a virtual instrument, as a working sound synthesis algorithm and an interpretation of these suggestions.

Psymbesis was designed specifically around musical instrument modeling but the general philosophy of *embedding modeling* extends beyond the field of computer music. For instance, we establish *embedding modeling* as a useful tool for the analysis of fairly small but highly non-linear deterministic dynamical systems of arbitrary nature. We expect that the introduction of *embedding modeling* will provide signal modeling with a new perspective, relaxing the constraint of linearity and filling the present gap between physical models and standard signal processing.

Thesis Supervisor: Tod Machover - Associate Professor of Music and Media

Doctoral Dissertation Committee

Thesis Supervisor

Tod Machover
Associate Professor of Music and Media
Program in Media Arts and Sciences

Thesis Reader

Neil Gershenfeld
Associate Professor of Media Arts and Sciences
Program in Media Arts and Sciences

Thesis Reader

Rosalind Picard
Associate Professor of Media Technology
Program in Media Arts and Sciences

Acknowledgments

None of this work would have been possible without the stimulating environment of the Media Laboratory. I'd like to take this opportunity to thank my advisor, Professor Tod Machover, for inviting me to become a part of this environment back in September 1992 and for supporting my work ever since. In addition to a unique opportunity of working in a creative and artistic environment, Tod provided me with a degree of trust and confidence which has unleashed my own creativity for my research.

Throughout the past four years, I've worked closely with Professor Neil Gershenfeld to whom I owe my early conversion to the philosophy of Embedding Modeling. His insights and encouragement have been an abundant source of inspiration to me in the past four years. I would also like to thank Professor Rosalind Picard for supporting my work as an insightful and detail-oriented reader. Sharing some of her ideas about modeling resulted into some major breakthroughs in the investigation of Embedding Modeling.

I would probably think computer music refers to some hip techno genre if it hadn't been for Professor David Wessel who has been a mentor figure to me for the past six years. Thanks for opening my eyes and ears during the most fun internship one could ever dream of at Berkeley's Center for New Music and Audio Technology.

My thanks are warmly extended to the remarkable staff and graduate student community of the Media Laboratory. Throughout four rocky years towards graduation, I was blessed by three wonderful offices-mates, Mike Wu, David Waxman, and John Underkoffler, who not only became good friends in spite of my undeniable "frenchyness", but also turned out to be among my best teachers. Rather than going through an exhaustive list of the lab's graduate students of the past four years, I'll just scream a loud thanks from the top of my lungs. I'm grateful for all of these insightful lunch breaks, trips to the trucks and to the coffee machine. I can only hope you all got as much out of them as I did.

To my friends from the "good old days", Philippe, Vincent, Anne, Hervé, Suzanne, Raphaël, Benoît, Rémi, Nanou: Thank you for not holding my inexcusable silences against me. I know that materialistic details such as time and distance can't suffice to keep us apart.

I'd like to thank my parents for trusting my judgment and supporting my decisions throughout the years, even when it meant an ocean between us. Finally, I would have probably never set foot anywhere close to MIT if it hadn't been for my best friend, soul-mate and wife, Karyn. For your patience, understanding and love, I'm indebted to you for life.

Support for this work was provided in part by Yamaha, Sega, Motorola and TTT.

Contents

Chapter 1

Introduction

After situating musical sound analysis and representation in its historical context, we will highlight the central issues associated with these problems and present the scope of this work. A major step will be to articulate the motivations for the processing of musical sound and show how these motivations should influence the choice of a model for the representation for musical sounds. This chapter will end with a statement of the author's motivations and an overview of this document.

Quick History

Experimenting with the sound produced by a vibrating string a little over twenty-six centuries ago, Pythagorus applied his newly developed theory of proportions to "pleasing musical intervals," leading to his own tuning system. Twelve centuries later, the Roman philosopher Boethius reinforced the relationships between science and music, suggesting notably that pitch is related to frequency. By the middle ages, music (referring to harmony) was perceived as one of the four noble fields of mathematics, along with arithmetic, geometry and astronomy. The notion of frequency itself had to wait until the early seventeenth century before it was scientifically defined by Galileo Galilei, an Italian scientist whose work, believed to be the foundation of the modern study of waves and acoustics, clearly reflected his interest in music.

This first demystification of pitch left the notion of tone color (or timbre) in its original obscurity until Helmoltz, at the end of the nineteenth century, suggested a characterization of timbre based on the set of sinusoidal components of the quasi-periodic part of a sound. Clearly influenced by the highly controversial series introduced by Fourier in 1822, this study gave birth to the *classical conception* of timbre. At the time of this suggestion, Helmoltz was already aware of the weaknesses of this approach and the importance of temporal information became obvious when technology provided recording and editing tools for sounds. Adding a

temporal dimension to this classical conception led to sonograms, an extensively used representation for sound ever since.

After the early 1950s, the availability of general purpose computers and the refinement of electronics relieved composers from the rigid boundaries of acoustically produced sounds and sent them on a quest for timbre synthesis, representation and manipulation. Soon additive, subtractive, formantic, wave-shaping and FM synthesis were born, each one offering a different representation for timbre and a different set of parameters for controlling it. Yet the growth of tone color's role in composition, from "musique concrète" to computer music and today's elaborately produced popular music, raised more questions than it resolved mysteries. Musicologists, psychoacousticians and computer musicians are left with the same frustration: in spite of timbre's omnipresence in composition throughout these numerous centuries of science and music history, there is still no satisfactory universal language, structure or characterization for musical sounds past their loudness and their pitch.

Musical Sound Representation: The Issues

Perhaps this lack of structural understanding of the exact role of sound color in music comes from the wide variety of phenomena involved in any musical process. If music starts where words stop, it should be understood as a medium of ideas and emotions. The following figure is an attempt to illustrate this point by representing a musical process as a chain of communication. Very much like for language, this chain spans both the cognitive and the physical worlds, requiring both of them in order to make any sense.

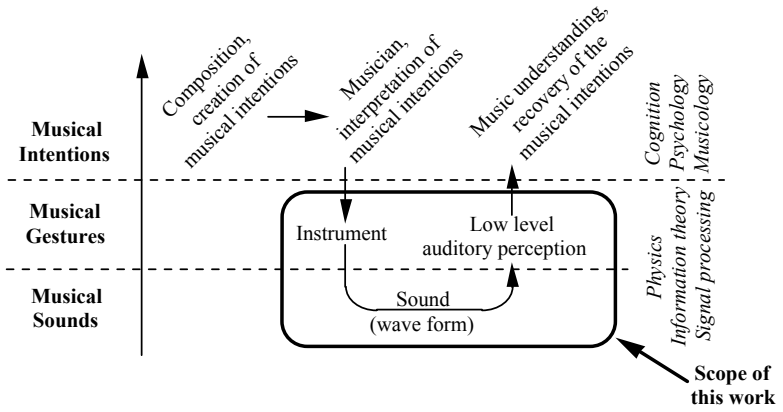


Fig. 1.1 - Music as a chain of communication.

In this diagram, *Sound* refers literally to the wave form produced by the instrument. By *Low level auditory perception* we refer to the set of features provided by the first stages of our auditory system (external and internal ear). This is not to be taken literally in its physiological sense; we refer hereby to some fairly straightforward signal processing artifacts (such as frequency analysis) which might be related to those taking place in our cochlea. This explains why "Low level auditory perception" was excluded from the "cognitive field" in the preceding figure.

This figure is also an opportunity to state clearly what the scope of this work is. It will not venture into the high spheres of psychology, cognitive science and musicology.

Musical Intentions versus Musical Gestures

Although the preceding diagram might look somewhat trivial, it is not rare to come across attempts to recover musical intentions from sounds via signal processing only, underestimating the role of human perception. This confusion illustrates the obscure boundary between *Musical Intentions* and *Musical Gestures*. Neither of these notions have the pretension to be universal. They reflect the author's convictions concerning the boundary between the roles of information theory and of psychology in this chain of communication.

Musical Intentions

Musical intentions are objects that require some knowledge or expectations about what music is supposed to be. If we decide to keep language as an analogy, these objects have a similar nature to the one of words, sentences and meaning. They can often be seen as the results of some decision making given the prior knowledge of a context. The lowest-level musical intention is probably a note played in a specific fashion on a specific instrument.

In his discussion about words and ideas, Marvin Minsky [Min85] suggests that a word could be a *polyneme* which, once activated, would act as a switch for the multiple *agencies* it's connected to. This web of connections would be the result of a learning process that would be specific to the individual. In many ways, music and its ability to communicate ideas and emotions could be thought to fit a similar scheme. Once recognized, a particular *musical intention* would activate its associated polyneme and evoke ideas, or more likely emotional states in the case of music, through the subsequent activation of several agencies. The number and the nature of these *musical polyneme/agencies* connections would reflect the individual's personal musical experience. This would explain why music can sometimes sound desperately meaningless when it crosses cultural boundaries. Some might think that all rap music sounds the same while some others could probably argue that this month's MTV top 20 offers more musical diversity than all the music ever written before the 20th century. The diversity of musical understanding is undoubtedly much larger than for words as the associated learning process is not supervised as explicitly as for language.

The nature of these musical intentions and the mechanism of their eventual connections with other agencies of the human mind will not be addressed in the context of this thesis. These issues will be religiously left to psychologists, musicologists and ethno-musicologists' expertise. We will hereby simply acknowledge the existence of such musical intentions and their ability to communicate ideas and emotions. Our ability to recover these intentions from audio streams implies that their nature is subject to the artifacts of our auditory perception. This work will attempt to identify a set of low-level measurements that are likely to reflect some of these artifacts in the hope that this set will lead to some perceptually meaningful musical gestures that can communicate musical intentions.

Musical Gestures

There is a diversified set of objects spanning the gap between the lowest-level musical intention (cognition, psychology, musicology) and a simple wave form (physics). These objects will be referred to as *musical gestures* and they should be seen as the features based on which musical intentions will eventually be recovered through some decision making. Here, the terms "decision making" should be taken fairly loosely as the author not intending to trivialize the mechanism of the human mind. Back to our analogy with language, these objects have a similar nature to the one of formants, phonemes and intonation.

Implied by the preceding diagram is also the claim that although the information fed to an instrument and to a listener's brain are of different nature, they are of similar levels (see the following figure). The gestures that are fed to the instrument are of physical nature (fingering, pressure, energy, etc.) whereas the gestures resulting from our auditory perception are not. However, both present the ability to communicate musical intentions at a higher level than an audio wave form. The similarity of their level of abstraction motivated the author to label them both as *Musical Gestures*. This assimilation will become crucial when we face the question of how to control a virtual instrument (sound production).

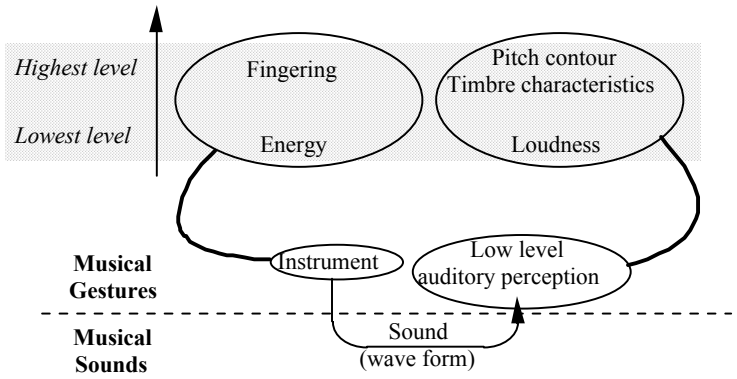
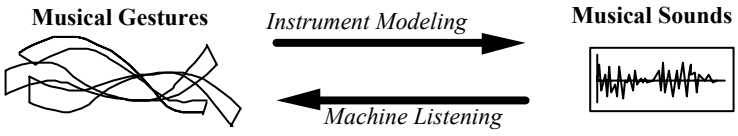


Fig. 1.2 - Musical gestures

Machine Listening versus Instrument Modeling

As expressed in what precedes, the scope of this work is centered around the boundary between *Musical Gestures* and *Musical Sounds*. Crossing this boundary in one direction or the other defines the two major tasks of computers for music applications: *Machine Listening* and *Instrument Modeling*.



Machine Listening

This task is crucial in the context of the development of an interactive musical system that would be intended to follow, respond to or jam with a musician. Without pretending to understand the essence of music, such a system should be able to parse sound at a similar level than our lowest level of perception. It should capture the set of musical gestures that it needs in order to make any coherent musical decision. Its analysis of the incoming sound will lead to a set of features which can be seen as a representation (or model) for sound.

It seems clear that given such a task of estimating perceptual features, an appropriate model for sound should try to reflect the artifacts of our own perception. As we will identify some musical gestures in Chapter 3, we will suggest a collection of approaches and algorithms that have been implemented and used for several projects at the Media Laboratory's Hyperinstrument group.

Instrument Modeling

This task is undoubtedly the oldest use of computers and technology for music. Ever since Max Mathew's first generation of the MUSIC program, people have turned bits into sounds, freeing music from the constrained world of acoustic instruments. Recording, transforming, stretching, cutting, pasting and sound wave editing raised a large enthusiasm due to their novelty. The boundless world of sound synthesis could appear as the sonic playground that musicians and composers had been dreaming of for a long time. Yet, the lack of navigational tools or language for sound quality prohibited any exploration of that world that could go beyond the empirical. The diversity of sound synthesis techniques allowed a set of parametric descriptions for subsets of this sonic world. Each synthesis algorithm can be seen as a navigational tool that will span a specific subset of timbre space by offering a set of controls which could be interpreted as a language, defining a model for sound.

Its behavior as a dynamical system will determine the model's *physical meaningfulness*. The nature of its control set (i.e. its language and its relationship with music) will determine the model's *musical meaningfulness*. In the ideal case, the control set of a virtual instrument should be a comprehensible set of musical gestures and its resulting behavior should reflect the expectations one has about the behavior of a physical system.

The diversity of the timbre subspace that it spans will measure the model's *universality*. The control set of an algorithm that will systematically produce similar sounds won't deserve the status of a language for timbre. An ideal universal model would span the entire space of sounds that can reasonably be qualified as musical.

The last issue associated to the choice of a model is its *ease of inference* from a prototypical sound or system. Indeed, it is becoming rare to design a virtual instrument from scratch as the demand for "out-of-this-world" instruments has been replaced by a concern for realism.

Motivations and Overview

Music and Media - Motivations

The universal frustration raised by the constraints of frozen HTML documents and the flourishing of creative CGI scripts and Java applets on the Internet are only a small portion of the signs indicating that Multimedia requires interaction. If we want Multimedia to carry more than a fashion statement, it will have to offer content, features and experiences that couldn't be delivered without it. Interactivity may very well be what the point of Multimedia boils down to. At the same time, while entertainment cannot be considered a necessity to our survival, the overwhelming size of its industry is the clear sign of an addiction to it. In light of what happened with television, it leaves no doubt in the author's mind that the informative value of Multimedia will turn out to be an artifact of its entertainment power.

Music evolved both as an art form and as a source of entertainment to match the constantly evolving technology of its time. From acoustic performance to recording, broadcasting and editing, music has always tried to use the latest media that were made available. Multimedia technologies are only another trend that music has to follow. Already, artists from the music industry were among the first ones to produce interactive CD-ROMs, and private CD collections were among the first few clichés of what people posted on the Internet. In order for music to become a coherent part of Multimedia, it needs to be delivered in a format that allows interactivity. MIDI and General MIDI have already pushed music in that direction and they already justify any effort put towards instrument modeling and machine listening.

If silicon and bits are called to substitute acoustic instruments in the context of particular musical activities, it is crucial to capture the essence of the original instrument's behavior. The quality of a virtual instrument should not only be a function of its ability to reproduce a particular wave form, but rather a function of the realism of its behavior as a musical instrument. For this purpose, it is very important to identify the wide variety of concerns and issues associated with the modeling of a physical system. Because of its novelty, *Embedding Modeling* is a perfect opportunity to build a general framework from the ground up and identify these issues and concerns without the bias of previous traditional approaches (such as linear system theory for instance).

Once the original instrument's behavior is dissociated from its physical body, it is dissociated from its interface as well, and the question remains as to how to play this virtual instrument. In an ideal case, one would want the virtual instrument to respond to some meaningful musical gestures. Because of the specificity of an acoustic instrument's interface, it takes years for a musician to learn how to control the musical gestures associated with that particular instrument. The nature of a virtual instrument's interface can be arbitrary and therefore, one could very well imagine a single interface that could control a variety of virtual instruments. To some extent, MIDI could have become this universal musical gesture (or interface) if it had not been so influenced by the omnipresence of keyboards in computer music.

Overview

We recall that the ultimate goal of this research is to infer virtual instruments from observed audio streams, without any major pre-conception concerning the model's architecture. In addition to exhibiting physically meaningful (or realistic) behaviors as a dynamical system, we want these virtual instruments to be controllable by a set of perceptually *meaningful musical* gestures. Given this ambitious goal, both *machine listening* and *Instrument modeling* are relevant to this work. The three major steps of this investigation are 1) the identification of appropriate *musical gestures*; 2) the investigation of the inference of non-linear model in the wide context of dynamical systems; 3) the suggestion of an approach to the inference of virtual instruments that are controlled by our *musical gestures*.

After a review of previous related work and notions, we dedicate Chapter 3 to the identification of perceptually meaningful *musical gestures*. While doing so, we highlight issues that are inherent to *machine listening* as well as real-time concerns. We also suggest means by which such *musical gestures* can be extracted in real-time from monophonic audio streams. The resulting tools for the analysis of musical audio streams have been used in the context of various projects at the Media Laboratory and although these don't necessarily refer to *instrument modeling*, we provide pointers to these applications as well.

The core of this work discusses the inference of physically meaningful non-linear models from observations. Chapter 4 raises the issues and concerns that are associated to such a task. It sets the foundation of *embedding modeling* by applying Floris Takens' *embedding theorem* [Tak81] to the observation of time series. It is an opportunity to revisit some well-established notions in modeling and information theory, and to draw interesting concepts which lead to the philosophy underpinning *embedding modeling*. The concept of sampling the physics of a system rather than sampling the wave-form of an observation is the basis of our faith in *embedding modeling*'s ability to lead to physically meaningful models. The two following chapters (Chapter 5 and Chapter 6) suggest general purpose interpretations of this modeling scheme. They also illustrate two extreme approaches by estimating respectively global and local models in the observation's state space. Cluster-based modeling is the central issue of Chapter 6, and we present a very promising, innovative and versatile scheme as *cluster-weighted modeling*.

Chapter 7 focuses back on the modeling of virtual instruments. We use the insights of the previous general approaches in order to suggest *pitch synchronous embedding synthesis* (or *Psymbesis*) as a means by which we can achieve our original goal. *Psymbesis* is an original scheme which can be seen as a set of constraints and hypotheses concerning the nature of a musical instrument. Rather than being imposed arbitrarily by the specifics of a modeling approach, these constraints are derived from typical observations of the data's nature. *Psymbesis* is presented in a way that may lead to various interpretations but it is also reduced to the simplest possible implementation in a concern for applicability and validation of the approach.

Chapter 2

Background

This chapter will survey some previous work that contributed to today's understanding of timbre and its relationship with musical expression. Timbre, as the ultimate characterization of the perceptual quality of a sound, is central to any investigation of synthesis and instrument modeling. Discussing timbre in the context of musical aesthetics will situate this work within computer music at large. We will then review quickly some of linear system theory's main notions and assumptions which are relevant to modeling. The definitions of entropy, information and redundancy will then be recalled as we will refer to them subsequently. Finally we will say a few words about dynamical systems before introducing Floris Taken's embedding theorem.

Timbre and Musical Expression

In 1911, Arnold Schoenberg wrote: *"The evaluation of tone color, the second dimension of tone, is in a much less cultivated, much less organized state than is the aesthetic evaluation of pitch. Nevertheless, we go right on boldly connecting sounds with one another, contrasting them with one another, simply by feeling; and it has never yet occurred to anyone to require of a theory that it should determine laws by which one may do that sort of thing. Now, if it is possible to create patterns out of pitches, patterns we call "melodies," progressions, whose coherence evokes and effect analogous to thought processes, then it must also be possible to make progressions out of "tone color," progressions whose relations with one another work with a kind of logic entirely equivalent to that logic which satisfies us in the melody of pitches."*

In this quote, Schoenberg points out the lack of structure for the notion of "tone color" as opposed to the wide vocabulary and set of rules that surrounds pitch. He foresees the possibility of using color as the predominant element of a musical piece and of creating form

exclusively from a progression of tone color. Underlying this thought is the assumption that there should be a way to express or predict relationships between tone colors the same way classical solfège does it for pitch.

Timbre-based composition

The use of timbre and its ability to create form has been a source of experiments ever since instrumental music and vocal music were dissociated (between 1400 and 1600). The refinement of instrumental music in the 17th century stated clearly that there was more to music than simply harmony (as the "science of horizontal and vertical pitch arrangement"). Yet, the rigidity of acoustic musical instruments didn't allow composers to go very far in these experiments. As recording technology made its appearance in the 20th century, one could no longer deny the ability for a sound alone (with no visual cues) to carry emotions, ideas and even convey images. This observation evoked the awareness of a sonic world populated by what are usually referred to as *sound objects*, a notion first introduced by J.P. Schaeffer in the 60s.

It can be argued however that composers didn't need to wait for modern technology in order to use timbre as a flexible component from which they could express their art. Varèse conceived of his music as *"bodies of sound in space"* before he even gained access to any electronic equipment. Still, the access to recording and editing techniques might have boosted composers' curiosity and helped Schoenberg's dream of timbre-based composition to become reality. In an initial euphoria, composers recorded as much of our sonic world as they could and *Musique concrète* was born. This almost "maniacal" recording and pasting of natural sounds provided a huge variety of material but *"one can only transform these sounds in ways that are rudimentary in comparison to their richness"* (J.C.Risset) [Ris88]. A little later, when computers appeared as musical tools of an apparently limitless flexibility, timbre structure could no longer be ignored and it became obvious to the musical, psychological and scientific populations that Schoenberg's concern had to be taken seriously.

Some examples

By the 1950s and with electronics, the rigidity of acoustical instruments' timbre being bypassed, it seemed that composers and musicologists were finally armed to make Schoenberg's dreams come true. Composers and scientists began an enthusiastic quest for timbre representations and manipulations. Soon additive, subtractive, and formantic synthesis were born, each one offering a different representation for timbre and a different set of knobs for controlling it. Then *frequency modulation* (John Chowning [Cho73]) and *waveshaping* (Daniel Arfib [Arf79]) appeared as simplifications or short-cuts and "tone color progressions" were no longer the dreams of an exclusive set of specialists. In 1968, with the collaboration of Max Mathews (the father of the MUSIC program's first generation), Risset composed *"Computer Suite from Little Boy"* where he clearly demonstrated his ability to use timbre progression as a source of musical form. Aware of some peculiar psychological aspects of sound perception, this piece is a perceptual puzzle where the composer clearly plays with the listener's mind. Risset's representation of timbre is exclusively based on additive synthesis which added a temporal dimension to the classical conception stated by Helmoltz. The flexibility and the coherence of such a representation can still be appreciated today and it has traditionally been the preferred representation of sound quality. This piece was also an opportunity for the composer to introduce inharmonic structuring of timbre as well as the

musical use of paradoxical perceptual behaviors. Around the same time John Chowning [cho73], while working on sound spatialization, stumbled upon frequency modulation (FM) as a sound synthesis technique. Being less intuitive an approach than the previously known synthesis techniques, it took Chowning a few years of experimenting before he could control this computationally cheap algorithm to his satisfaction. As he was building up his increasing expertise, he composed *Sabelithe* in 71, *Turenas* in 72, *Stria* in 77 and *Phoné* in 81. By the 80's, he was able to produce sounds that had a human voice quality to them and *Phoné* can be heard as a large FM texture in constant motion from which or to which human voice-like sounds seem to appear and disappear in a very ambiguous way. This ambiguity is another game with the mechanisms of our perception and our ability to group and separate familiar information. We will come back to these perceptual issues later. Of course Risset and Chowning were not the only ones who pushed the doors of Schoenberg's dreams and we could evoke the works of many other composers: J.B. Barrière, T. Murail, etc.

A lack of structure for timbre

Composers such as Risset and Chowning themselves admit the fact that their creativity with timbre manipulation was essentially driven by curiosity and intuition. In an interview he gave to Curtis Roads, Chowning reveals the timbre structures of most of his FM pieces directly reflect some aspects of his newest technical discoveries. In many other cases (*"Desintégrations"* from Tristan Murail for instance) one can wonder if most of the creativity is not dictated by the tool. Some, like Chowning or Risset, simply admit it. After all there is nothing wrong with that; it shouldn't matter where the inspiration comes from. Still it does show one thing and that is the lack of structure for timbre.

While retracing the history of Western music's evolution since antiquity, Hugues Dufourt [Duf88] points out the fact that these "new" representations for musical sounds introduce new bases that are radically independent from the long evolution of our pitch based musical system. They don't intrinsically carry a structure for timbre that resembles the one we inherited for pitch. Therefore he suggests that a structure for timbre should be built from scratch the same way our Greek ancestors dealt with pitch. When it comes to using these representations in order to sculpt a sound texture, the composer has still no other tool than his technological knowledge of the system and an abstract feeling that drives his creativity. There is nothing wrong with using one's feeling while trying to be creative. After all, art is a human expression and it should stay that way. The issue brought up by Schoenberg in his quote is not to prevent the use of one's feeling but rather to possess a vocabulary of "laws" that would allow an objective analysis of a given piece. Being able to put concepts down on a piece of paper, using symbols and rules, is a way for a human mind to relieve its memory (with a discrete set of symbols), clarify its knowledge, and therefore to build up new and more powerful notions and concepts. This is a common point on which people like McAdams [Mca88] and Lerdahl (p.182 of [Bar91]) insist clearly. The need for a symbolic representation reflects the internal mechanism of our mind and perception. As an illustration, knowledge and innovation start with literacy.

At this point we can wonder if the second half of Schoenberg's dream about a structure for timbre is elusive or if it simply belongs to a later future. In what follows we will have a look at some of the major approaches that people have taken in that direction.

Suggested structures for timbre

It is always easier to experiment with a phenomenon than it is to understand it. This is especially true when perception is the major key of the phenomenon. Even though one can't talk about revolutionary results concerning structure of timbre as a musical feature, the problem itself hasn't been ignored. In fact it has been addressed by a wide variety of people from a wide variety of angles. Among the works relevant to this problem, one can discern two major schools. The first one counts both musicologists and psychologists who choose a *top-down* approach by identifying first a set of postulates that such a structure should satisfy and then by trying to fit musical sounds somehow in that scheme. The second school, which represents mostly scientists, is a *bottom-up* approach which starts with empirical measurements of sound perception before working itself up to higher levels of musical features and structures. We will now have a look at both of these approaches.

The "top-down" school

Pierre Schaeffer [Sch77] and his insightful notes about our auditory system has definitely influenced the majority of these works. One of his main insights was to say that it would be vain to consider that music could bypass the essential function of our auditory system, which is to inform us about surrounding phenomena. If our auditory system can provide us with an awareness of the physical world by grouping acoustic information in ways to identify familiar sources or objects, then our perception of tone color should be the result of a similar process of grouping and separation. This observation of common sense gave birth to the notions of *analytic* versus *synthetic* listening, also referred to as *fission* versus *fusion* or *global* versus *local* listening. An *analytic listening process* will try to extract several elementary components from a given sound, leading to the perception of several simultaneous sound objects. A *synthetic listening process* will do the exact opposite, leading to the perception of a single complex *sound object*. McAdams (p.164 of Bar[91]) draws a figure representing the relationship between the dimension *fission/fusion* and the spectral density of a tone.

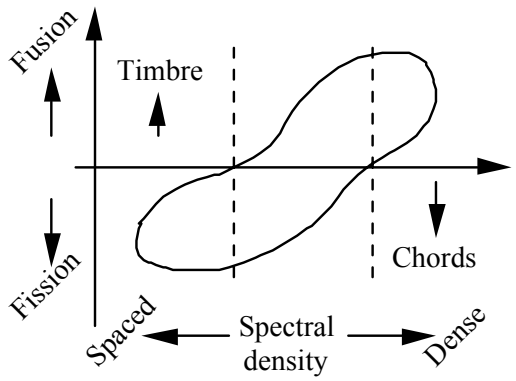


Fig. 2.1 - McAdams' *fusion/fission* diagram. The central shape stands for sound populations. For instance, fission is less likely to occur for sounds that exhibit dense spectra.

It appears that *fusion* occurs for a rather complex but somehow coherent tone and allows the listener to group that complex information into a single sound object *(synthetic listening)*. The perceptual frontier between *fusion* and *fission* is rather unclear and it has to do with the culture as well as the society that the listener belongs to. In his study of Transvaal Venda people's

music and society *"How musical is man?"*, John Blacking [Bla73] writes: *"The sound may be the object, but man is the subject; and the key to understanding music is in the relationships existing between object and subject, the activating principle of organization"*. Of course, this relationship between object and subject is a function of the subject's experience and culture. Psychologists argue that our perception is a heavily filtered version of our world and that rather than perceiving what things are, we tend to perceive what we want them to be. Playing with the threshold of recognition of sound objects is a game that some composers are already familiar with (*Phoné* by Chowning for instance).

Related to this notion of *fusion* is the concept of *consonance*. When fusion occurs, this latest concept measures the *stability* of the perceived object. *Consonance* (or *stability*) is an essential notion when it comes to building a functional structure for timbre. Indeed, both McAdams and Lerdahl agree that elements can not carry form if their functional relationships don't allow phases of tension and release. These phases could be phases of instability and stability that allow the composer to provide his piece with a sense of direction. Going a little further in the use of this notion, Lerdahl proposes a hierarchical organization of timbre based on the relationships between lower level features (such as vibrato and harmonicity) and *consonance*. Harmonicity is indeed a straightforward component of tension as it was illustrated in various timbre-based pieces such as *Le souffle du doux* from D. Arfib or even some parts of *Chréode I* from J.B Barrière.

This top-down school provides very valuable insights on perception as well as some directions for structure and functionality but it lacks an important element: representation. This family of works is still at a stage where it's defining the postulates of an eventual timbre structure. No concrete representation of timbre or sound is actually suggested, and any connection between these abstract postulates and wave-forms is still too vague to contribute in any useful way to sound synthesis.

The "bottom-up" approach

A classic reference for timbre representation is David Wessel's famous timbre space [Wes79]. The first idea is that a coherent quantitative model for a perceptual phenomenon should be based exclusively on perceptual data. Recording perceptual data requires a fair amount of skill and the awareness of a survey's weaknesses. The data recorded by Wessel and Grey were perceptual distances between pairs of sounds presented to a large number of subjects. This data was then fed into a multidimensional scaling procedure which outputs a collapsed version of a space and an arrangement of the original sounds in that space which respects the input distances as well as possible. This way, they ended up with a two-dimensional space of sound which has some perceptual value. The two axes of this space are unknown a priori and that's the main point: we don't know a priori what features are perceptually relevant because if we did, Schoenberg's dream would stop being a dream. In order to have a real continuous timbre space, one needs to interpolate between the original sounds that built that space. Here, Grey and Wessel chose to represent sounds in an additive synthesis fashion. By correlating these parameters along the two unknown axes of their timbre space, they were finally able to attach labels to these axes. The first axis was called *brightness* and is related to the width of the sound's spectrum. The second axis was called *bite* and has to do with the attack of the sound.

The validity and coherence of this timbre space was made clear when Wessel submitted "transposed timbres" to the perceptual judgment of some subjects. Yet, this two dimensional space still lacks functional relationships and the "rules" that Schoenberg is referring to are still to be discovered. As the top-down approach was defining what timbre does but not what it is,

this approach has the opposite problem. We have a representation (what it is) but we don't know much about its functionality (what it does).

It may be that the main weakness of this representation come from its arbitrary relationship with additive synthesis. There are many reasons for referring to additive synthesis and spectrograms but none of these are related to the insightful notion of *sound objects* introduced by Schaeffer. Indeed, it seems reasonable to think that given the essence of our auditory system and its main function, a good representation of timbre should be linked to the physical world. As we will discuss it within this document, the investigation of embedding modeling in the context of musical signals' representation is primarily motivated by filing up this gap between perceptually meaningful and physically meaningful representations. Our approach can be qualified as "bottom-up" as well as we refuse to dictate a specific structure for timbre representation from pure musicology or aesthetics. Rather, we believe that such structures should be derived from the observation of a real system's behavior.

Linear System Theory: Notions and Assumptions

The analysis and the representation of musical signals have traditionally made an extensive use of Fourier transforms, spectral measurements, and similar types of time/scale representations. In order to elevate the analysis or the representation of musical sounds to a higher ground, we must first get familiar with the foundations of the tools that we usually take for granted (linear system theory). In the general context of modeling, issues such as determinism and stochasticity are systematically raised and identifying their interpretation within linear system theory will highlight the limitation of these tools and the need for alternative non-linear modeling techniques.

Assumptions behind spectral analysis

First of all let's recall that the nature of spectral analysis is based on an assumption concerning the randomness of the signals. Indeed, we are viewing sampled sounds here as discrete time, wide-sense stationary stochastic processes x. The spectral distribution of a stochastic process x is traditionally defined as the Fourier transform of its autocorrelation function.

$$S_x(f) = \sum_{n \in Z} R_x(n).e^{-2i\pi nf}$$

As the autocorrelation of an ergodic process can be written as a convolution product of the process with its reversed $R_x(n) = x(n) \otimes x(-n)$, it is fairly common to encounter the following expression for a process' spectrum:

$$S_x(f) = FT[R_x(n)] = \big(FT[x(n)]\big)\big(FT[x(-n)]\big) = \big(FT[x(n)]\big)\big(FT[x(n)]\big)^* = \big\|FT[x(n)]\big\|^2$$

In fact this definition is not completely correct as this series may not always converge. The correct notion of the spectral distribution relies on the concept of the spectral measure and the foundation of this notion is the *Bochner theorem*.

Theorem: Let $(r_n)_{n\in Z}$ be a sequence of elements of C. The sequence $(r_n)_{n\in Z}$ is positive semi definite (i.e. the function $K(m,n)=r_{m-n}$ is positive semi definite) if and only if there is a positive measure μ on I such that

$$r_n = \int_I e^{2i\pi nf}\mu(df) \qquad \forall\, n \in Z$$

In this case the measure μ is uniquely defined.

If $x=(x_n)$ is a discrete-time stationary stochastic process and $(R_X(n))$ its autocorrelation, then this autocorrelation function is a positive semi definite sequence and one can define uniquely a positive measure $\mu_X(df)$ such that

$$R_x(n) = \int_I e^{2i\pi nf}\mu_x(df) \qquad \forall\, n \in Z$$

this measure is called the ***spectral measure*** of the process x. When this measure is continuous with respect to the Lebesgue measure, one can find a positive function $S_X(f)$ such that $\mu_X(df)=S_X(f)df$. The process is then said to be purely non-deterministic and this function defines the spectral density of the process x. Substituting this into the previous integral gives us a familiar relationship between the spectrum and the autocorrelation function:

$$R(n) = \int_I e^{2i\pi nf} S_x(f)df$$

So one should realize that the notion of spectrum as a positive (and bounded) function can break down easily if the spectral measure is not continuous with respect to df.

In addition to the assumption that the analyzed sound is a wide sense stationary random process, the notion of spectrum relies also on the ***purely non-deterministic*** property of the process. We will see exactly what this means.

Deterministic / non-deterministic processes

Let $x=(x_n)$ be a discrete time wide sense stationary stochastic process and $(R_X(n))$ its autocorrelation, we recall that one can define uniquely its spectral measure $\mu_X(df)$ by

$$R_x(n) = \int_I e^{2i\pi nf}\mu_x(df) \qquad \forall\, n \in Z$$

Definition: Let's write $H_f(x)$ the vectorial subspace of finite linear combinations of the random variables x_n. H(x) is the Hilbert subspace of $L^2(\mu_x)$ spread by the random variable $(x_n)_n$, i.e. the closure of $H_f(x)$. Also, $H_n(x)$ will designate the subspace spread by x_k for k<=n.

One observation is that $L^2(\mu_x)$ and H(x) are isomorphic. In fact, one can find a unitary operator linking these two spaces. We'll skip the justification of this observation and introduce directly Kolmogorov's isomorphism.

Definition: The Kolmogorov isomorphism associated to the process x is the unitary operator V_x from $L^2(\mu_x)$ to H(x) defined as follows:

$$\left(\sum_n a_n e^{2i\pi f}\right) \leftrightarrow \left(\sum_n a_n x_n\right) = V_x\left(\sum_n a_n e^{2i\pi f}\right)$$

(sums are intended to be finite)

The purpose of linear prediction theory is to evaluate the projection of the random variable x_n onto the spaces $H_{n-p}(x)$ spread by the random variables x_k for k<=n-p (where p>=1). In general, it is fairly easy to evaluate the projection of a vector on a subspace for which we know an orthonormal basis. Therefore in our case where orthogonality is equivalent to uncorrelation, it would be ideal to extract a white random process ν of variance 1 which would be correlated with x and verify:

$$H_n(x) = H_n(\nu) \text{ for any } n$$

In this case, the process x could be written under the causal form:

$$x_n = \sum_{k=0}^{\infty} h_k \nu_{n-k} \quad \text{where} \quad \sum_{k=0}^{\infty} |h_k|^2 < \infty$$

because of the equality $H_n(x) = H_n(\nu)$, we'd get

$$x_n \,/\, H_{n-p}(x) = \sum_{k=p}^{\infty} h_k \nu_{n-k}$$

(where $x_n \,/\, H_{n-p}(x)$ refers to the orthogonal projection of x_n onto the $H_{n-p}(x)$.)

In particular for p=1, we would observe that $x_n - \left(x_n \,/\, H_{n-1}(x)\right) = h_0 \nu_n$. This brings us to the definition of the *innovation process*.

Definition: The **innovation** of a process x is the process I defined as $I_n = x_n - x_n / H_{n-1}(x)$.

x is said to be **deterministic** (resp. **non-deterministic**) when $I_n=0$ (resp. $I_n!=0$). When x is non-deterministic, we can define the normalized innovation process as $\nu_n = I_n \Big/ \left(E\left[I_n^2\right]\right)^{1/2}$.

Intuitively, the variance of the process I_n is a measure of the information brought by the realization of x_n after the random variables x_k had been observed for k<=n-1. The bigger this variance is, the more x will behave as a white noise.

Remark: If x is deterministic, x_n will belong to $H_{n-1}(x)$; but x_{n-1} will belong to $H_{n-2}(x)$, so by substitution, x_n will belong to $H_{n-2}(x)$. Iterating this scheme will prove that x_n belongs to all the $H_k(x)$, which is to say that $H(x) = \bigcap_k H_k(x) = H_n(x)$.

If x is non-deterministic and ν is its normalized innovation, then by construction it is obvious that ν_n belongs to $H_n(x)$, which is to say that $H_n(\nu) \subset H_n(x)$.

The random process x will be said to be ***purely non-deterministic*** when $H_n(\nu) = H_n(x)$.

Wold's decomposition

With the following theorem, Wold states the existence of an orthogonal decomposition of $H_n(x)$. Because of this decomposition, any stochastic process can be written uniquely as a sum of a *deterministic* and a *purely non-deterministic* processes.

Theorem: For any random process x, $H_n(x)$ has the following orthogonal decomposition:

$$H_n(x) = H_n(\nu) \oplus \bigcap_k H_k(x)$$

This decomposition is referred to as *Wold's decomposition*. This result shows us that x is purely non-deterministic if and only if $\bigcap_k H_k(x) = \{0\}$.

Proof: To prove this statement, all we need to show is that a vector y of $H_n(x)$ is orthogonal to all the ν_k for k<=n if and only if it belongs to all the $H_k(x)$. By construction of the innovation, we clearly have $H_n(x) = H_{n-1}(x) \oplus \{\nu_n\}$. Therefore y is orthogonal to ν_n if and only if it belongs to $H_{n-1}(x)$. The same way, $H_{n-1}(x) = H_{n-2}(x) \oplus \{\nu_{n-1}\}$ and so forth. Iterating this process

shows that y of $H_n(x)$ is orthogonal to all the ν_k for k<=n if and only if it belongs to all the $H_k(x)$ (i.e. $y \in \bigcap_k H_k(x)$).

Let's go back to the spectral measure of our process for a moment. As we saw, $\mu_x(df)$ doesn't have to be continuous with respect to Lebesgue's measure df but we can always decompose it uniquely as $\mu_x(df) = S_x(f)df + \mu_x^s(df)$, where $\mu_x^s(df)$ is a singular measure with respect to df (i.e. the support of $\mu_x^s(df)$ is of measure zero with respect to df). There is a strong relationship between Wold's decomposition, the decomposition of the spectral measure, and the notion of determinism.

Theorem: Let x be a non-deterministic process and ν its normalized innovation. Then the two random processes defined by

$$y_n = x_n / H_n(\nu) \quad \text{and} \quad z_n = x_n / \bigcap_k H_k(x)$$

are respectively purely non-deterministic and deterministic, moreover we have:

$$\mu_y(df) = S_x(f)df \quad \text{and} \quad \mu_z(df) = \mu_x^s(df)$$

A few words about the proof: We already know from earlier that any element of $H_n(\nu)$ will be purely non-deterministic and that any element of $\bigcap_k H_k(x)$ will be deterministic.

An element of $H_n(\nu)$ is a linear combination of a white noise, which can be seen as the output of a linear filter excited by a white noise. A recall to linear filtering will tell us that the power spectrum of such a process is the square of the absolute value of the transfer function of that filter and that therefore, it will be a nice continuous function.

Any element y of $\bigcap_k H_k(x)$ being linearly predictable, it will obey a relationship of the form $y_n = \sum_k a_k y_{n-k}$ which can be interpreted as G(z)Y(z)=0 where $G(z) = 1 - \sum_k a_k z^{-k}$. In the spectral domain, this last form (seen as a linear filtering scheme) implies that $\int_I \left|G(e^{2i\pi f})\right|^2 \mu_y(df) = 0$. The spectral measure of y being non-negative, it should equal zero everywhere except on the countable set of points on which $\left|G(e^{2i\pi f})\right|^2$ is zero. If we are not convinced that this set is indeed countable, we can define $S(z) = G(z)G^*(1/z)$ (which implies that $S(e^{2i\pi f}) = \left|G(e^{2i\pi f})\right|^2$) and then realize that, being factorable, it should verify the Paley-Wiener condition which states that

$\int_I \left| \ln\left(S(e^{2i\pi f})\right) \right| df < \infty$. This proves that the spectral density of Y will be singular with respect to df, being in the form: $\mu_y(df) = \sum_k c_k \delta(f - f_k)$.

The final step to the proof of this theorem is to use the uniqueness of the spectral measure's decomposition and the orthogonality of Wold's decomposition.

This is the essence of the relationship between the purely non-deterministic property of a process and the meaning of its power spectrum.

Entropy, Information and Redundancy

Entropy

Let's consider a particular random variable x, S the set of its possible values, and p_i its probability distribution. If the random variable is of discrete-type, its distribution p_i implies a partition of S. The notion of entropy is an established measure of the uncertainty of this partition.

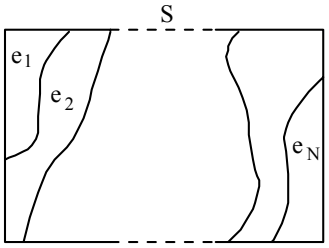


Fig. 2.2 - Partition implied by a discrete-type random variable ($Pr\{e_k\}=p_k$).

The precise definition of entropy of a partition was derived from a set of postulates imposed by our understanding of uncertainty and the uniqueness of a function verifying these postulates. The entropy of a discrete-type random variable is defined as the entropy of this partition.

Definition: The entropy H(x) of a discrete-type random variable x is defined as:

$$H(x) = -\sum_i p_i . \ln(p_i) = E\{-\ln(p(x))\}$$

In the case where x is a continuous-type random variable, we can't directly relate a partition to it any longer. We are reduced to forming a discrete-type random variable by rounding x as follows:

$$x_\delta = n\delta \quad \text{if} \quad (n-1)\delta < x \le n\delta$$

which leads to $P\{x_\delta = n\delta\} = \int_{(n-1)\delta}^{n\delta} p_x(X).dX \approx \delta . p_x(n\delta)$

but then,

$$\begin{aligned} H(x_\delta) &= -\sum \delta . p_x(n\delta).\ln\left(\delta . p_x(n\delta)\right) \\ &= -\ln(\delta) - \sum \delta . p_x(n\delta).\ln\left(p_x(n\delta)\right) \xrightarrow[\delta\to 0]{} \infty \end{aligned}$$

Therefore, H(x) can't be defined as the limit of $H(x_\delta)$ when δ gets infinitely small, but as the limit of the sum :

$$H(x) = \lim_{\delta\to 0}\left[H(x_\delta) + \ln(\delta)\right] = -\int p_x(X).\ln\left(p_x(X)\right)dX$$

Definition: The entropy H(x) of a continuous-type random variable is defined as:

$$H(x) = -\int p_x(X).\ln\left(p_x(X)\right)dX = E\left\{-\ln\left(p_x(x)\right)\right\}$$

The generalization from the discrete to the continuous cases is not intuitive. This should be kept in mind when one wants to estimate the entropy of a continuous-type random variable from a sampled and quantized version of its observation. After all, a computer will only manipulate discrete and finite resolution data and a blind estimation of the entropy of this discrete data might be a heavily biased measure of the original random variable's entropy.

When x is a random vector, these definitions generalize in a straightforward way through the notion of *joint entropy* (or *block entropy*) as follows:

$$H(x,y) = E\left\{-\ln\left(p_{x,y}(x,y)\right)\right\}$$

We can also define *conditional entropy* using this expectation form.

Mutual information and redundancy

Based on the notion of unions of partitions, the mutual information of two random variables x and y is defined from entropy as the function:

$$I(x,y) = H(x) + H(y) - H(x,y)$$

which yields to:
$$I(x,y) = E\left\{ \ln \frac{p_{x,y}(x,y)}{p_x(x)p_y(y)} \right\}$$

and also to $I(x,y) = H(x) - H(x|y) = H(y) - H(y|x)$ (using Bayes' rule)

When more than two random variables are involved, the mutual information can be generalized to the notion of *joint mutual information* as:

$$I(x_1, x_2, .., x_d) = \sum_{k=1}^{d} H(x_k) - H(x_1, x_2, .., x_d)$$

and another useful object is the *redundancy*, defined as the increment of the joint mutual information as the number of random variable grows:

$$\begin{aligned} R(x_1, x_2, .., x_d) &= I(x_1, x_2, .., x_d) - I(x_1, x_2, .., x_{d-1}) \\ &= H(x_d) + H(x_1, x_2, .., x_{d-1}) - H(x_1, x_2, .., x_d) \end{aligned}$$

The case of a sampled strict sense stationary stochastic process

In order to become more familiar with the particular case that will interest us in what follows, let's look at these same objects when the random variables $(x_n)_n$ are samples of a strict sense stationary stochastic process x.

The strict sense stationarity of x states that all of its statistics (of any order) are invariant through time:

$$p\big(x(t_1), x(t_2), .., x(t_d)\big) = p\big(x(t_1 - \tau), x(t_2 - \tau), .., x(t_d - \tau)\big) \qquad \forall (d, \tau) \in N \times \Re$$

So when $(x_n)_n$ designate successive samples of x, this property implies:

$$H\big(x_n, x_{n-1}, .., x_{n-d+1}\big) = H\big(x_k, x_{k-1}, .., x_{k-d+1}\big) \quad \forall (d, n, k) \in N^3$$

which allows us to simplify our notations:

$$p(x_n, x_{n-1}, .., x_{n-d+1}) \equiv p_d(\tau)$$
$$H(x_n, x_{n-1}, .., x_{n-d+1}) \equiv H_d(\tau)$$
$$I(x_n, x_{n-1}, .., x_{n-d+1}) \equiv I_d(\tau)$$
$$R(x_n, x_{n-1}, .., x_{n-d+1}) \equiv R_d(\tau)$$

where τ is the corresponding sampling period.

Another aspect of a time-sampled observation is that it will always be quantized in amplitude. There is no such thing as an infinite resolution measurement and our observation will always look like the instance of a discrete-type random process. We know from before that there's no nice continuity between discrete-type and continuous-type processes when it comes to entropy measurement and so if N is the number of bins imposed by our finite resolution, it is only fair that N should be taken as an extra variable of our estimates:

$$H_d(\tau, N) \; ; \; I_d(\tau, N) \; ; \; R_d(\tau, N)$$

The last point concerns the estimation of the multi-dimensional statistics of the process. Indeed, one has very rarely access to several instances of the same process but rather a single observation in time. The best we can do is to estimate statistics by averaging measurements of this single observation in time. The validity of such an estimation rests on the assumption that the process is ergodic.

Nonlinear Dynamics and the Embedding Theorem

Dynamical Systems

The notion of dynamics can be traced back to the fifteenth century when Newton invented differential equations. The initial enthusiasm surrounding this new invention was quickly replaced by the hopeless realization that most of these equations are impossible to solve in the sense of obtaining an explicit form for the solution. This frustration had to wait a couple of centuries until Poincaré suggested that these equations should be thought of as representations of systems' behaviors. As quantitative questions such as the state of the system at a particular time couldn't be answered, he emphasized qualitative questions such as overall behavior and stability. Any further experimentation and intuitive understanding of nonlinear systems had to wait for the development of high-speed computers in the fifties. Only then were scientists such as Lorenz armed to discover chaotic behaviors, strange attractors and fractals.

There are two types of dynamical systems: differential equations and iterative maps (also referred to as difference equations). The appropriateness of each one depends on whether the problem is set in continuous or discrete time.

A very general form for ordinary differential equations is the system $\frac{\partial \mathbf{x}}{\partial t} = f(\mathbf{x})$, where:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ | \\ x_m \end{bmatrix}$$ is the set of the system's "state" variables.

and $$f(\mathbf{x}) = \begin{bmatrix} f_1(x_1, .., x_m) \\ | \\ f_m(x_1, .., x_m) \end{bmatrix}$$ describes the system's behavior.

Some of these state variables might be external inputs if the system is non-autonomous. An analog form for a general iterative map would be $\mathbf{x}_{t+1} = f(\mathbf{x}_t)$. The linearity (resp. nonlinearity) of the system is the linearity (resp. nonlinearity) of the function f(). Partially due to their limitless variety, most nonlinear systems are unsolvable analytically. The difficulty of their analysis is mainly attributed to the fact that unlike linear systems, they can not be broken down into parts. Without this feature, none of the methods such as Laplace transforms and Fourier analysis hold.

As much as we'd like everything to be a linear system for simplicity's sake, most things in nature don't work this way. As an amusing illustration of our world's nonlinearities, Steven Strogatz [Str94] notes that *"If you listen to your two favorite songs at the same time, you don't get double the pleasure"*.

The Embedding Theorem

Given two spaces A and B, a **mapping** between them is a function f that associates every element α of A with the uniquely determined element $\beta = f(\alpha)$ of B. The element β is then called the image of α and α the preimage of β. When the spaces A and B are metric, the notions of continuity and smoothness can be introduced in that scheme. A C^k mapping that is bijective is called a **diffeomorphism**. A smooth mapping f that is injective is called an **immersion**. If we also want the mapping to preserve topological properties, it will have to be **proper**. A map is proper if the preimage of every compact set is a compact set. Finally, a proper immersion is called an **embedding**. If the details of the definition of an embedding seem a little tedious or obscure, one can think of an embedding as being a smooth local change of coordinates. It might disfigure a subset S of A but will keep its local properties and its fine structure intact.

The following result was stated and proved by Floris Takens in 1981 in his paper "Detecting strange attractors in turbulence" [Tak81] and we present it here in its original form.

Theorem: Let M be a compact manifold of dimension m. For pairs (φ, y), $\varphi: M \to M$ a smooth diffeomorphism and $y: M \to \Re$ a smooth function, it is a generic property that the map $\phi_{(\varphi, y)}: M \to \Re^{2m+1}$, defined by:

$$\phi_{(\varphi, y)}(x) = \left(y(x), y(\varphi(x)), \ldots, y(\varphi^{2m}(x))\right)$$

is an embedding; by "smooth" we mean at least C^2.

This result being a "generic" property means that it is not always true but that the set of cases for which it'll break is of probability 0. In other words, perturbing an unlucky case in an infinitely small way will make the result hold. In the same paper, Takens proves two other theorems of the same flavor that generalize this result to different types of map ϕ. The same year, Mañé in an independent study came up with a similar result.

In the context of a dynamical system $\frac{\partial \mathbf{x}}{\partial t} = f(\mathbf{x})$, the manifold M would typically be the set of a system's states **x**(t) for which, as we said earlier, it is very rare to find an analytic form. The map ϕ is a very general means by which one can refer to a fairly arbitrary switch of the space's nature and this theorem states that in spite of this radical process, the new set $\phi(M)$ carries the same fine structure as the original manifold M. This implies that this new set in $\Re^{2m+1}$ can be interpreted as the solution of a dynamical system which exhibits a behavior that is very similar to the one of the original system $\frac{\partial \mathbf{x}}{\partial t} = f(\mathbf{x})$.

Chapter 3

Machine Listening - Real-time Analysis

If a musical instrument is a sound producing device, it is also a medium of ideas and emotions used by musicians and composers. This vocabulary of musical gestures is directly linked to our ability to parse sound. Without pretending to implement an understanding of music on a machine, this chapter investigates means by which a machine could parse sound at a similar level to our lowest level of perception. The usefulness of such systems is often a function of their ability to run in real-time as their standard applications range from "jamming" with a computer to the development of a "hyperinstrument." As the first step toward the inference of virtual instruments that are controlled by perceptually meaningful parameters, we will identify appropriate perceptually meaningful musical gestures and suggest means by which one can estimate them from an audio stream in real time.

Perceptual Components of a Musical Sound

A musical sound is nothing but a meaningless acoustic wave until it reaches a human auditory system. The musical qualities of this wave are only defined through a set of perceptual components that can be seen as the artifacts of a complex biological mechanism that allows us be aware of these air pressure variations. These qualities are usually addressed in terms of three perceptual components: Volume, Pitch and Timbre. There is no universal or rigorous definition for any of these three components; they are merely assumed to be independent. For our identification of appropriate perceptually meaningful musical gestures, we'll suggest a set of measurements that addresses all three of these.

Volume

Among all the possible perceptual components of a sound one can think of, volume is probably the easiest one to understand and to extract. It is common knowledge that our perception of volume is directly linked to the energy of the acoustic signal via a logarithmic scale and some correction due to the limited bandwidth of our auditory system. These empirical relationships provide a sufficient framework for an accurate measurement of this perceptual component. In other words, it seems straightforward to come up with a rigorous definition of volume that would allow a machine to perceive loudness the same way we do.

Yet, an accurate simulation of human volume perception would take much more than this simple mapping procedure. Experiments on this subject have revealed some hidden complexities through phenomena known as post-masking for instance, and a very precise understanding of volume has been an active subject of study in psychoacoustics.

Pitch

Going one step further in terms of complexity, pitch has obviously played an important part in the development of music. It is directly linked to our ability to differentiate frequencies in sound waves. The study of the human external and internal ear have given clues to theories on how this information is fed to our brain. One observation is that the cochlea, due to its shape, can identify different frequency bands through the localization of its resonances. The *tonotopic* hypothesis is the belief that this phenomenon is responsible for our ability to distinguish between frequencies and it is mainly due to the works of Evans and Stevens. Another school of thoughts is that frequency might be encoded in the impulse trains sent by each nervous cell of the cochlea. This belief comes from the observation of temporal regularities of these impulses but this phenomenon does not occur for high frequencies. In other terms, the tonotopic and the temporal coding hypotheses are equally likely for frequencies up to 5 kHz but for higher frequencies, only the tonotopic hypothesis holds.

Our ability to differentiate frequencies seems to obey a logarithmic rule. As a first approximation, if df is the frequency resolution of our auditory system around the frequency f, then one can observe that the ratio df/f if fairly constant along the full range of audible frequencies. Yet, a more precise study reveals that this ratio degrades as the duration of the stimulus decreases and also that it degrades systematically for frequencies higher than 5 kHz (maybe because the temporal encoding stops being helpful).

The notion of pitch is only relevant in the case of periodic or pseudo-periodic sounds. The pitch of such a tone seems to be directly linked to the inverse of its period. In the case of a nice harmonic structure, this frequency corresponds to the fundamental but it is important to remember that this spectral component doesn't have to be a maximum of energy in order to be perceived as the pitch of a sound. In fact, this fundamental can even be missing completely from the spectral decomposition of the signal. For pseudo-periodic sounds, the best we can do is to associate the perceived pitch to the fundamental frequency of the "most likely" harmonic structure of the signal but we insist here on the fact that this association cannot be taken as a definition. Indeed, such a definition would fail at explaining the perceptual phenomena and the illusions that have been observed in human subjects. As an example, it seems that the perceived pitch of a simple tone may vary as a function of the ambient noise that surrounds the listener. It is a common belief that this curious phenomenon in the context of musicians tuning

their instruments in large orchestras may be partially responsible for the progressive increase of middle A's frequency throughout history.

Timbre

In our context of musical sound representation and characterization, timbre is by far the most interesting one of these perceptual components. It raises numerous questions that have yet to find good answers. This perceptual component needed to be stated, as people realized that there obviously had to be more than simply volume and pitch to musical sounds. A general agreement on a precise definition of this notion is even less realistic than for pitch, and the American Standard Association (ASA) decided in 1960 to define timbre by the negative. This is to say that timbre is the perceptive component that allows us to distinguish two sounds of the same volume and pitch.

Early in the century the works of Helmholtz, inspired by Fourier's transform, led to the **classic conception of timbre**. Helmholtz, and a large number of physicists after him (Bouasse, Olson,...), considered that the majority of the timbral information was contained within the quasi-stationary part of a sound. Over this restricted temporal span, the signal can be written as a Fourier series and it was believed that the spectrum of this harmonic series would define timbre. The simplicity of this definition may seem attractive but when a more flexible recording technology became available, a few basic observations revealed its weaknesses. For instance, this spectrum can vary dramatically with the room's acoustical properties or the listener's position and yet, the timbre of a given instrument is perceived to be unchanged. Another observation is the timbre change one can perceive when playing a sound backwards even though this process doesn't alter the spectral structure given by this Fourier series. All these observations illustrated the fundamental role played by the temporal evolution of sound. Nonetheless, the classic conception of timbre was not completely abandoned, as it presented some interesting features in some cases, and most of the works that subsequently have dealt with timbre still refer to it.

Pitch Extraction

Pitch is undoubtedly the most obvious musical gesture we may want to extract. We shall dedicate this part to the task of pitch estimation and highlight the main issues that are encountered while investigating this well-established problem.

The hidden difficulties

A lack of rigorous definition

Pitch extraction has been approached with a wide variety of methods, each one implying some assumptions about the sound to be analyzed and a definition for pitch. Although the performance of these algorithms has kept on improving, there is no rigorous and indisputable

definition for pitch. Pitched sounds are not always perfectly periodic (time domain) and they can often lack their fundamental frequency (frequency domain). Human auditory systems' ability to switch between analytic and synthetic listening is very often a source of ambiguity as for whether a sound should be considered monophonic or not. Time-domain versus Frequency-domain analysis is the closest analog phenomenon a computer can deal with.

Real time: some intrinsic limitations

It is wrong to consider that our auditory system is faster at recovering pitch from a sound than a computer. In fact, chances are that pitch extraction algorithms are not only more accurate but also much faster than our auditory system. The task of period or frequency analysis has some intrinsic limitations that even our ears have to deal with. The information of pitch needs some time to be resolved. This time is not related to computation but to the amount of sound that one has to listen to in order to make any decent decision concerning an eventual period of the sound wave. Both our auditory system and computers make decision errors, especially during the transitional stage at the beginning of a note. The main difference is the fact that our brain recovers more elegantly from these errors and ambiguities through some sort of post-masking phenomenon. One way to comprehend this perceptual artifact would be to say that our perception of time is flexible enough to allow the merging of related information into a simultaneous event, even when this information was collected during a substantial lapse of time. A machine could very well be programmed to behave similarly and listen passively to a stream of music but such a setup wouldn't meet the expectations one has about a real-time interactive musical system. For a usual task such as pitch following, computers are expected to respond to music much faster than we can. These algorithms are not given the time they need to recover from intrinsic ambiguities, leading to disappointing performances and to the widely spread bad opinion people have about pitch extraction in general. This is not to say that pitch following is an overly ambitious task which should be abandoned but rather that we should readjust our expectations and keep these limitations in mind in order to make a better use of what can be extracted. Decision errors should not systematically be attributed to the bad performance of a system, they should instead be treated as artifacts of our measurements from which much can be learnt. Such an understanding usually leads to more appropriate uses of the algorithms and in the long run, to better performances.

MIDI: the keyboard influence

Let's put ourselves back into the context of real-time pitch following. Once we finally accept the fact that the information set *"a 110Hz A is being played on this instrument with loudness L"* needs some time to be resolved in the incoming sound, we are entitled to ask the question of what to do while some of this information is still ambiguous. In order to be perceptually responsive, our system needs to "play something" within the next 10ms following the first burst of sound. Within this time-frame, a computer can probably make a decent guess concerning the loudness of the incoming sound but it is very unlikely that this incoming sound has already left its transitional stage. In other words, chances are that this first sound frame is not periodic at all. Furthermore, the period of a 110Hz signal is in the same range as the size of this frame. Therefore regardless of the CPU one has, there is no chance one can estimate accurately the pitch, and even less the timbral nature, of this incoming sound within 10ms.

In the case of a keyboard, a complete description of the sound produced can be **inferred** from the initial physical action on a key and this justifies the concept of *NOTE ON* and *NOTE OFF* on which MIDI is based. Keyboard players were the first consumers of sound synthesis and

MIDI was developed around their needs, capturing physical actions on a keyboard-like controller and sending them to a large variety of computers and sound modules. This protocol (or language) was never intended to be extracted from an audio stream as it relies on a very strong assumption concerning the nature of the instrument that is being played.

The omnipresence of MIDI in computer music misled us to believe that it stood for an atomic representation of a musical stream. People refer to "audio to MIDI" devices as if the space of sounds and the space of notes were related through some sort of systematic transform. The previous illustration of a 110Hz A illustrates the fundamental differences in nature of these two worlds. A complete description of a note is not a low-level musical gesture. It is already a musical intention. Of course, there are lots of instances where MIDI users have addressed these issues by using continuous controllers; we are not criticizing MIDI as a transfer protocol but rather as an elementary language for musical events.

From MIDI to musical gesture

Given a small chunk of a monophonic signal, one can attempt to estimate the most likely period of the incoming sound in various ways. An ambiguity will always be associated with the result of this estimation. Again, this ambiguity should not be systematically attributed to a weakness of the chosen algorithm for pitch estimation. In fact, it is itself an additional measurement of the incoming sound stream and it shouldn't be ignored. As we receive successive chunks of audio, both of these values (most likely period and ambiguity) will provide us with a precious musical gesture.

Given these specs, the system is now relieved, at least partially, from the difficult task of making perceptual decisions and it is more likely to provide us with what we expect. If we decide later that we need to infer a score of what is being played, we might decide to threshold this ambiguity value. If real-time is an issue (such as for a MIDI pitch follower), then we will have to accept the fact that such a process will introduce a delay. The only way to overcome this delay would be to avoid making this perceptual decision throughout the musical "communication chain" and leave it to the listener's brain which is subject to post-masking.

Pitch contour estimation in the time-domain

We will now describe an algorithm which estimates the pitch contour and the associated ambiguity of an incoming audio stream. This method does not use any Fourier decomposition, which is why it is qualified as being "time-domain". It was originally inspired by an algorithm described in [MYC91] in the context of speech signals. It was then modified and re-interpreted for music pitch following by the author for the purpose of a Hyperinstrument piece written by Tod Machover for the violin and orchestra *"Forever and Ever"*, premiered in Saint-Paul in the fall '93.

Statement of the problem - Approach

This method assumes that the incoming sound is monophonic. Given this assumption, we hereby adopt the "most likely period" to be the basis of pitch. If the sound were not monophonic (more than one pitch) then this definition would not hold. Given this definition, we will now attempt to solve this estimation problem through a Bayesian scheme.

Let s(t) be our input signal. Assuming that s(t) is quasi-periodic (with pseudo period T) implies that one can write s(t) ≈ a s(t+T) where 'a' is a scalar that accounts for eventual damping or other amplitude changes.

Let's pick an arbitrary number of samples d, a sampling period τ, and let's define the following objects:

For t, $\mathbf{v}(t) = [s(t), s(t+\tau), \ldots, s(t+(d-1)\tau)]^T$.

For T, ω_T is the class corresponding to the hypothesis that T is a pseudo period for s(t).

Given what we consider a pseudo period to be, it seems reasonable to infer the following class conditional densities for **v**(t) given ω_T:

$$p_{\mathbf{v}(t)|\omega_T}(\mathbf{v}(t)|\omega_T) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{\|\mathbf{v}(t) - a(t,T)\mathbf{v}(t+T)\|^2}{2\sigma^2}\right)$$

where σ^2 is a sort of "tolerance" and a(t,T) is such that it minimizes $\|\mathbf{v}(t) - a(t,T)\mathbf{v}(t+T)\|^2$.

$$\text{i.e. } \frac{\partial}{\partial a(t,T)}\left(\|\mathbf{v}(t) - a(t,T)\mathbf{v}(t+T)\|^2\right) = 0 \quad \text{i.e. } a(t,T) = \frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\|^2}$$

Given some observation of **v**(t) for various value of t, our estimation of the most likely period $\hat{p}$ will result from the choice of the hypothesis ω_p which will maximize the posterior probability $Pr[\omega_T|\mathbf{v}(t)]$ to which we can apply Bayes' rule:

$$\hat{p} = \arg\max_T \left(Pr[\omega_T|\mathbf{v}(t)]\right) = \arg\max_T \left(\frac{p_{\mathbf{v}(t)|\omega_T}(\mathbf{v}(t)|\omega_T)\ Pr[\omega_T]}{p_{\mathbf{v}(t)}(\mathbf{v}(t))}\right)$$

$Pr[\omega_T]$ is the prior for T being the period of a signal in general and it is fair to assume that across all possible periods T, this prior is uniform. Furthermore , the denominator $p_{\mathbf{v}(t)}(\mathbf{v}(t))$ does not depend on T so we end up with:

$$\hat{p} = \arg\max_T \left(p_{\mathbf{v}(t)|\omega_T}(\mathbf{v}(t)|\omega_T)\right)$$

In view of the form of the class conditional densities for **v**(t) given ω_T, this leads to:

$$\hat{p} = \arg\min_T \left(\|\mathbf{v}(t) - a(t,T)\mathbf{v}(t+T)\|^2\right), \text{ where } a(t,T) = \frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\|^2}$$

$$\text{i.e. } \hat{p} = \arg\min_T \left(\|\mathbf{v}(t)\|^2 - 2\frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\|^2}\mathbf{v}^T(t)\mathbf{v}(t+T) + \left(\frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\|^2}\right)^2 \|\mathbf{v}(t+T)\|^2\right)$$

i.e. $$\hat{p} = \arg\min_T \left(\|\mathbf{v}(t)\|^2 \left(1 - \left(\frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\| \, \|\mathbf{v}(t)\|} \right)^2 \right) \right)$$

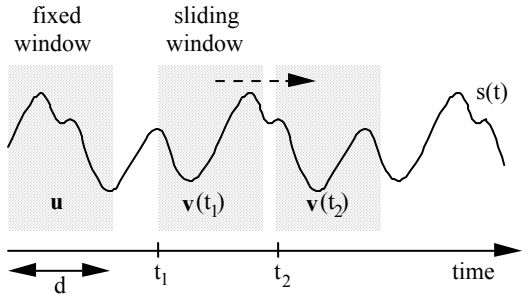
and as $\|\mathbf{v}(t)\|^2$ isn't a function of T, this leads us to:

$$\hat{p} = \arg\max_T \ \lambda^2(t,T) \quad \text{, where } \lambda(t,T) = \frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\| \, \|\mathbf{v}(t)\|} \in [-1,1]$$

Note: $\lambda(t,T)$ will have the same sign as a(t,T). When we introduced the scalar a(t,T) as a way to account for eventual amplitude changes, we obviously implied that it should be positive. A negative sign for a(t,T) would indicate an opposition of phase and not a pseudo period. Therefore $\lambda(t,T)$ should also be positive in order for T to be considered a pseudo period and this leads us finally to:

$$\hat{p} = \arg\max_T \ \lambda(t,T) \text{ , where } \lambda(t,T) = \frac{\mathbf{v}^T(t)\mathbf{v}(t+T)}{\|\mathbf{v}(t+T)\| \, \|\mathbf{v}(t)\|} \in [-1,1]$$

In the real world however, s(t) is sampled (i.e. t is an integer) and in order to estimate the integer value of the pseudo period of s(t), we will have to look for a maximum of this "normalized correlation" between fixed and sliding windows on a chunk of signal s(t) (t=0,...,N-1). These two windows (or vectors) are defined as follows:



$$\mathbf{u} = [s(0),\ldots,s(d-1)]^T \text{ and } \mathbf{v}(n) = [s(n),\ldots,s(n+d-1)]^T,$$

Once again, d is some integer (length of these windows). The "normalized correlation" between these two windows will be given by:

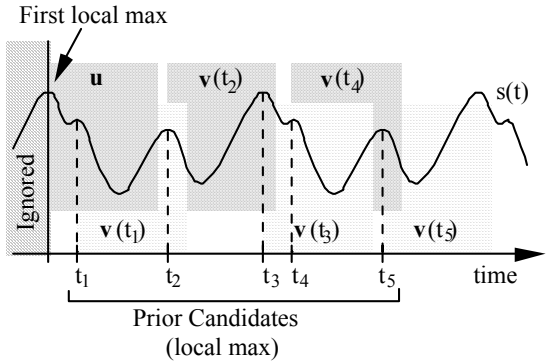
$$\lambda(n) = \frac{\mathbf{u}^T\mathbf{v}(n)}{\|\mathbf{u}\| \, \|\mathbf{v}(n)\|} \in [-1,1],$$

and the integer part of the period will be some integer p_0 (preferably the smallest one) which will maximize $\lambda(p_0)$.

Simplifications and short-cuts

The maximization of the correlation $\lambda(n)$ doesn't necessarily imply that one should compute this function for all possible values of n (n=0,...,N-d). In fact, a very quick (and computationally cheap) observation of s(t) can already provide us with some insights that will suggest a small number of prior candidates ($t_1, t_2,...,t_k$) for the period of s(t).

We can choose to align our reference window **u** with the first local maximum we observe in the current chunk of s(t) we have access to. With this choice, we already know that it is no use to consider sliding windows that are not aligned with a local maximum of s(t). Furthermore, we can also decide that we will consider only the sliding windows that are aligned with a local maximum of the "same range" (amplitude check) than the one the reference **u** was aligned with. This very simple and computationally cheap local maximum lookup will provide us with a fairly small set of prior candidates. The process is illustrated by the following figure.



Note: The presence of parasite high frequency components (noise) might obviously confuse the process of finding these local maxima and it is recommended for the signal s(t) to be previously low-pass filtered. The particular choice of this low-pass filter is not a big issue and in the context of this work, a simple IIR of order 2 works fine.

Resolution - Fractional part of the pseudo period

Let p_0 be the integer period that we've estimated through the previous process and p be the real period of the incoming sound. We know that $|p - p_0| \leq 1$ but let's evaluate this error in the frequency domain:

$$\frac{\Delta f}{f} = \frac{\left| 1/p - 1/p_0 \right|}{1/p} = \frac{|p - p_0|}{p_0} \leq 1/p_0$$

So if we recall that a semi-tone corresponds to a $\frac{\Delta f}{f} = \sqrt[12]{2} - 1 \ (= 0.0595)$, we realize that the resolution given by this integer period is not acceptable if $p_0 \leq 15 \text{ or } 20$ (the frequency

resolution becomes comparable to a semi-tone). Increasing this resolution will require us to interpolate between samples of s(t) in order to find the α such that:

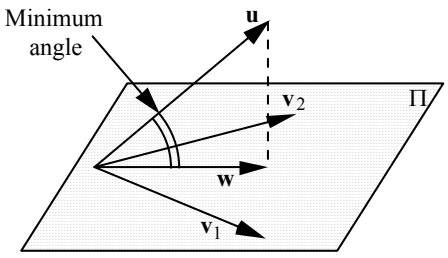
$$\alpha = \arg\max_{\beta\in[0,1]} \lambda(p_0+\beta) = \arg\max_{\beta\in[0,1]} \frac{\mathbf{u}^T\tilde{\mathbf{v}}(p_0+\beta)}{\|\mathbf{u}\|\,\|\tilde{\mathbf{v}}(p_0+\beta)\|}$$

(This assumes of course that p_0 is indeed the integer part of p i.e. $p_0<=p$)

The choice of a linear interpolation is by far the most elegant as it avoids another recursion and the computation of the correlations $\lambda(p_0+\beta)$. Indeed, a linear interpolation would give us:

$$\forall\beta\in[0,1],\quad \tilde{\mathbf{v}}(p_0+\beta) = (1-\beta)\,\mathbf{v}(p_0)+\beta\,\mathbf{v}(p_0+1) \quad \in\Pi$$

where Π is the plane that $\mathbf{v}_1=\mathbf{v}(p_0)$ and $\mathbf{v}_2=\mathbf{v}(p_0+1)$ span. Therefore the optimal value α should correspond to the minimum angle between the vector **u** and vectors belonging to Π. We already know (from Pythagorus) that this minimum is achieved by the orthogonal projection **w** of **u** onto Π.



Hence, α doesn't need any recursion to be estimated as we already know that it will be such that $(1-\alpha)\,\mathbf{v}_1+\alpha\,\mathbf{v}_2$ and **w** are parallel. In the non-orthonormal basis ($\mathbf{v}_1$, $\mathbf{v}_2$) of Π, this property is equivalent to the equality:

$$\frac{w_1}{(1-\alpha)}=\frac{w_2}{\alpha} \text{ where } \mathbf{w}=w_1\mathbf{v}_1+w_2\mathbf{v}_2$$

The coordinates (w_1,w_2) of **w** in ($\mathbf{v}_1$,$\mathbf{v}_2$) are derived from:

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \left[\mathbf{Y}^T\mathbf{Y}\right]^{-1}\mathbf{Y}^T\mathbf{u} \text{ where } \mathbf{Y}=\left[\mathbf{v}_1;\mathbf{v}_2\right]$$

And by combining this with the previous relationship, we finally get a closed form for α:

$$\alpha = \frac{(\mathbf{u}^T\mathbf{v}_2)\|\mathbf{v}_1\|^2-(\mathbf{u}^T\mathbf{v}_1)(\mathbf{v}_1^T\mathbf{v}_2)}{(\mathbf{u}^T\mathbf{v}_2)\left[\|\mathbf{v}_1\|^2-(\mathbf{v}_1^T\mathbf{v}_2)\right]+(\mathbf{u}^T\mathbf{v}_1)\left[\|\mathbf{v}_2\|^2-(\mathbf{v}_1^T\mathbf{v}_2)\right]}$$

Although this expression might look somewhat tedious, half of the scalar products it involves have previously been computed in order to estimate the integer part of the period.

Note: This expression does not guarantee that the computed value for α will be in [0,1]. It could be, for instance, that the estimated integer part of the period is off by 1 (or more). If the computed value for α is negative, then we will try again using (p_0-1) for the integer part. If the computed value for α if greater than 1, we will try (p_0+1).

Recapitulation

Adopting the most likely period as the basis for pitch contour estimation, we've discussed the mechanism of a time-domain pitch follower. At any time, the most likely period of the audio stream is estimated by the means of the maximization of an intercorrelation between a fixed and a sliding window of samples. In addition, the frequency resolution of such an estimator is not limited by the sampling of the audio stream as one can estimate the fractional part of the most likely period in terms of an orthogonal projection. Finally, this method provides naturally a notion of pitch ambiguity in terms of the measured maximum of intercorrelation, We will see in what follows that this ambiguity deserves the status of a perceptually meaningful musical gesture as much as the pitch contour itself.

The use of intercorrelation measurement for pitch extraction is by no means an original idea. Most pitch extraction techniques make reference to similar ideas ([MYC91]) and they often only differ in their philosophy and implementation.

Timbre Listening

Pitch ambiguity/Noisiness

As we pointed it out earlier, an ambiguity will always be associated with the estimation of the signal's pitch. In the context of the time-domain approach that we've presented, this ambiguity is related to the maximum correlation between a fixed and a sliding window. There is a clear relationship between this maximum of correlation and the notion of a signal to noise ratio. Indeed, we will hereby pose the pitch estimation problem in terms of a maximization of signal to noise ratio and show that the outcome is rigorously equivalent to the maximization of the normalized correlation we defined previously.

Keeping the previous notations for our fixed and sliding window, let's write:

$$\mathbf{u} = a(t)\mathbf{v}(t) + \mathbf{e}(t) \quad \text{where a (t) is a scalar}$$

e(t) can be interpreted as an additive noise which will get smaller as t reaches the period of our signal. As we saw earlier, a(t) is a scaling factor which will attempt to minimize the energy of **e**(t). Let's define the signal over noise ratio:

$$SNR(t) = \frac{\|\mathbf{u}\|^2}{\|\mathbf{e}(t)\|^2} = \frac{\|\mathbf{u}\|^2}{\|\mathbf{u} - a(t)\mathbf{v}(t)\|^2} = \frac{\|\mathbf{u}\|^2}{\|\mathbf{u}\|^2 - 2\ a(t)\ (\mathbf{u}^T\mathbf{v}(t)) + a^2(t)\ \|\mathbf{v}(t)\|^2}$$

Maximizing SNR in terms of a(t) will be achieved by minimizing the denominator of the preceding expression. Of course we end up with the same expression that we got earlier:

$$\frac{\partial}{\partial a(t)}\left(\|\mathbf{u}\|^2 - 2\ a(t)\ (\mathbf{u}^T\mathbf{v}(t)) + a^2(t)\ \|\mathbf{v}(t)\|^2\right) = 0 \quad \Rightarrow \quad a(t) = \frac{\mathbf{u}^T\mathbf{v}(t)}{\|\mathbf{v}(t)\|^2}$$

which leads us to the following expression for SNR(t):

$$SNR(t) = \frac{1}{1 - 2\ \frac{\left(\mathbf{u}^T\mathbf{v}(t)\right)^2}{\|\mathbf{v}(t)\|^2\|\mathbf{u}\|^2} + \frac{\left(\mathbf{u}^T\mathbf{v}(t)\right)^2}{\|\mathbf{v}(t)\|^4\|\mathbf{u}\|^2}\ \|\mathbf{v}(t)\|^2} = \frac{1}{1 - \lambda^2(t)}$$

It is now clear that the maximization of this signal over noise ratio is rigorously equivalent to the maximization of our normalized correlation. In other words, the ambiguity resulting from our pitch estimation is not only a measurement of reliability but it is a true measurement of timbre which deserves the status of musical gesture. To the author's knowledge, such measurement has never been used before in the context of musical sounds' characterization. In Chapter 7, we will refer to it in terms of "noisiness" and use it as a control parameter to virtual instruments in conjunction with loudness, pitch, and brightness, which we are about to discuss.

Brightness

The usage of brightness in the context of musical sound is clearly inspired from the works of David Wessel. After having built his 2-dimensional timbre space from perceptual data, Wessel[Wes79] observed a clear correlation between the width of a sound's spectrum and one axis of his timbre space. This led him to define brightness as a measurement of the energy distribution among a sound's harmonics. Such a measurement implies some frequency domain representation of the incoming sound and although the usage of an FFT jumps to mind, we should be aware of some artifacts.

Artifacts of a short-term FFT

First of all, let's recall that an FFT is equivalent to circular convolutions with truncated sine waves. This implies that the FFT of our fixed size chunk of signal will be the Fourier transform of a virtual signal obtained by multiple concatenations of this chunk. If the size of this window is much larger than the signal's period, then the effect of this phenomenon will be minimal but in the context of real-time machine listening, the window size is very often comparable to the signal's period and the effect of circular convolution can be catastrophic. Let's illustrate this effect with a specific case study.

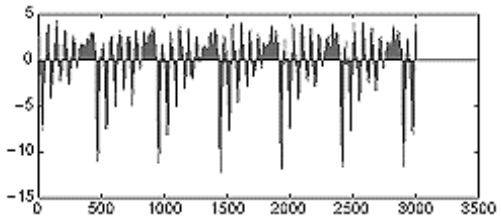


Fig. 3.1 - Vocal recording (period around 505 samples i.e. around 87.3 Hz)

The preceding plot is a quasi-periodic chunk of a 44.1 kHz voice recording. It is obviously pitched and we'd like to measure its brightness in real time based on the center frequency of its spectrum. Once again, "real time" means that we'd like to make an estimation within a 10ms time frame and in this case, this means that this estimation will be based on the observation of no more that 512 samples.

Picking 512 samples from our audio input (this audio recording in our case study) and applying a Hanning window prior to an FFT might sound like a reasonable idea... It's not. Following are two examples of 512-samples chunk we could end up with.

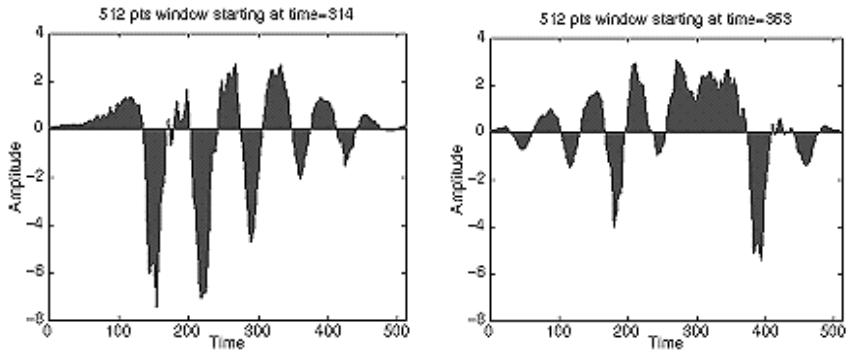


Fig. 3.2 - Two examples of windowed 512-samples taken from the input 1 ms apart.

As we can see, these two chunks already look fairly different although they were taken from the same audio input only 1ms apart. It gets even worse once we realize that feeding these two chunks of signal to an FFT will imply a period of 512 samples for the corresponding analysis. The following plots illustrate this implication.

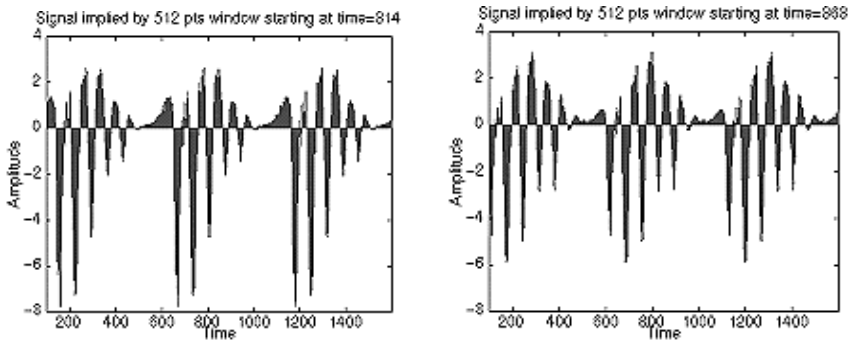


Fig. 3.3 - Corresponding signals "implied" by the FFT analysis. Their period now matches the arbitrary length of the windows and not the pitch of the input.

Each different choice for our 512-samples chunk will result in a different estimate for our spectrum. The following plot illustrates this variety. It was produced by considering 100 different choices for the 512-samples chunk of signal. These successive chunks are only 5 samples apart (about 0.1 ms) and the whole diagram spans only 10 ms of sound.

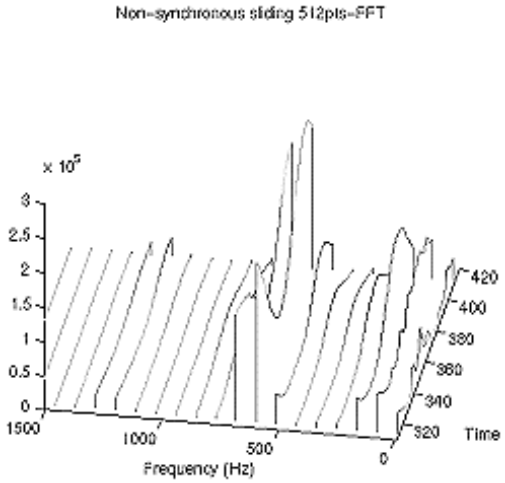


Fig. 3.4 - A variety of spectral estimations.

If we decided to rely on any of these spectral estimations in order to measure the sound's center frequency (the basis of brightness the way we defined it), then we would end up with a variety of different values as illustrated by the following figure.

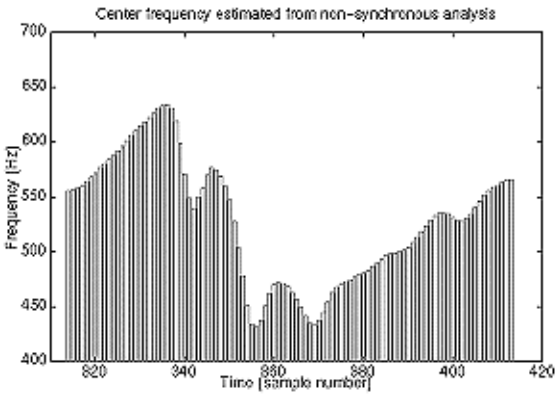


Fig. 3.5 - A variety of estimation of the input's center frequency.

These variations can obviously not be attributed to some variations in our signal given this incredibly short time-frame. Furthermore, this input is quasi-stationary. Given that our "brightness analyzer" will base its estimation on a single frequency analysis, this diagram tells us that it will provide us with any frequency from 425 Hz to 640 Hz as its estimate of the input's center frequency.

Pitch synchronous frequency analysis

However, assuming that we already estimated the period of this sound from what precedes, we can adjust the size of our window in order to match a multiple of this period. This will validate the implicit concatenation of our chunk of sound and justify the FFT's circular convolutions. Such an adjustment of the Fourier window size to the signal's period is known as a *pitch-synchronous frequency analysis*. The idea of pitch synchronous analysis of musical sounds can be traced back to Michael Portnoff's FFT-based phase vocoder [Por76].

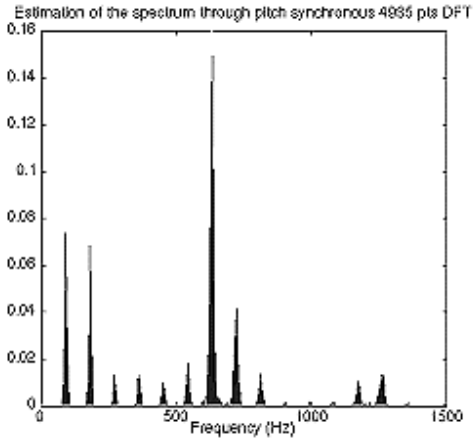


Fig. 3.6 - Long-term estimate of the input's spectrum.

Keeping the same example of our vocal recording and based on a long-term frequency analysis, we find that the center frequency of our input is somewhere around 551 Hz.

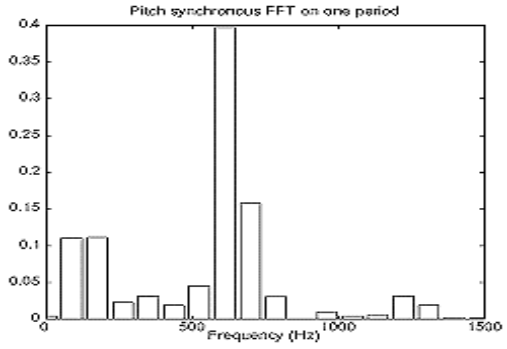


Fig. 3.7 - Result of a pitch synchronous short-term (i.e. 10 ms) frequency analysis.

Based on a single pitch-synchronous frequency analysis, we estimate that the input's center frequency is somewhere around 542 Hz.

Brightness estimator

We recall that we define brightness as a measurement of a sound's center frequency with respect to its fundamental frequency. What precedes clearly justifies the use of pitch-synchronous analysis for our purpose of the extraction of brightness. The estimation of the pitch will result from the time-domain pitch contour follower we introduced previously. The following figure illustrates the mechanism of the brightness estimator.

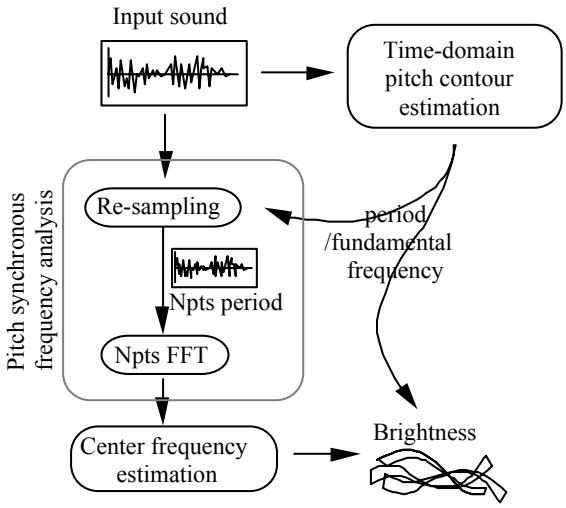


Fig. 3.8 - Brightness estimation by means of a pitch-synchronous frequency analysis.

By now we have gathered enough tools to estimate the major ***musical gestures*** that might interest us from a monophonic audio stream. As we defined them in what precedes, *volume, pitch, noisiness* and *brightness* determine the set of gestures that we will use later on to control virtual instruments. Of course, in spite of what we've restricted our attention to so far, a musical audio stream is very rarely monophonic and it wouldn't be fair to simply ignore

polyphonic audio streams. Without venturing to far in its complexity, the following section is a very fast glance at the problem of polyphonic audio stream analysis.

Analytic Listening

Once again, this section is flirting with the boundaries of this work's scope. In fact, audio scene analysis is a Ph.D. subject on its own and the author recommends the reading of Dan Ellis's dissertation [Ell96] for any further detail on the subject. Throughout our previous sections on pitch extraction and timbre listening, we have implied that the incoming audio source was a single "Sound Object". Back to the notions of *Fusion* and *Fission* we reviewed in Chapter 2, this means that we have restricted our attention to perfect cases of *auditory fusion* (or *synthetic listening*). When confronted to a lack of obvious harmonic relationships or ambiguous variations in a slightly longer time scale, our auditory system provides us with the ability to switch from synthetic to analytic listening, partitioning sonic elements into multiple sound objects which are perceived to be distinct.

Analytic Listening and the Frequency Domain

Whether we are listening to an entire orchestra or to a chord played on a single instrument, we can no longer rely on the existence of a meaningful periodic behavior exhibited by the incoming audio stream. Therefore, most of what we suggested earlier in the case of monophonic signal no longer holds. At this point, everything from the notion of harmonic relationships to MacAdams' observations in Bar[91] points us once again to some sort of frequency domain representation. Whether we decide to use Fourier's decomposition or an alternative constant-Q filter bank or wavelet analysis, a time/scale representation of the incoming audio stream stands for our last hope to emulate our analytic listening ability. Such a representation will systematically break the input sound into multiple "partials" and the remaining task consists of grouping these numerous narrow-band signals into separate but coherent entities. In identifying these entities, one should take both harmonic relationships and coherence of modulations in account.

In a sketchy but successful attempt, the author developed a "harmony analyzer" in the context of an improvisational piece performed by Tod Machover (electric cello) and Anthony Davis (MIDI keyboards) at San Francisco's Yerba Buena center in January '95. The question of analytic listening was raised as the system was expected to extract the overall harmonic content of what was played on the cello (double stops and resonating strings) from nothing but the sound it produced. The real-time constraint prohibited any long windowing of the input signal and any frequency domain estimation had to be very short term.

The resulting harmony analyzer is based on the following mechanism. After a short-term FFT analysis of the input stream, the dynamics of the frequency bins' energy was used to indicate possible new notes and related partials. By "frequency bins" we refer to the coarse frequency resolution filter-bank that is implied by such a short-term spectral estimation. By a simple observation of these bins and a process of "masking" and "grouping" which we'll describe later, the system was able to extract its idea of the "principal notes", taking harmonic relationships in account as well as time occurrences. However, the coarse resolution of the

frequency analysis that was used required a more precise estimation of frequency within each "active" bin of the analysis. For this purpose and given one frequency bin, the system used an approximation for that bin's instantaneous frequency.

Instantaneous frequency approximation

What follows may appear as old news to some readers. However, it is striking to see how many works using FFTs ignore the simple and yet very useful approximation that we are about to discuss. Considering the resolution of an FFT as being an upper limit for the resolution of any further estimated frequency component is a very common mistake. Not only can the instantaneous frequency of a component be estimated within a bin, but in lots of cases, this estimate can be obtained from a single FFT. We shall now illustrate this approximation with a case study.

The following is a plot of the sum of six sinewaves where frequencies and amplitudes were chosen arbitrarily. The values of these frequencies are given in Hz with respect to a 10kHz sampling frequency F_S.

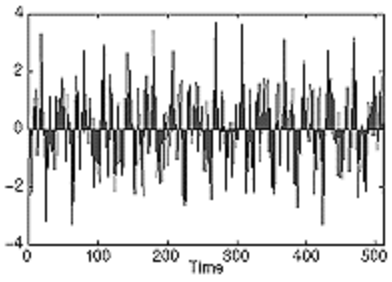


Fig. 3.9 - Sum of six sinewaves (310Hz, 550Hz, 800Hz, 1000Hz, 2425Hz and 3210Hz) at a 10kHz sampling rate (F_S).

Let's consider the discrete Fourier transform of the sampled signal s(n), weighted by a Hanning window:

$$X_k(n) = \sum_{m=0}^{N-1} s(m+n)\, h(m)\, e^{-j\omega mk},$$

where N is the number of points taken in account, $\omega = \frac{2\pi}{N}$ and $h(m) = 1 - \cos(\omega m)$.

Based on this short term FFT with N taken to be 256, an estimation of the signal's spectral distribution from the squared amplitude of the Fourier transform leads to the following diagram (Fig. 3.10). This figure is a plot of the energy associated with the first 85 frequency bins of our 256-point FFT. The first axis is labeled in Hz with respect to a 10kHz sampling frequency.

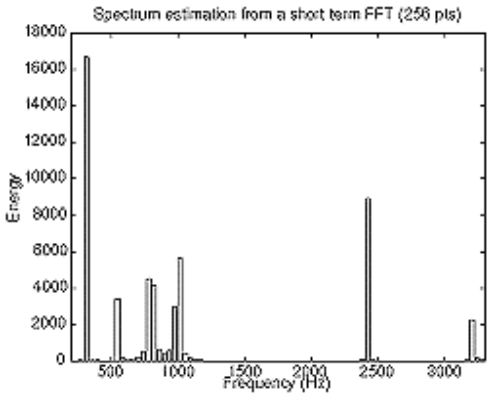


Fig. 3.10 - Spectrum estimation resulting from a 256pts FFT applied to the previous signal. (Bins, numbered from 1 to 256, with energy over 1000: 9, 15, 21, 22, 26, 27, 63, 83.)

An N-point FFT applied to a signal sampled at F_S implies that the bandwidth of each frequency bin is F_S/N. In our particular case study, this means that any frequency estimation based uniquely upon the previous estimation of the signal's spectrum will be biased by a resolution of a little over 39Hz. Furthermore, one can easily observe that we get a few occurrences where two successive bins share a comparable amount of energy, leading to an even greater ambiguity. Blindly using this spectrum estimation may lead us to think that the signal is a weighted sum of the following frequencies: 312.5Hz, 546.88Hz, 781.25Hz, 820.31Hz, 976.56Hz, 1015.63Hz, 2421.88Hz, and 3203.13Hz.

Finally, in the context of musical signals, recall that a semi-tone corresponds to $\Delta f / f = 0.0595$; which is to say that for a pitch in the neighborhood of 100Hz, an ambiguity of 39Hz leads to an ambiguity of almost 7 semi-tones (a fifth). Needless to say, we need much better than this. We will achieve better results by computing the instantaneous frequencies associated with the frequency bins of high energy.

If we write $X_k(n) = \alpha_k(n)\, e^{j\beta_k(n)}$ (polar form) then the instantaneous frequency associated with the k-th bin can be expressed as:

$$F_{inst}(k) = \frac{F_s}{2\pi}\left(\beta_k(n) - \beta_k(n-1)\right) = \frac{F_s}{2\pi}\ \mathrm{Arg}\left[\frac{X_k(n)}{X_k(n-1)}\right] \qquad (3.1)$$

This expression implies the computation of two FFTs. The first one corresponds to a window of samples starting at time 'n' and the other one corresponds to the same window shifted by one sample (time 'n-1'). FFTs are not that expensive and one could very well stop at this expression for the estimation of the instantaneous frequencies. However, a very simple approximation enables us to estimate these instantaneous frequencies from a single FFT.

Let's consider the discrete Fourier transform of s(n) without the Hanning window:

$$Y_k(n) = \sum_{m=0}^{N-1} s(m+n)\, e^{-j\omega mk}$$

The first observation is a simple relationship between our non-windowed FFT and the previous windowed FFT (using a Hanning window):

$$\begin{aligned} Y_k(n) - \frac{1}{2}\left[Y_{k-1}(n) + Y_{k+1}(n)\right] &= \sum_{m=0}^{N-1} s(m+n)\left(1 - \frac{e^{j\omega m} + e^{-j\omega m}}{2}\right) e^{-j\omega mk} \\ &= \sum_{m=0}^{N-1} s(m+n)\,\left(1 - \cos(\omega m)\right) e^{-j\omega mk} \\ &= X_k(n) \end{aligned}$$

The second observation is an approximation relating $Y_k(n-1)$ to $Y_k(n)$. This approximation holds especially because of the absence of any special window applied to s(n) prior to the FFT:

$$Y_k(n-1) = \sum_{m=0}^{N-1} s(m+n-1)\, e^{-j\omega mk} = e^{-j\omega k}\left(\sum_{m=1}^{N} s(m+n)\, e^{-j\omega mk}\right) \approx e^{-j\omega k}\, Y_k(n)$$

Combining these two observations allows us to express both $X_k(n-1)$ and $X_k(n)$ (implying two FFTs) in terms of $Y_k(n)$, $Y_{k-1}(n)$ and $Y_{k+1}(n)$ (only one FFT) as follows:

$$X_k(n) = Y_k(n) - \frac{1}{2}\left[Y_{k-1}(n) + Y_{k+1}(n)\right] \text{ and } X_k(n-1) = e^{-j\omega k}\left(Y_k(n) - \frac{1}{2}\left[e^{j\omega} Y_{k-1}(n) + e^{-j\omega} Y_{k+1}(n)\right]\right)$$

Substituting these into the expression (3.1) for the instantaneous frequencies finally leads us to the following estimate for a bin's instantaneous frequency:

$$F_{inst}(k) = F_s\left(\frac{k}{N} + \frac{1}{2\pi}\,\mathrm{Arg}\left[\frac{A}{B}\right]\right)$$

where $A = Y_k(n) - \frac{1}{2}\left[Y_{k-1}(n) + Y_{k+1}(n)\right]$ and $B = Y_k(n) - \frac{1}{2}\left[e^{j\omega} Y_{k-1}(n) + e^{-j\omega} Y_{k+1}(n)\right]$

Back to our case study, the application of the previous estimate will lead to the following estimation of each bin's instantaneous frequency.

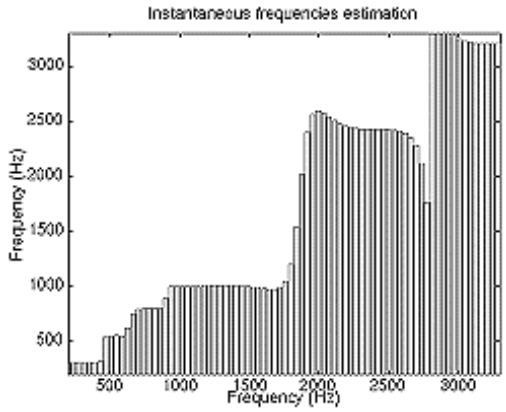


Fig. 3.11 - Estimation of each bin's instantaneous frequency (using a single non-windowed FFT)

Therefore, by looking up the frequency bins that have high energy (from the estimated short term spectrum estimation) and computing the instantaneous frequencies associated with these same bin, we can recover the frequency components of the signal with a much greater resolution than the one implied by the size of the FFTs. The following diagram is the result of such a process applied to our case study.

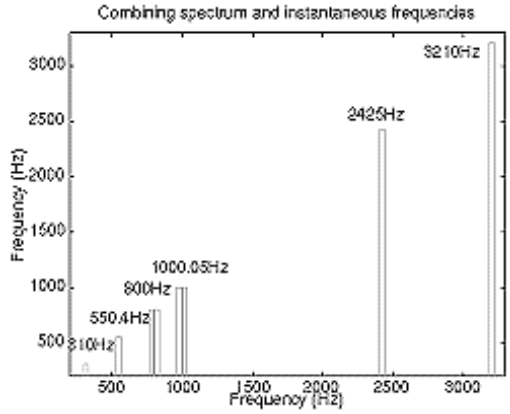


Fig. 3.12 - Recovery of the original frequencies by keeping only bins with high energy (from spectrum estimation) and looking up their instantaneous frequency. The new approximations for the frequency components are: 310Hz, 550.4Hz, 800Hz, 1000Hz, 2425Hz, and 3210Hz.

This instantaneous frequency approximation was extensively used in the context of the harmony analyzer which we are about to discuss.

Harmony Analyzer

This system was originally inspired by fact that the author couldn't afford any commercial MIDI converter for guitar. The idea was to provide a computer with the ability to add a synthetic layer of sounds to any type of guitar playing, without any proprietary hardware such

as a special pickup. The system (named appropriately "GuitarSynth") was fed with the composite sound of the guitar's six strings and was expected to estimate the major harmonic components that were being played. The result of this real-time polyphonic analysis was then plugged directly into a fairly simple wave-table-based synthesis module implemented in software along with the rest of the system. The initial and surprisingly satisfying results of this toy project motivated the author to incorporate it along with the rest of the real-time analysis tools described previously in this chapter. Replacing the sketchy software synthesis part with a full blown MIDI capability (including playing/looping scores and scheduling) eventually led to yet another toy project called the "FunkJammer". This last toy project had the ability to loop a drum track (imposing a tempo) while improvising a bass line based on what was being played on the guitar. In January '95, this harmony analyzer grew out of its original toy status and was incorporated into an improvisational piece played by Tod Machover and Anthony Davis at San Francisco's Yerba Buena center.

A System Walk-Through

The system could be qualified very loosely as a frequency-based multi-pitch extractor. The dynamics of a coarse spectrum estimation based on a short term FFT is used to detect "new notes". This notion of "notes" is to be taken fairly loosely as the system may mask a new note that is in harmonic relationship with a note that is currently playing. Each frequency bin resulting from the initial FFT can potentially become an "active voice". However, at any time, the system keeps track of a frequency mask which attempt to prevent multiple "voices" to be activated by a single sound that has a few harmonics.

The initial FFT is not pitch synchronous (the signal can potentially be polyphonic) and we already know from earlier in this chapter that the estimate for the signal's instantaneous spectrogram will be modulated as an artifact of the window size we chose. In order to minimize the impact of this annoying side-effect, the estimated spectrogram is smoothed (or averaged) in time via a non-linear smoothing process (this is because we want the smooth version to respond more quickly to increasing rather than to decreasing of energy). In addition, the "differential spectrogram" should be regularized before it is passed through some threshold in order to take into account the fact that energies associated with higher frequency bin will tend to be more spastic than for lower frequencies.

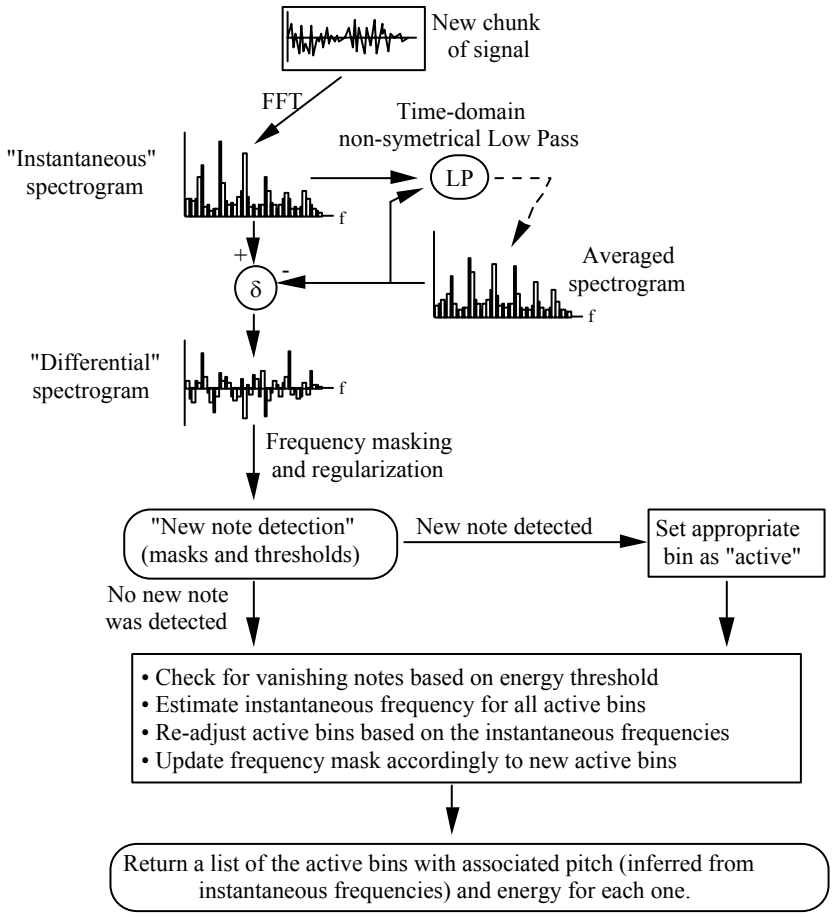


Fig. 3.13 - Walk through the "harmony analyzer"

Chapter Summary

Perceptual Musical Gestures and Real-time Issues

More important than the description of the algorithms suggested in this chapter for the extraction of musical gestures from a sound stream, is the realization that any method will be based on a choice of definition for the values which will be measured. As predicted in the introductory chapter of this document, the fine line between ***musical gestures*** and ***musical intentions*** is the main source of ambiguity for any machine listening task. We clearly stated the lack of indisputable definition for perceptual components such as volume, pitch and timbre and therefore, any specific choice for the definition of a measurement will appear as a coarse simplification over our complex perception of sounds.

A major lesson that should be learned from attempts such as the preceding ones is that rather than ignoring the simplifications and short-cuts imposed by an initial choice of definition, one should keep them in mind and try to take as much advantage of their artifacts as possible. A perfect example was the realization that the ambiguity resulting from the suggested pitch extractor turned out to be a precious measurement of timbre.

As for real-time issues, our flexible and somewhat puzzling perception of time tends to mislead us in expecting much more from a computer than we can achieve ourselves. It is the author's strong belief that delays associated with any machine listening system have very little to do with the quality of the implemented algorithm or even the amount of CPU that was thrown at it. These delays have a more fundamental origin, and that is the non-causal nature of the perceptual components that we're attempting to extract. Our auditory perception is subject to the same non-causality and to probably even worse ambiguities but the fuzzy computer that is our brain has the ability to recover from these ambiguities in a more elegant manner that makes it imperceptible to us. In his discussion of auditory perception, Stephan Handel [Han89], specifically addresses this phenomenon in terms of the identification of perceptual events. In that process, he refers to some experiments that were conducted by Rash in 1978, and which suggested that the perception of a single onset could result from a succession of a couple of audio stimuli that are as much as 30 milliseconds apart. This is related once again to the notion of fusion that we reviewed quickly in Chapter 2 of this document. It turns out that this phenomenon is not proper to auditory perception and that similar artifacts can be observed throughout the whole range of human perception. Handel goes even further relating auditory fusion to the early works of the Gestalt psychologists on the *articulation of visual scenes*. From these elaborate discussions and theories, an interesting statement is that perception (including auditory perception) is probably not a hierarchical process in terms of levels of abstraction. Instead, all levels of abstraction would coexist simultaneously and collaborate in order to recover the most plausible (or simplest) cause that could explain the stimuli. In other words, no perception ever goes without a context.

Such an observation is a plea to reconsider the exact role of a computer in the context of real-time machine listening, by humbly readjusting our expectations. Eventually, a clearer understanding of what we might consider at first as annoying artifacts or weaknesses will always turn in our advantage as we substitute frustration with appropriateness.

Resulting Software Package

All the previously suggested algorithms were compiled along with some underpinning utilities into a musical sound-oriented digital signal processing tool box, as a standard C library ("libtsd.a"). This tool box was originally developed on an SGI Indigo but low level utilities that deal with the audio hardware and audio files represent the only machine specific elements throughout this library. This package was ported successfully and fairly painlessly to the Windows platform by John Yu (EE MS at MIT) for its use in Tod Machover's the Brain Opera.

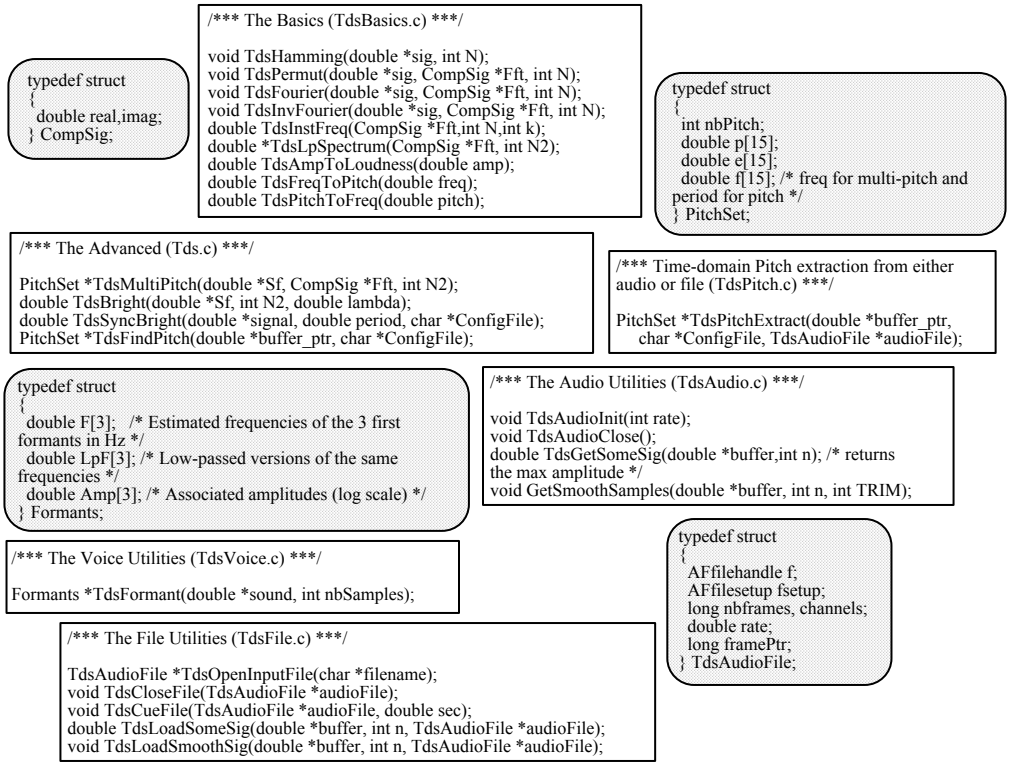


Fig. 3.14 - Real-time sound-oriented digital signal processing tool box (libtds.a)

The "TdsBasics" part of the package implements most of the underpinning tools upon which the more sophisticated functions are based. It includes some general utilities such as Fourier and inverse Fourier transforms, Hanning windows and instantaneous frequency estimation as well as music-oriented converters such as frequency/pitch or amplitude/loudness. The time-domain pitch extractor that we previously discussed is implemented in such a way that it can transparently deal with audio streams from the machine's hardware of audio files (AIFF or AIFC). Brightness estimation and the harmony analyzer are implemented in the "TdsAdvanced" part of the package. Although we will always prefer to apply some pitch-synchronous frequency analysis in order to determine brightness, this package implements a more general version as well which doesn't require the knowledge of the sound's pitch (at the cost of an inferior accuracy and a bigger latency).

A formant follower was added to this package in anticipation of its usage for the Brain Opera. This system is based on a Cepstrum analysis and a chirp Z-transform followed by some peak tracking. Although this piece of code is original, the author didn't feel the necessity to describe precisely the mechanism of this formant tracker as similar systems have been described many times throughout the speech processing literature. For further details, we refer the reader to [SR70] which introduced the chirp Z-transform in this context.

The only machine dependent parts of this package are the utilities that deal with audio hardware ("TdsAudio") and audio files ("TdsFiles").

Applications

"Forever and Ever" (September '93)

The time-domain pitch extraction algorithm was first implemented as a standalone application for the purpose of Tod Machover's third Hyperstring piece "Forever and Ever". This piece was premiered in Saint Paul (Minnesota) in the fall '93, featuring Any Kavafian on the violin and the Saint Paul orchestra. It was also performed along with the two previous pieces of this trilogy ("Begin Again Again", "Song of Penance" and "Forever and Ever") at the Lincoln Center in New York City in July '96. At various stages of this piece, the computer would decide to enhance or sustain notes that were played on the violin. The pitch extraction algorithm we discussed earlier was implemented on an SGI Indigo which was communicating the results of its analysis back to the master computer (Macintosh) of the piece via MIDI. Decisions concerning onset and offset of notes were made locally on the SGI which had no knowledge of the score played by Any Kavafian ahead of time. This setup was an eye opener for the author concerning the intrinsic difficulties associated with the incorporation of such decisions in a real-time environment and the trade-off between responsiveness and ambiguity. However, this software implementation ended up out-performing any commercial alternative that was available at the time (IVL "Pitch Rider" for instance). The expertise of the audio analysis in the context of this piece was limited to pitch and loudness. The pitch ambiguity that resulted from the maximum correlation within the method was used in the context of local decisions concerning onsets and offsets but we hadn't yet realized that this measurement could qualify the timbre of the analyzed sound.

Improvisation for Cello, Keyboards and a Disklavier (January '95)

When San Francisco's Yerba Buena center decided to organize a concert featuring both Tod Machover's and Anthony Davis' compositions, the two composers/musicians suggested the development of an improvisational setup which they would both conclude the concert with. The resulting system (implemented by the author in C and C++ on an SGI Indigo) was a collection of algorithms that would feed on the two musicians' inputs and react appropriately on a Yamaha Disklavier (MIDI controllable grand piano). In addition to providing the minimal set of utilities that is necessary for such MIDI applications (including scores and scheduling), the system needed to make sense out of the cello's output which was nothing but sound. This was a perfect opportunity to consolidate the author's DSP ideas into a single and uniform package.

This project was also a stepping stone for the "harmony analyzer" which we described earlier. The most challenging part of this project was the last section of its third (and last) movement. For this section, the computer had to compare musically the "togetherness" of the two instruments without any previous knowledge concerning what the musicians would play. The harmony analyzer was used to provide the harmonic content of the music played on the cello. This harmonic content would then be wrapped around one octave, leading to a 12-dimensional vector where each component stood for the amount of energy associated with a note in a chromatic scale. A similar measurement was derived from the MIDI stream flowing from Anthony Davis' electronic keyboard. With the help of David Waxman (MAS MS - MIT) and his expertise in music theory, the author came up with a change of coordinate (or linear transformation) for this 12-dimensional vector after which the resulting space would be "harmonically orthogonal". This change of coordinates was derived from a perceptual rating of pitch intervals that David provided. Once projected onto this new set of coordinates, the angle

between the vectors associated to both instrument in that new basis provided the computer with a surprisingly satisfying real-time measurement of the desired "togetherness".

The "Brain Opera" (July '96)

By far the most ambitious project undertaken by Tod Machover and the Music and Media group at the Media Laboratory to this day, the Brain Opera is a 40-computer setup which mobilized a team of no less than fifty people. Based on an interpretation of Marvin Minsky's *Society of Mind*, the project is a large musical interactive installation (the "mind forest") followed by the performance of a trio using alternative gestural instruments which are based on the sensor technology developed by Professor Neil Gershenfeld's Physics and Media group and by Joe Paradiso. The "mind forest" is a set of suspended organic-looking musical experiences which the audience interacts musically with ("Rhythm trees", "Marvin stations", "Melody easels", "Harmonic driving", "Gesture walls", and "Singing trees").

The "singing tree", designed and implemented by John Yu (EE MS - MIT) and William Oliver (EE MS student - MIT), is a system which uses singing as a gestural control over music. Once again, the package described in this chapter was used as the basis for the signal analysis that was required for this project. The designers of this station suggested measuring the voice's formantic structure in order to distinguish between a few phonemes, so the author added a formant tracker to the algorithms that we've described. This package was then ported to the Windows platform which was running the rest of the experience. The music was generated on the fly by a parametric musical engine designed and implemented by John Yu. The parameters of this musical engine were controlled by the output of the voice's analysis via some appropriate mappings developed by William Oliver. The result is a rather involving experience which enables any one to create a rich and responsive musical texture from nothing but their singing. As he or she approaches a custom-designed suspended hood (designed by Maggie Morth, MAS PhD student - MIT) where a microphone, headphones and an LCD screen are embedded, the user is asked to sing and sustain a note of his or her choice. Loudness, pitch contour, noisiness and formantic structures are analyzed on the fly, leading to a set of parameters that characterizes the audio input. As the user attempts to sustain a note, the overall stability of these parameters gets rewarded both musically and visually. On a musical level, this reward consists in a harmonically coherent and stable embellishment of the sung note as opposed to a harsher and more chaotic sound texture which occurs when modulations are detected from the audio analysis. Visually, the user is rewarded by stepping smoothly through video frames towards the end of a sequence while any detected modulation steps the visuals backwards within this sequence.

The simplicity of their purpose and their responsiveness are the major keys to the singing trees' success. The choices that were made concerning the analysis of the incoming audio stream and the mapping to the musical engine turned out to be appropriate. Audience members have no trouble figuring out how to use this installation as they become quickly involved with the musical experience it provides. Yet, the same simplicity has some drawbacks as people sometimes feel they have explored to whole range of the experience too quickly. The trade-off between building self-explanatory setups and systems that provide a sustained degree of involvement is a difficult compromise that one encounters endlessly in the context of interactive installations, but the author will leave this debate open as we are digressing from the scope of this document.

Perceptually and Physically Meaningful Synthesis

Finally, we will see how these perceptually meaningful musical gestures can be used as direct controls over a synthesis engine in Chapter 7.

Chapter 4

Towards Physically Meaningful Models

*Taken as a time series, a sound can be described and modeled in limitless ways. Although it might seem that our primary concern in building a synthesis model should be its accuracy for resynthesis, the real challenge is to make it both **universal** (its ability to represent the widest variety of sounds) and **meaningful** (its behavior as a musical instrument). In this chapter, we will identify the issues and concerns that should be addressed. We will also introduce the general philosophy behind **embedding**.*

The Challenge

There is a clear distinction between modeling a system from observed data and measuring a specific set of features from the data set. Ideally, modeling should be approached without any pre-conception about the system's architecture. The training data should stand for the unique relevant source of information from which our task is to derive as much knowledge and understanding about the system's mechanism as possible. Measuring a specific feature from input data implies the prior choice of a definition for a supposedly relevant feature. Ironically though, these two tasks are traditionally so closely related that their distinction resides only in their purposes and not all that much in their implementation or mechanism. Until recently, linear system theory was the only modeling tool available and its extensive use made us forget about the strong assumptions it relies upon. We shall re-visit these quickly and state their limitations.

The power spectrum/sonogram fascination

Among the most classic references in the domain of timbre characterization are the works of Wessel [Wes79], Grey [Gre75], Slawson [Sla68], Risset [RW82]. All of these make reference to some time/frequency representations (such as sonograms, pitch-synchronous analysis or more rarely wavelets) which add the notion of temporal evolution to Helmoltz' classic conception of timbre. Pitch synchronous analysis can be seen as a special case of a sonogram for which the signal is locally re-sampled at a multiple of its fundamental frequency and assumed to be perfectly periodic.

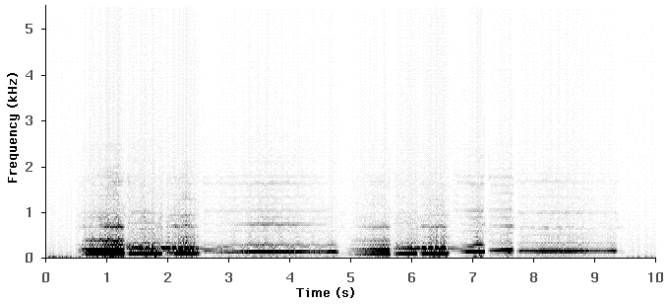


Fig. 4.1 - Example of a sonogram analysis applied to a short melody played on a cello.

Sonograms are nothing but a short-time Fourier analysis applied to a sound. Sonograms involve sliding a short temporal window on the signal and decomposing the windowed observation with Fourier's tool as if it were a piece of an infinite support stationary time series. This representation offers a set of interesting features related to human perception of sound. The main reasons for referring to the Fourier transform of a sound are the following popular beliefs:

(i) The sine functions which are the basis on which Fourier decomposes a signal play a very important physical and perceptive role.

(ii) The spectrum of a sound is a set of n couples (amplitude A_n, frequency f_n) which leads to a multidimensional representation of timbre. Additionally, the spectral envelope (i.e. the series A_n) carries a lot of information in some cases (such as voice for instance).

(iii) The separation between amplitude and phase for each spectral component was confirmed by some studies on phase perception.

(iv) The perceptual notion of harmony within a complex sound can be interpreted through a model based on a set of distinct sine waves in a satisfying way.

Another argument for the use of a spectral representation is the fact that estimates of higher order statistics can seem to be computationally expensive and to require a prohibitive amount of data.

Standard digital signal processing in the real world

In Chapter 2, we reviewed quickly some of the major foundations of linear system theory. In that review, we highlighted the close relationships between the notions of power spectrum, second order statistics, linear systems, innovation, Wold's decomposition and determinism. Although it was short and incomplete, this overview was sufficiently detailed to illustrate how all of these notions are tied up together and how they depend strongly upon each other in order to make any sense.

When observed samples replace stochastic processes and when the analyzing tool is a computer, objects such as autocorrelation functions, measures and the Fourier transform lose most of their theoretical meaning (leaving the clean world of Mathematics for the dirty reality of sampled and quantized measurements). Any estimate of statistics relies on the ergodicity of the system, allowing averages over instances and averages over time on a single instance to be equivalent. Expectations become averages over a limited number of observations and the eventual singularities of the spectral measure become spikes in the periodogram (or other approximate measure of the spectrum). The boundary between determinism and non-determinism introduced earlier relies on a decision rule that detects spikes. But besides these limitations imposed by the nature of digital signal processing, the notions of deterministic and non-deterministic processes suffer from some more intrinsic limitations.

Indeed, the definition of the innovation, as being a white noise uncorrelated with the past values of the process, only takes second order statistics into account. It is constructed on the orthogonality principle which finds its foundation in linear mean-square estimation and can be related to the Wiener-Hopf equation (where the purpose is to match correlation functions). While innovation is related intuitively to a measure of the additional information brought by a new observation when the past is known, we ought to be skeptical. Being uncorrelated with the past doesn't imply being independent from it. In fact, a correct measurement of this additional information requires the ability to estimate the joint probability of an increasing number of successive $(x_n)_n$ and to compute the corresponding entropies. These joint entropies lead to the notions of redundancy and information which are more likely to give a better answer to the "deterministic vs. stochastic" question.

The notion of a deterministic (or predictable) process that is introduced by Wold's decomposition characterizes only a subclass of deterministic systems: deterministic **and** linear systems (the future is a linear combination of the past). A process which, through Wold's tool, may appear to be non-deterministic or even purely non-deterministic, is not guaranteed to be stochastic at all. It might be the chaotic output of a non-linear deterministic dynamical system. The estimates of the second order statistics of a deterministic, but chaotic, system can be amazingly similar to the ones of a random white noise.

We are now aware that the linear approach to signal modeling will give up determinism as soon as the system presents some non-linearities.

So why does linear modeling "work"?

In the context of speech processing as well as musical sound analysis, the incoming signal presents long harmonic or periodic stages. During these stages, the signal fits Wold's decomposition perfectly and it can be qualified as deterministic in the "linear" sense. We will now see how any harmonic or periodic process can be expressed as the output of an autonomous linear system (auto-regressive or AR model).

The exercise

Let x_n be our harmonic stochastic process. x_n being harmonic means that it can be expressed as follows:

$$x_n = \sum_{k=1}^{p} \lambda_k e^{2i\pi n f_k}$$

where the λ_k are centered, uncorrelated random variables of variances γ^2_k. We can observe that a specific arrangement of the f_k and λ_k will be required if x_n takes only real values but what follows can apply regardless. If x_n is a periodic signal, the f_k should follow a harmonic series; the finite sum (k=1 to p) is justified by the assumption of a finite bandwidth for the signal.

As in Chapter 2, the spectral measure of this process will be:

$$\mu_x(df) = \sum_{k=1}^{p} \gamma_k^2 \delta(f - f_k),$$

where $\delta(f)$ refers to Dirac's distribution. Given that expression, let's built the following finite impulse response filter H(z) as follows:

$$H(z) = \prod_{k=1}^{p} (1 - z^{-1} e^{-2i\pi f_k}) = 1 - \sum_{k=1}^{p} h_k z^{-k}$$

Let's apply this linear filter to the process x_n and let y_n be the output of this filter. Then y_n will have the following expression:

$$y_n = x_n + \sum_{k=1}^{p} h_k x_{n-k}$$

and the energy of y_n will be given by:

$$E\left[|y_n|^2\right] = \int \left|H(e^{-2i\pi f})\right|^2 \mu_x(df) = \sum_{k=1}^{p} \left|H(e^{-2i\pi f_k})\right|^2$$

Of course, the filter H(z) was designed to make sure that the sum would be equal to zero and we end up simply with $y_n=0$; in other words:

$$x_n = -\sum_{k=1}^{p} h_k x_{n-k}$$

Therefore x_n is a linear combination of its past values. This relationship is an AR (auto-regressive) linear model for this process and it will lead (at least theoretically) to a perfect reconstruction of the time-series x_n given p initial conditions..

The lesson

This little exercise illustrated how any harmonic or periodic signal can be expressed as the output of an autonomous linear system. If the extent to which an approach "works" is measured in terms of prediction errors, it is the author's strong belief that this phenomenon is the main reason why linear modeling "works".

A call for non-linear modeling

Looking back a little more suspiciously at the previous exercise will lead to a few observations. The degrees of freedom and the general architecture of the resulting linear model are the artificial products of the approach; they don't necessarily reflect the physical (or dynamical) nature of the system that produced this signal.

Any lack of stationarity exhibited by the signal will lead to some energy between the major frequency components which, through the previous scheme, will automatically be assimilated as an additive noise. Therefore, any transitional stage or any modulation may be attributed to some random behavior regardless of their true predictability.

In terms of prediction error, these might not seem to be all that important due to the sound's strong tendency to be quasi-periodic over time. However, any quick listening exercise will convince a listener that purely stationary harmonic sounds (no modulation or envelope) don't carry much information musically. No matter how many harmonics a stationary wave may have, it will always tend to sound the same to us. Sounds only come to life when they start exhibiting modulations or peculiar transitional stages. In a way, it is their deviation from pure stationarity (no matter how small) that provides sounds with their identity.

Modeling Spaces - Embedding

Inferring non-linear models from observed data without any pre-conception concerning the architecture of an eventual model is no longer a dream. In what follows, we will see how Floris Takens' Embedding theorem can be applied to time-series and lead to a general scheme for the inference of physically meaningful models from observed behaviors.

State space and lag space

Let's consider a dynamical system described by its state variables **x** related to each other in a general fashion:

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x})$$

The evolution of the system from a given initial state can be monitored by the trajectory of the vector **x** as time passes by. This vector **x** lives in the ***state space*** and the observation of this trajectory can teach us a lot about the internal mechanism of this dynamic system (i.e. about the relationship f). However, the nature and even the number of these internal states (or degrees of freedom) are usually unknown and we only have access to a subset of them if not only one. Let's suppose the only observation we have is a single variable z=g(**x**). Even though the dimension of our observation is one, we can choose to build a vector of arbitrary dimension d by using lag values of z:

$$\mathbf{l}(t) = \big(z(t), z(t+\tau), \ldots, z(t+(d-1)\tau)\big)^T$$

This vector **l**(t) lives in the ***lag space*** in which it will draw another trajectory as time passes by.

Application of the embedding theorem

Let's recall the formulation of Floris Takens' original embedding theorem from Chapter 2.

> Theorem: Let M be a compact manifold of dimension m. For pairs (φ, y), $\varphi: M \to M$ a smooth diffeomorphism and $y: M \to \Re$ a smooth function, it is a generic property that the map $\phi_{(\varphi, y)}: M \to \Re^{2m+1}$, defined by:
>
> $$\phi_{(\varphi, y)}(x) = \Big(y(x), y(\varphi(x)), \ldots, y(\varphi^{2m}(x))\Big)$$
>
> is an embedding; by "smooth" we mean at least C^2.

Keeping the same notations as above, **x**(t) represents the system's state at time t, z(t) = g(**x**(t)) is our scalar observation at time t. The system has a general dynamical behavior:

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x})$$

Let's pick the manifold M to be the set of our system's states **x**(t). Let's chose the map $\varphi: M \to M$ to be such that $\varphi\big(\mathbf{x}(t)\big) = \mathbf{x}(t+\tau)$. This map represents the system's internal

dynamics and is obviously related to the function f(). Finally, as suggested in the previous theorem, let's define

$$\phi_{(\varphi, g)}: M \to \Re^{2m+1}$$

and

$$\phi_{(\varphi, g)}(\mathbf{x}) = \left(g(\mathbf{x}), g(\varphi(\mathbf{x})), \ldots, g(\varphi^{2m}(\mathbf{x}))\right)$$

$\varphi: M \to M$ and $g: M \to \Re$ verify the hypotheses of the embedding theorem and given that m is picked to be big enough, the map $\phi_{(\varphi, g)}$ should be an embedding.

Given the choices we made concerning φ() and g(), it turns out that $\phi_{(\varphi, g)}$ maps the system's state space to the observation's lag space:

$$\begin{aligned}\phi_{(\varphi, g)}(\mathbf{x}(t)) &= \left(g(\mathbf{x}(t)), g(\varphi(\mathbf{x}(t))), \ldots, g(\varphi^{2m}(\mathbf{x}(t)))\right) \\ &= \left(g(\mathbf{x}(t)), g(\mathbf{x}(t-\tau)), \ldots, g(\mathbf{x}(t-2m\tau))\right) \\ &= \left(z(t), z(t-\tau), \ldots, z(t-2m\tau)\right) = \mathbf{l}(t)\end{aligned}$$

In other words, the trajectory of a 1D observation in a d-dimensional lag space and the trajectory of the system's state in its state space differ only by a smooth local change of coordinates (given that d is big enough). In the context of modeling, classification or resynthesis, this result tells us that there is no need for "hidden variables" other than the lag values of the time series we are studying. Furthermore, the number of necessary lag values is directly related to the number of the system's degrees of freedom and this number can also be estimated by measuring statistics on the initial observation. We can also note that any invertible transformation of these lag vectors will also work. If this transformation were to be linear, this means that any linearly filtered version of the observation works just as well.

The state space of a system is an object we will never have access to whereas the lag space doesn't give us any such problem. This strong relationship between these two objects will allow us from now on to forget completely about the state space and to manipulate lag values of an observation as if they were directly state variables of our system. This is to say that the behavior of our observation in this new space obeys laws similar to those which the instrument obeys to in the physical world. Therefore, characterizing the behavior of our observation in this new space (as a dynamical system) will lead to a physically meaningful model for the instrument.

Embedding Dimension

The sufficient dimension d of the lag space and the number of degrees of freedom are now taken to be equivalent. Let's consider a deterministic system that produces the discrete time

observation $x_n=x(n\tau)$. By "deterministic," we mean here that the past values of this observation allow the prediction of its future values with no error. Unlike the case of Wold's decomposition, no assumption concerning linear relationships is implied in what follows. This can be written as:

$$p\left(x_n \middle| x_{n-1}, x_{n-2}, \ldots\right) = \delta\left(x_n - f(x_{n-1}, x_{n-2}, \ldots)\right) \qquad \text{(Dirac distribution)}$$

For this system to have a finite number of degrees of freedom, there has to be a dimension d such that:

$$\begin{aligned} p\left(x_n \middle| x_{n-1}, x_{n-2}, \ldots\right) &= p\left(x_n \middle| x_{n-1}, x_{n-2}, .., x_{n-d}\right) \\ &= \delta\left(x_n - f_d\left(x_{n-1}, x_{n-2}, .., x_{n-d}\right)\right) \end{aligned}$$

If such a dimension exists, and given what we said before about sampled strict sense stationary stochastic processes in Chapter 2, then we'd have:

$$\begin{aligned} p_{k+1}(\tau) &= p\left(x_n, x_{n-1}, x_{n-2}, .., x_{n-k}\right) \\ &= \delta\left(x_n - f_d\left(x_{n-1}, x_{n-2}, .., x_{n-d}\right)\right) p\left(x_{n-1}, x_{n-2}, .., x_{n-k}\right) \\ &= \delta\left(x_n - f_d\left(x_{n-1}, x_{n-2}, .., x_{n-d}\right)\right) p_k(\tau) \qquad \text{for any } k \geq d \end{aligned}$$

And therefore, for any k>=d, we'd have:

$$\begin{aligned} H_{k+1}(\tau) &= H_k(\tau) \\ I_{k+1}(\tau) &= (k+1)H_1(\tau) - H_{k+1}(\tau) = H_1(\tau) + I_k(\tau) \\ R_{k+1}(\tau) &= I_{k+1}(\tau) - I_k(\tau) = H_1(\tau) \end{aligned}$$

In that case, this dimension is referred to as the ***embedding dimension***. A natural way to determine this embedding dimension is to evaluate the observation's joint entropy for various successive dimensions and watch its evolution as the dimension increases. The estimation of the embedding dimension is a quest for a maximum of predictability. While discussing general concerns associated with modeling (later within this chapter), we will see the notion of predictability (or at least determinism) can be approached from the point of view of conditional variance instead of entropy.

Binary tree method for entropy estimation

Having gone through the precise definition of entropy for continuous-type random variables in Chapter 2, we are now aware that its estimation from a quantized (i.e. finite resolution) observation can be tricky. Considering our quantized observation as the instances of a discrete-type random variable can mislead us to a biased measure of the system's entropy (especially for chaotic systems for which the attractor has a fractal dimension). However, in our particular context, it is very unlikely that we should encounter such dramatic behaviors and it seems reasonable to estimate the data's entropy from a histogram-like representation.

The internal representation of a multidimensional histogram of our data in a lag space can be very large and tedious to use but as an alternative, Neil Gershenfeld [Ger92] suggests the

usage of a binary tree for the estimation of the data's joint entropy in lag spaces of increasing dimension.

$$\left\{ \begin{matrix} l_1 = (7,4,5) \\ \text{i.e. } (111,100,101) \end{matrix} \right. ; \quad l_2 = (7,4,4) \text{ i.e. } (111,100,100) \; ; \quad \left. \begin{matrix} l_3 = (7,6,5) \\ \text{i.e. } (111,110,101) \end{matrix} \right\}$$

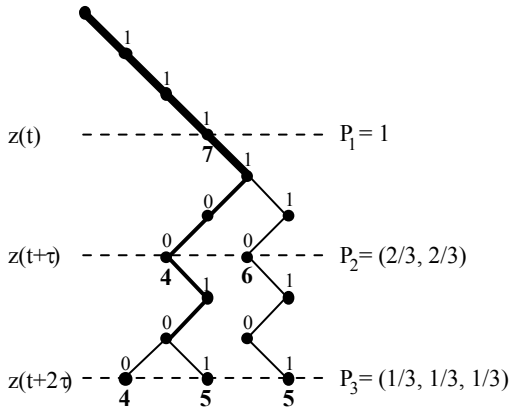


Fig. 4.2 - O(N) sorting of fixed resolution data on a binary tree [Ger92]

Much like for regular histograms, the first issue concerns the data's resolution. Deciding on a bin size for a histogram or on a binary representation for the data are essentially the same thing. Our observed and recorded data has already gone through a quantization stage, imposing an upper bound to the resolution we have access to.

Let b be the number of bits required to represent a single observation; Dmax the maximum size of the joint entropy we are trying to estimate (Dmax is also maximum dimension of the lag space); $\mathbf{l}(t) = \left(z(t), z(t+\tau), \ldots, z(t+(D\max-1)\tau)\right)^T$ the lag vector of the observation at time t.

To each lag vector **l**(t) let's associate a (b.Dmax)-long binary word w(t)=1001...110... by concatenating the binary expressions of the successive elements of the vector **l**(t). The various words w(t) can then be sorted lexicographically in a binary tree in which each node contains a counter keeping track of how many times it's been visited. A histogram-like representation of the data's joint probability for successive dimensions (from 1 to Dmax) will be provided by the appropriate level of this tree. From these joint probabilities, the associated entropy is then estimated.

$$H_d = - \sum_{\substack{\text{visited nodes i on} \\ \text{level } (b.d) \text{ of the tree}}} p_i . \ln(p_i) \text{ , where } p_i = \frac{\text{counter}_i}{\sum_{\substack{\text{all nodes j on the} \\ \text{same level than node i}}} \text{counter}_j}$$

The previous figure illustrates this process with b=3, Dmax=3 and three arbitrary lag vectors l_1, l_2 and l_3.

Computationally, in addition to providing an O(N) sorting procedure for the data, this approach also implies that memory is allocated only for occupied nodes, leading to a much more conservative storage requirement than a brute-force histogram.

Example

As an example, let's consider a sampled audio recording of a bowed violin string. We restricted our attention to the quasi-periodic part of this recording, which we normalized between -1.0 and 1.0. The following figure shows a plot of this data in a three-dimensional lag space as well as an estimate of its spectrum.

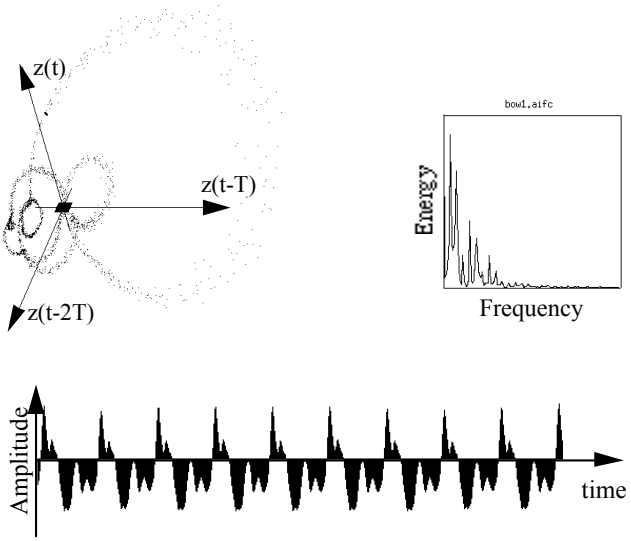


Fig. 4.3 - Lag space plot, spectrogram and wave form of a quasi-stationary chunk of a violin recording.

Applying the binary tree method that we discussed previously to this audio data with a 8 bits of resolution, we obtain the following plots an estimating of the data's joint entropy and redundancy.

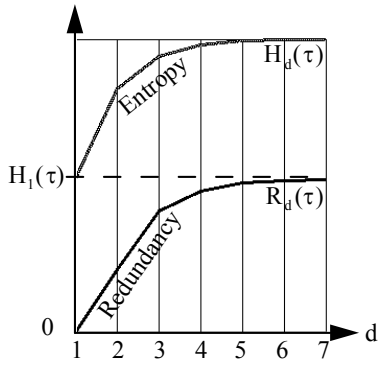
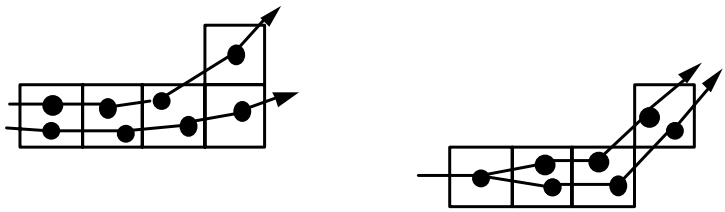


Fig. 4.4 - Joint entropy and redundancy estimations versus dimension.

Regardless of the accuracy of the previous measurement, a first qualitative observation already leads to an interesting and disturbing result. The obvious saturation of redundancy past a lag space dimension of 4 or 5 implies that the observed chunk of data doesn't have more than 4 degrees of freedom. Looking back at the data's spectrogram, one can count at least 12 or 13 harmonics. We recall that a linear system of dimension 4 can not have more than 2 peaks in its spectrum. It turns out that, indeed, through a cluster-based method that we will introduce later on in Chapter 6, we were able to infer a non-linear model of dimension 4, that reconstructs that data accurately.

Resolution

An evaluation of entropy based on an estimated parametric form for the data's probability mass function might be more reliable but it is important to keep in mind the fact that the finite resolution of our observation can mislead us anyway. The following figure is an attempt to illustrate two cases where such a phenomenon happens. The points linked by a continuous line represent the "real" infinite resolution trajectory of the observation in a lag space whereas the boxes are the quantized version of the same trajectory.



(a) (b)

Case (a): We are locally observing two separate trajectories that don't cross. The system could very well be deterministic but if we only have access to the quantized observation, it will appear that the trajectory splits, which could mislead us to believe that the system displays some random behavior.

Case (b): The trajectory really splits but this split can not be detected through the low resolution of our observation. We might end up concluding that the system is deterministic whereas it's not.

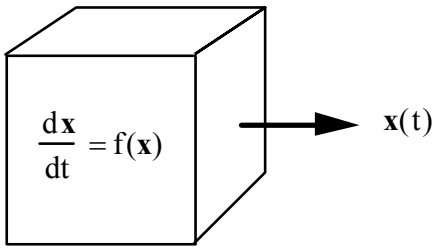
The evolution (or should we say growth) of entropy with an increasing resolution is related to the ***attractor's dimension***. Indeed, if d is big enough (i.e. at least the embedding dimension) then the object

$$\lim_{N\to\infty} \frac{\sum_{x_i} p_d(x_i).\ln(p_d(x_i))}{-\ln(N)} = \lim_{N\to\infty} \frac{H_d(\tau,N)}{\ln(N)}$$

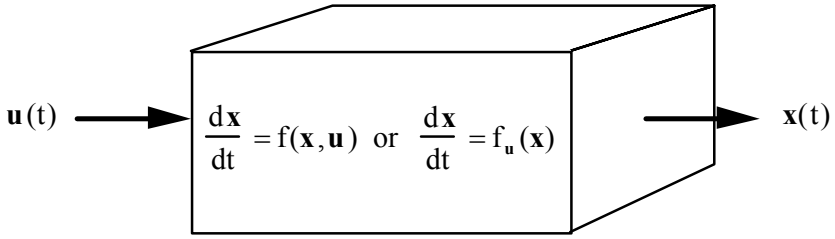
will measure the dimension of the set of our system's solution (i.e. the dimension of the set corresponds to the attractor's dimension if the system were deterministic).

Autonomous / Non-autonomous systems

In all that precedes, we've implied that the observed system had the following general form:



This expression implies that the observed system is autonomous; the evolution of the state x is only a function of this same state. Yet in most cases, we will be called to consider non-autonomous systems. Such systems have inputs **u** and it would sound more general to write the dynamics of a system as:



As illustrated by the previous expressions, there are basically two schools of thoughts for dealing with non-autonomous systems. The first is to consider a composite lag space of input and output (i.e. inputs are just extra observations). The other school could be qualified as parametric embedding (i.e. the input conditions the dynamics of the system). Most of what follows can be generalized to either one of these points of view so we will stick to our original notations, keeping in mind that the observed system doesn't need to be autonomous.

Data Characterization and Modeling Space Evaluation

The application of the Embedding theorem provided us with the confidence that we can reconstruct from observations a space which exhibits dynamics that are very closely related to the physical mechanism of the original system that we observed. We shall now look into ways to characterize the system in this reconstructed (modeling) space.

Local linear modeling as an evaluation scheme

Local linear modeling is a computationally expensive approach to the characterization of the function f(). However, this method was suggested as a means to evaluate the validity of a deterministic approach for a given system. A precise description of its implementation is provided by Tim Sauer in [GW93] p.175 -194. In a lag space of sufficient dimension d, let's consider instantaneous (d-1)-dimensional state $\mathbf{l}^*(t) = (z(t), z(t+\tau), \ldots, z(t+(d-2)\tau))^T$. The basic idea behind local linear modeling is to predict the value $z(t+(d-1)\tau)$ by identifying an appropriate set of neighbors, regressing a linear model over these neighbors, and using the regressed model in order to predict this value.

The following is a slightly more detailed explanation of this process as it is suggested by Tim Sauer:

(i) Identify some of $\mathbf{l}^*(t)$'s nearest (d-1)-dimensional neighbors $(\mathbf{n}^*_1, \ldots, \mathbf{n}^*_k)$ among the training data and compute the associated d-dimensional center of mass $\mathbf{c} = \frac{1}{k}\sum_{i=1}^{k} \mathbf{n}_i$.

(ii) Regress a linear model from $(\mathbf{n}_1, \ldots, \mathbf{n}_k)$ (and $\mathbf{c}$). Tim Sauer suggests the usage of singular-value decomposition in order to identify the smallest dimension for that linear model over this restricted neighborhood. Such a decomposition should lead to the identification of a basis for that subspace.

(iii) Use the projection of $\mathbf{l}^*(t)$ on that subspace in order to predict the future state. This prediction will lead to a d-dimensional lag vector $\mathbf{l}(t)$ which will be a linear combination of the neighbors $(\mathbf{n}_1, \ldots, \mathbf{n}_k)$.

If k is the number of neighbors used to construct the local linear model around each possible lag vector of dimension d, the evolution of the performance (for prediction) of the local linear model with the number k can reveal important information about the system. When k is the smallest (i.e. k=1), the model is a lookup of the closest neighbor in lag space. When k is very large (i.e. k=number of observation), the model is a global linear model (auto regressive). The plot of this evolution for different dimension was introduced by Casdagli in 1991 in terms of "deterministic vs. stochastic" plot (DVS).

If we restrict this study to a given dimension d for our lag space, the following figure illustrates what these plots might look like for three different systems. This figure is only an illustration, it is not the result of a particular analysis.

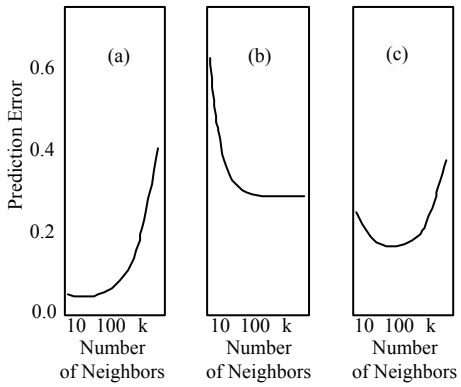


Fig. 4.5 - Example of DVS plots for three systems.

Case (a): The prediction error reaches a low minimum for the dimension d with which the plot was made, which tells us that the system is fairly deterministic. The error increases with k and that means that the system in non-linear.

Case (b): The failure of a local linear model with a small k can be seen as overfitting. The flat part for large k tells us that the system is fairly linear. The globally poor performance tells us that the system is most probably stochastic.

Case (c): This case falls between the two previous. It could be a higher dimension non-linear deterministic system or a non-linear stochastic system.

This example illustrates the eventual ambiguity between stochasticity and high dimension non-linearity of these plots. As we can recall, linear system theory could not differentiate between stochasticity and non-linearities (even for low dimension system) so this new ambiguity is still an improvement.

Probability Mass Function (PMF) estimation

Rather than interpreting the success or failure of a particular modeling approach like local linear models, one might consider estimating the data's probability mass function in a lag space. In addition to providing valuable information concerning the data and the modeling space, such an estimation could be used as our final model for the system. Furthermore, the stochastic nature of a probability distribution relieves us from making any a priori guess concerning the predictability of the system.

Given a sampled instance of a stationary ergodic stochastic process **x** and a dimension d, the problem here is to estimate a parametric form of the PMF $p_d(\mathbf{x})$. Natural objects to think of are a histogram and Parzen windows. A histogram counts occurrences of particular values, leading to a sparse estimate of the data's probability distribution. Overcoming this sparse property is the goal of a Parzen window, and it will do so by smoothing the histogram. Here, "smoothing" can be taken quiet literally as in most cases, the application of such a window is rigorously equivalent to the application of a non-causal finite impulse response filter. We recall that given

a kernel (or Parzen window) $\gamma(\mathbf{z})$ and a number N of observations $\mathbf{x}^{(i)}$, this approach will lead to the following estimate for the data's probability mass function:

$$\tilde{p}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}\gamma\left(\mathbf{x}-\mathbf{x}^{(i)}\right)$$

In order for this estimate to be a valid probability mass function, it is implied of course that the kernel $\gamma(\mathbf{z})$ should integrate to one. If we started with a histogram of "infinite resolution," then we could interpret the previous expression as a convolution product between the kernel and our histogram. This is to say that the outcome can be seen as a filtered version of an infinite resolution histogram; the filter (for which $\gamma(\mathbf{z})$ stands for the impulse response) is the means by which we generalize the estimated probability density to areas where no data was observed.

Yet, as the dimension d increases, the internal representation of the histogram with a decent resolution requires an exploding amount of memory (L^d where L is the number of bins given by our resolution). Therefore a straightforward application of Parzen's idea becomes quickly unfeasible. Kris Popat and Rosalind Picard [PP93] adapted Parzen's technique *"by replacing the original data with a smaller set of representative points and by adapting the sizes and shapes of the kernel to match the statistics of the regions they represent"*. These most representative points are chosen by a standard clustering algorithm. The general form for the resulting estimation of the probability distribution is almost the same as for the Parzen technique with the expectation that the sum is over the number M of identified clusters instead of the number of data points and $\mathbf{c}^{(i)}$ the centroids of these clusters.

$$\tilde{p}(\mathbf{x}) = \frac{1}{M}\sum_{i=1}^{M}\gamma\left(\mathbf{x}-\mathbf{c}^{(i)}\right)$$

In addition to this dramatic reduction of our problem's size, the kernels they propose are separable probability density functions, which allows a recursive estimation of the conditional probabilities.

Without any deep structural information within each identified cluster, a possible form for the estimate of the data's probability mass function could be the following:

$$p\left(x_n, x_{n-1}, .., x_{n-d+1}\right) = \sum_{m=1}^{M} w_m \cdot \prod_{k=0}^{d-1} K_{m,k} \cdot e^{-(x_{n-k}-\mu_{m,k})^2/(2\sigma_{m,k}^2)}$$

where M is the number of *representative* points, $w_m>0$, and $\sum_{m=1}^{M} w_m = 1$.

Of course, the approach of cluster-based PMF estimation is not limited to this specific choice for the kernels. We will come back to this approach later on in this document and provide a few examples and alternative choices for the form of the PMF that could result from such a process.

Assuming that we now possess a fair parametric estimate of our observation's joint probability distribution for different dimensions, the question is what we'll use it for. The estimated PMF is an elegant summary of the original data set and the first type of use one can think of is data characterization. We could use it to evaluate entropies for the successive dimensions in the hope of finding the system's embedding dimension.

Once this embedding dimension is detected (or estimated) we could then consider that the system is deterministic and forget about these joint probabilities when we model it. This would be a ***deterministic approach***. The other option is to use our estimates of these joint probabilities as a model of the system. Unless the parametric form of these probabilities have some Dirac distribution terms (which is very unlikely as the method is a "smoothing" of the original histogram) the model they will describe will carry some uncertainty. In other words, the system would be modeled as a random number generator which probability distribution is derived from the conditional probability distributions of the data. This would be a ***stochastic approach***.

Deterministic approach

As we pointed out earlier, this approach assumes the existence of an embedding dimension. Having gone several times through the difficulties involved in entropy estimations, we know that the existence of such an object might not appear from our data as clearly as we'd like. This dimension d will be the result of a fairly arbitrary decision rather than an unquestionable observation. This assumption can be stated as:

$$p\left(x_n \middle| x_{n-1}, x_{n-2}, .., x_{n-d}\right) = \delta\left(x_n - f(x_{n-1}, x_{n-2}, .., x_{n-d})\right)$$

$$\text{or simply } x_n = f(x_{n-1}, x_{n-2}, .., x_{n-d})$$

Our system is therefore entirely characterized by a set of d initial conditions and a representation of the function f(). In order for our model to generalize the behavior of our system with variations on the initial conditions for instance, the representation of f() should be defined on a wider set than the training data (i.e. our observation). Given also that we want to avoid a prohibitive size of this representation, the goal of this training should be to estimate a parametric form for f().

For this purpose, there are two basic sets of approaches, the global and the local approaches. A global approach assumes some fixed architecture for a closed form of the function f() on the entire set on which it's defined, and tunes the parameters of this architecture in order to fit the training data by minimizing some criteria. An example would be to fit a polynomial of dimension d and fixed order N to the observation with a least mean square criteria. A local approach will typically use the training data as the model and an interpolating method as the means to generalize it. Local linear modeling is an example of a local approach. These approaches are not mutually exclusive as one can choose to take a local approach on a representative subset of the training data. Each one of these representative points carries some information about the system's behavior in the corresponding neighborhood to the rest of the model. These points are sometimes referred to as the anchor points of radial basis functions. If the number of anchor points is fixed, than there is an assumption concerning the closed form of the model even though it is based on interpolations between observed data. We could qualify this method as being "glocal" if we felt like inventing a word.

A local approach usually gives better performance as it doesn't constrain the model as much as a global method. Yet, the representation it provides is just as big as the training data, which makes it heavy and rigid. By rigidity we mean here that the model often lacks a restricted number of knobs allowing mutations of the system. In that respect, global models are preferable because of their smaller size, their usually smaller computation requirements, and their limited fixed number of parameters (knobs).

Stochastic approach

Supposing that we have an estimation of the data's probability mass function in a (d+1)-dimensional lag space, deriving conditional expectations from that estimate would be a way to express a d-dimensional deterministic model. However, we might decide to keep the estimated PMF as our model itself. Instead of taking expectations, deriving conditional probability distributions is a way to express a stochastic model for our system. Given some current state (in lag space) for our system, the next predicted state becomes the instance of a random variable which behaves accordingly to these conditional probability distributions.

Given the values ($X_{n-1},..,X_{n-d}$) of the last d lag values, the prediction of the next lag value will be the output of a random number generator whose PDF matches the following conditional PDF:

$$p(x_n \mid x_{n-1},..,x_{n-d}) = \frac{p(x_n, x_{n-1},..,x_{n-d})}{\int_{X_n} p(X_n, x_{n-1},..,x_{n-d})\,dX_n}$$

Note: We recall here how easily one can create instances of an arbitrary PDF random variable from the instances of a uniformly distributed random variable. Let's consider two random variables x and y related to each other by x=g(y) where g() is a diffeomorphic function g:R->[0,1]. Let's suppose also that x is uniformly distributed on [0,1]. The relationship $p_X(X)dX=p_Y(Y)dY$ gives us:

$$\forall X \in [0,1],\ \frac{dy}{dx} = \frac{d}{dx}\left(g^{-1}(X)\right) = \frac{1}{p_y(Y)}$$
$$= \frac{1}{g'\left(g^{-1}(X)\right)} = \frac{1}{p_y(Y)}$$
$$= \frac{1}{g'(Y)} = \frac{1}{p_y(Y)}$$

and so $\forall Y \in R,\ p_y(Y) = g'(Y)$ i.e. $P_y(Y) = g(Y)$ (the cumulative function of y).

This tells us that if we possess a typical random number generator providing us with instances X (in [0,1]) of x, we can create an instance Y of y (with arbitrary PDF $p_Y(Y)$) by applying the simple mapping :

$$Y = P_y^{-1}(X)$$

It is important to note that such an approach to the system might not be an improvement over the deterministic approach. As an illustration, let's see what happens when the observation is the output of a simple 2D deterministic system. Let's even chose that system to be linear, namely:

$$x_n = (2\cos\theta)x_{n-1} - \lambda . x_{n-2} \quad \text{(where } \lambda \in]0,1[)$$

(x_n is a damped sine wave)

If we decide to estimate the probability distribution of this variable based on some cluster analysis like we suggested earlier and decide not to use any specific structural information (such as local linear or others) within each cluster, then we will end up with a limited number of clusters (or zones) over which the data will be summarized via some averaging. The next figure illustrates the estimate of the conditional probability distribution of x_n given x_{n-1}=v and x_{n-2}=u.

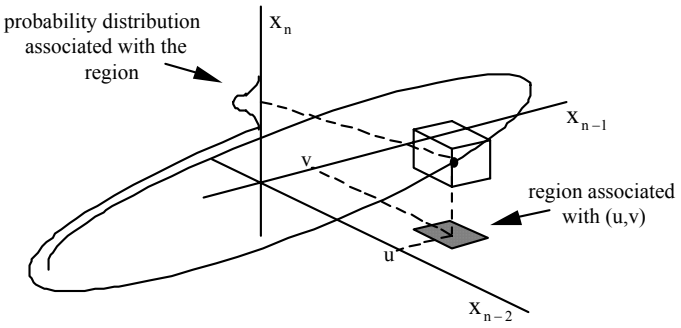


Fig.4.6 - Illustration of the "pessimism" of the PMF approach for modeling in the context of a damped sine-wave.

In the previous figure, the box is a representation of the spatial zone associated with a specific cluster. Chances are that given the finite number of "representative points" used for the estimation of the PMF, the variance of the estimate of this conditional probability distribution will not be zero (as it should be). The resulting conditional variance of x_n given (x_{n-2},x_{n-1}) is the artifact of a model mismatch as local regions are summarized by a single scalar (conditional mean) instead of capturing the linear structure of the system. This is why the figure is called "pessimism" of the PMF approach. The next figure illustrates the same point in terms of the prediction surface itself.

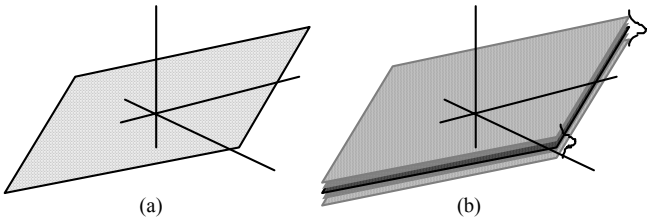


Fig.4.7 - Stochastic model of a deterministic linear 2D system.

(a) represents the true deterministic system, a simple plane, whereas (b) represents the model induced by the PMF approach, a fuzzy plane. Again, the predictor's fuzziness is only an artifact of a model mismatch. Instead of characterizing the behavior of the observed system, it is a function of the size of the clusters that were identified.

General Concerns

In the following, we will discuss some of the major issues and concerns that one encounters when attempting to build a non-linear model from the observation of an arbitrary system. The author doesn't pretend to be exhaustive at this point as most of the following concerns point rapidly to very involved material which we might not need to worry about in our restricted context. We shall simply take a quick overview of these concerns to develop an intuitive understanding and a general awareness.

Predictability versus Determinism

With the access to powerful computers, the study of non-linear dynamics has captivated the attention of an increasing number of scientists from various fields. Along with this new interest appeared a new set of notions and concerns; some of which are useful and others which are plain frustrating.

In 1963, Lorenz was the first to experiment with a simple non-linear system which, although it was deterministic, would never settle down to equilibrium or to a period state. This led to the definition of chaos which, in addition to captivating science fiction writers, questioned the relationship between the notions of determinism and predictability. The system being deterministic implies that there is a strict relationship between its past and its future; no randomness or ambiguity occurs concerning the state that follows the present state. In the case of a deterministic but chaotic system, tiny variations applied to the initial conditions result rapidly in dramatically different behaviors in spite of this non-ambiguous causality. From an experimental point of view, this means that the system is inherently unpredictable because no measurement of a system's current state can pretend to be error-free. This is not to say that no useful information can be extracted from the observation of a chaotic system. Short-term predictions could still be fairly accurate if its chaotic behavior is not too dramatic and a global analysis of its behavior can lead to a good understanding of its mechanism. Indeed, chaotic systems are not deprived from a structure. The set of states that a chaotic system visits (in its own state space or another embedding) turns out to be a fractal.

The present work is fairly free with the intuitive assimilation of determinism and predictability. This is mainly due to the fact that in the context of the systems that one encounters with musical instruments, chaotic behaviors are very unlikely.

Entropy versus Variance

In the light of the previous remark, we are now aware that the sentence *"The estimation of the embedding dimension is a quest for a maximum of predictability"* in our earlier introduction of the embedding dimension, should be taken with caution. All we meant to express is the desire to find the smallest dimension with which the system can be modeled as a deterministic system with a conservative out-of-sample error. We also related the search of the embedding dimension to some maximization of joint entropy.

Let's consider the stochastic approach which we introduced earlier. Keeping the model in the form of a conditional probability function can be seen as describing an ambiguous prediction function via a "fuzzy" hyper-surface. Intuitively, the "skin depth" of this fuzzy surface is related to the ambiguity (or predictability or deterministic property) of the model. This visualization of "ambiguity" seems to be pointing more towards a conditional variance than it does towards entropy. Instead of staying confused, let's work out the relationship between variance and entropy in the Gaussian case and realize that these two points of view are not as different as they might sound.

General relationship in a Gaussian case

Let **x** and **y** be two jointly Gaussian random vectors. It is a well known fact that under these circumstances, **x**, **y** and **x**|**y** are gaussian random vectors as well. Through substitution and identification, we get:

$$\mathrm{E}\left[\mathbf{x} \middle| \mathbf{y} = \mathbf{Y}\right] = \bar{\mathbf{x}} + \Lambda_{\mathbf{xy}}\Lambda_{\mathbf{y}}^{-1}(\mathbf{Y} - \bar{\mathbf{y}}) \text{ and } \Lambda_{\mathbf{x}|\mathbf{y}} = \Lambda_{\mathbf{x}} - \Lambda_{\mathbf{xy}}\Lambda_{\mathbf{y}}^{-1}\Lambda_{\mathbf{xy}}^{\mathrm{T}},$$

where $\Lambda_{\mathbf{x}}$, $\Lambda_{\mathbf{y}}$, $\Lambda_{\mathbf{xy}}$ and $\Lambda_{\mathbf{x}|\mathbf{y}}$ stand respectively **x** and **y**'s covariance matrices, joint and conditional covariance matrices. If N is the dimension of the Gaussian random vector **x**, then we have the following expression for the conditional probability distribution:

$$p_{\mathbf{x}|\mathbf{y}}(\mathbf{X}|\mathbf{Y}) = \frac{1}{(2\pi)^{N/2}\left|\Lambda_{\mathbf{x}|\mathbf{y}}\right|^{1/2}} \exp\left(-\frac{\left(\mathbf{X} - \mathrm{E}\left[\mathbf{x}\middle|\mathbf{y} = \mathbf{Y}\right]\right)^{\mathrm{T}} \Lambda_{\mathbf{x}|\mathbf{y}}^{-1} \left(\mathbf{X} - \mathrm{E}\left[\mathbf{x}\middle|\mathbf{y} = \mathbf{Y}\right]\right)}{2}\right)$$

We recall (Chapter 2) that the entropy of a **continuous-type** random variable (or vector) is defined as the expectation of the natural log of its probability distribution:

$$H_c(\mathbf{u}) = -\int_{\mathbf{U}} p_{\mathbf{u}}(\mathbf{U}) \ln\left(p_{\mathbf{u}}(\mathbf{U})\right) d\mathbf{U} = \mathrm{E}\left[-\ln\left(p_{\mathbf{u}}(\mathbf{u})\right)\right],$$

and we also recall that there is no elegant continuity between the definition of entropy for discrete-type and continuous-type random variable without the introduction of an extra term taking quantization (or resolution) in account:

$$H_c(\mathbf{u}) = \lim_{\delta \to 0} \left[H_d(\mathbf{u}_\delta) + \ln \delta \right]$$

Implied by what precedes, we'll never have access to anything other than an estimate of the "discrete-type version" of entropy so when we refer to entropy in our context, we refer to H_d and not H_c. This point being clarified, let's find an expression for the conditional entropy of $\mathbf{x}|\mathbf{y}$ and relate it to the corresponding conditional covariance matrix:

$$H_d(\mathbf{x}|\mathbf{y}) = H_c(\mathbf{x}|\mathbf{y}) - \ln \delta = E\left[-\ln\left(p_{\mathbf{x}|\mathbf{y}}(\mathbf{x}|\mathbf{y}) \right) \right] - \ln \delta$$

which, given the form of this conditional probability, leads to:

$$H_d(\mathbf{x}|\mathbf{y}) = \frac{N}{2} \ln(2\pi) + \frac{1}{2} \ln\left| \Lambda_{\mathbf{x}|\mathbf{y}} \right| - \ln \delta + \frac{1}{2} E\left[(\mathbf{x} - E[\mathbf{x}|\mathbf{y}])^T \, \Lambda_{\mathbf{x}|\mathbf{y}}^{-1} \, (\mathbf{x} - E[\mathbf{x}|\mathbf{y}]) \right]$$

Let's define the following temporarily for notation simplification purposes:

$$\mathbf{a} = (a_i)_i = (\mathbf{x} - E[\mathbf{x}|\mathbf{y}]); \; \Lambda_{\mathbf{x}|\mathbf{y}}^{-1} = (g_{i,j})_{i,j} \; ; \; \Lambda_{\mathbf{x}|\mathbf{y}} = (l_{i,j})_{i,j}$$

Then

$$E\left[(\mathbf{x} - E[\mathbf{x}|\mathbf{y}])^T . \Lambda_{\mathbf{x}|\mathbf{y}}^{-1} . (\mathbf{x} - E[\mathbf{x}|\mathbf{y}]) \right] = E\left[\sum_{i=1}^{N} a_i \sum_{j=1}^{N} g_{i,j} a_j \right] = \sum_{i=1}^{N} \sum_{j=1}^{N} g_{i,j} E[a_j a_i]$$

but of course, $E[a_j a_i] = l_{j,i}$ by definition of the covariance matrix and therefore:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} g_{i,j} E[a_j a_i] = \sum_{i=1}^{N} \sum_{j=1}^{N} g_{i,j} l_{j,i} = \mathrm{Tr}\left[\Lambda_{\mathbf{x}|\mathbf{y}}^{-1} \Lambda_{\mathbf{x}|\mathbf{y}} \right] = N \, ,$$

and we end up with:

$$H_d(\mathbf{x}|\mathbf{y}) = \frac{N}{2} \ln(2\pi) + \frac{1}{2} \ln\left| \Lambda_{\mathbf{x}|\mathbf{y}} \right| - \ln \delta + \frac{N}{2}$$

The case of lag spaces

In the context of the estimation of a prediction surface in an embedding, x will stand for the next sample that we wish to predict. It will most likely be a scalar and therefore, N will be equal to 1 and the covariance matrix will reduce to a scalar variance. The previous relationship will then reduce to the following:

$$H_d(x|\mathbf{y}) = \frac{1 + \ln\left(2\pi\sigma^2_{x|\mathbf{y}} / \delta\right)}{2}$$

Although this expression was worked out from the assumption of Gaussian distributions (which is often justified), it suffices to illustrate a striking relationship between conditional entropy and conditional variance.

Data resolution and other artifacts of measurements can lead to serious difficulties in the estimation of entropy. Hence, estimating the data's probability distribution and building a model in the form of a conditional probability distribution appear as a more robust technique to get similar information about a system's predictability in terms of conditional variances. In Chapter 6 of this document, we'll discuss *Cluster-Weighted Modeling*, a novel approach to the estimation of such probability distributions.

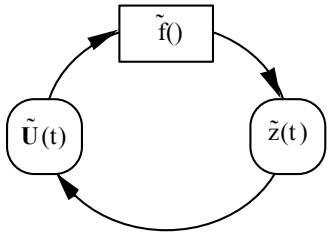
Stability / Non-locality

Non-locality and stability issues are inherent to the modeling of a prediction function, whether it is linear or not. Let z(t) be the time series we wish to model and the vector **U**(t) be lag vectors of z(t) with appropriate dimension (embedding dimension). We wish to model our time series via a prediction function which maps **U**(t) to z(t). Let's suppose we estimate a form for this prediction surface from some observed data and let's suppose that we achieve a very acceptable accuracy in terms of out-of-sample prediction:

$$\left|z(t) - \tilde{f}(\mathbf{U}(t))\right| < \varepsilon,$$

where $\tilde{f}(\mathbf{U}(t))$ stands for our estimated prediction surface applied to the input **U**(t).

As soon as we feed the estimation of a new sample back into the prediction function in order to predict a large amount of successive samples, the once acceptable prediction error can propagate in a dramatic way, eventually leading to instabilities for the system.



Even if the estimated model doesn't exhibit instabilities, it can't be relied upon in terms of the error $|z(t) - \tilde{z}(t)|$ or $\left|\mathbf{U}(t) - \tilde{\mathbf{U}}(t)\right|$ as t increases. This is what we refer to as ***non-locality***.

In the case of a linear model, the prediction function can be written as the scalar product of two vectors, or even more generally with a matrix H such that $\tilde{\mathbf{U}}(t+1) = \mathbf{H}.\tilde{\mathbf{U}}(t)$ in the

case of discrete time or $d\tilde{\mathbf{U}}(t)/dt = \mathbf{H}.\tilde{\mathbf{U}}(t)$ in the continuous case. The largest eigen value of this matrix will provide an upper bound for the rate at which an initial error $\varepsilon_0 = \left|\mathbf{U}(0) - \tilde{\mathbf{U}}(0)\right|$ may propagate through this recursion.

In the case of non-linear models, the best we can do is not all that different. A local linearization around any relevant point on the prediction surface will lead to a similar (but local) matrix form. Averaging sorted eigenvalues of these local matrices along the relevant points of the prediction surface (i.e. observed data) leads to the definition of the Lyapunov Exponents. For any further discussion about these exponents, the reader should consult literature on non-linear dynamics. The author suggests Steven Strogatz' text book *"Non-linear Dynamics and Chaos"* [Str94].

As for stability, while heavenly properties of a linear model could turn the study of stability into the geometrical distribution of a rational function's poles, non-linear systems are out of luck. A non-linear system can be forced not to diverge as one can constrain it with conditions that satisfy stability but to this date, there is no general necessary and sufficient condition for the stability of a non-linear system. Furthermore, unlike for linear-systems, the stability of a non-linear system can depend upon its initial conditions.

Generalization

The data from which a prediction function may be estimated will always be finite and sparse while the support of the estimated prediction function will eventually be continuous. This is to say that our model will implicitly ***generalize*** the observed data. Because generalization is by essence not dictated by the training data, it will have to be based upon convictions and common sense. This generalization will be implied by the chosen architecture of our model and we will encounter this issue several times in the following chapters.

Chapter Summary

Applying Floris Takens' embedding theorem to the relationship between the state space of a dynamical system and a lag space reconstructed from the time-series of an observation provides a general and solid ground for the inference of physically meaningful models. We've introduced some ideas which provide means by which one can analyze and characterize a dynamical system's behavior without the bias of an initial choice of architecture. Given this basis, modeling a system from observed behavior is turned into a prediction surface estimation problem which can lead to, but is not limited to, the construction of more familiar linear models. We've also introduced a few objects which are more likely to characterize the stochasticity of a system than the ones that linear system theory has to offer.

Embedding modeling is a new approach to building universal and meaningful models from an observation. Although the various notions that we discussed here are well established in the literature ([ABST93], [Bro94], [Cas92], [ER85], [Ger88], [Ger92]), such an inference of non-linear systems is still considered marginal and controversial. This approach should be

understood as sampling the physics of the system, retaining exactly the information needed to reproduce its behavior and no more. To the author's knowledge, embedding modeling has never been applied to the modeling of musical sounds, but has been confined within the area of non-linear dynamics and chaos theory. This chapter presents the convictions upon which any synthesis or modeling ideas that will follow in this document are based. The remaining questions concern the choice of a specific approach to the estimation of a prediction surface. These questions are not minor as one could argue that they constitute the heart of the modeling task; but whether we choose global, local, deterministic, stochastic, general or specific architectures for our prediction surfaces, the resulting models will exhibit a faith in their ability to capture physical behavior. This is a major step from something like the minimization of an out-of-sample prediction error.

Finally, we referred to time-series and dynamical systems throughout the entire chapter rather than sounds and musical instruments. This was a way to emphasize the generality of the suggested modeling scheme. Any physical system which can be observed is a potential subject for embedding modeling.

Chapter 5

Global Polynomial Models

To construct global non-linear prediction functions for time series, multi-dimensional polynomials appear as reasonable and general choices in the absence of any specific knowledge about the data. The use of polynomial functions in this context is not original but we present an original approach in what follows. We'll show how the estimation of a polynomial model, in spite of its non-linear nature, can be turned into a linear estimation task. More specifically, this chapter examines the use of a Kalman filter for this task, leading to a recursive estimator of non-linear models from a data stream as it is being observed. We will review the mechanism of a standard Kalman filter and evaluate the behavior of the estimated predictors.

Global polynomial models

In order to build a global parametric representation of the prediction function f() we introduced earlier, we need to state clearly what these parameters are and what the generic architecture of the model is. As we might not have access to any specific knowledge about the system that produced our data, this architecture should be as general as possible. By "general" we refer to its ability to describe any arbitrary surface. Multi-dimensional polynomials appear to be a reasonable and general enough architecture. Once we make that choice of architecture, we need an approach to estimate of this polynomial's coefficients from the observed data.

There are two obvious ways to view this problem. The first way is to try to build a basis of orthonormal polynomial functions on which we will project f() (which is sampled by our data) . As our training data for f() will obviously not span the entire d-dimensional space, the term "orthonormal" for our basis has to be taken carefully. Indeed, our basis will have to be orthonormal with respect to some measure in the d-dimensional space that will be related to the support of our training data. This approach has been taken by Reggie Brown [Bro94].

Given a particular dimension d for the lag space, Brown recursively builds an orthonormal basis of polynomial functions through a Gram-Schmidt orthonormalization process. This approach requires an estimate of the observation's probability mass function which will be used to define a scalar product for the d-dimensional lag space. Brown's approach is expectation-based, which is equivalent to taking the data's histogram as an estimate of its probability mass function.

For the same goal, we chose an effortless alternative based on the realization that this task can be stated in a linear form.

As a linear estimation problem

Another way to estimate the coefficients of this polynomial is to fit a parametric form directly to the data. This is the approach we took in this work. In order to limit the size of our problem we will make an arbitrary decision concerning the maximum order q of that polynomial function. The order q is nothing else than a fitting control parameter for our method. The smaller q is, the smoother our estimated surface will be. The ability to control the value of this parameter might be a good way to avoid overfit. Our goal is now to fit a polynomial function P() of d variables and order q to our data with respect to some criteria.

$$z_n \approx P(z_{n-1}, z_{n-2}, .., z_{n-d})$$

Let's write as ($f_{n,k}$) the set of all the possible cross-products of our d variables of order q or less:

$$f_{n,k} = (z_{n-1})^{b_{k,1}} (z_{n-2})^{b_{k,2}} \ldots (z_{n-d})^{b_{k,d}} \quad (\neq x_{n,j} \text{ if } j \neq k),$$

(where the $b_{k,l}$ are integers such that $\forall k \in \{1,..,M\}, \sum_{l=1}^{d} b_{k,l} \leq q$)

Then in terms of these cross-products $f_{n,k}$, an equivalent expression of our desired polynomial model is the following weighted sum:

$$z_n = \sum_{k=1}^{M} x_k f_{n,k} \ ,$$

And the problem of model fitting is now turned into the estimation of the coefficients x_k, which is a linear problem. With the help of this elementary writing artifact, we have essentially reduced our non-linear modeling task to a very familiar linear fit.

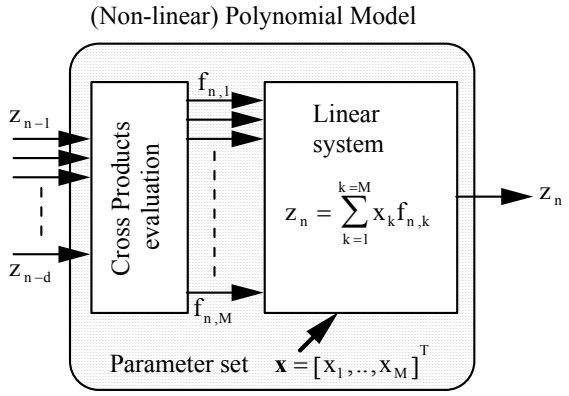


Fig. 5.1 - Reducing a polynomial model to a linear system.

Let's illustrate this simple point with an example. More specifically, let's imagine the polynomial function P() takes a single variable and is such that $P(X) = X^2 - X$.

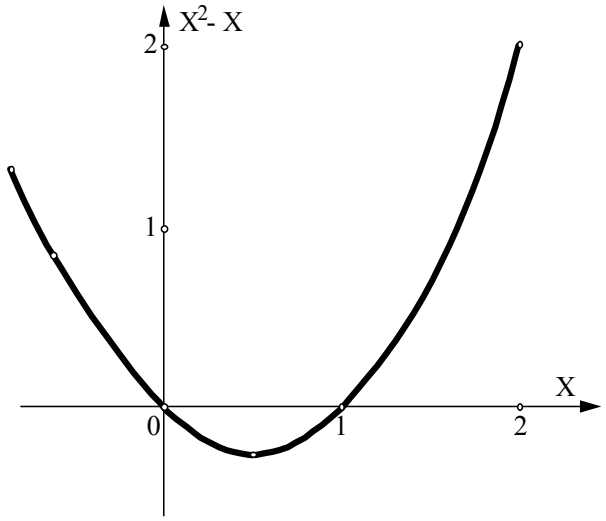


Fig. 5.2 - 2D plot of the simple polynomial $P(X) = X^2 - X$.

The following figure (Fig. 5.3) illustrates the expression of the same polynomial function as a linear function of the two variables $u = X$ and $v = X^2$.

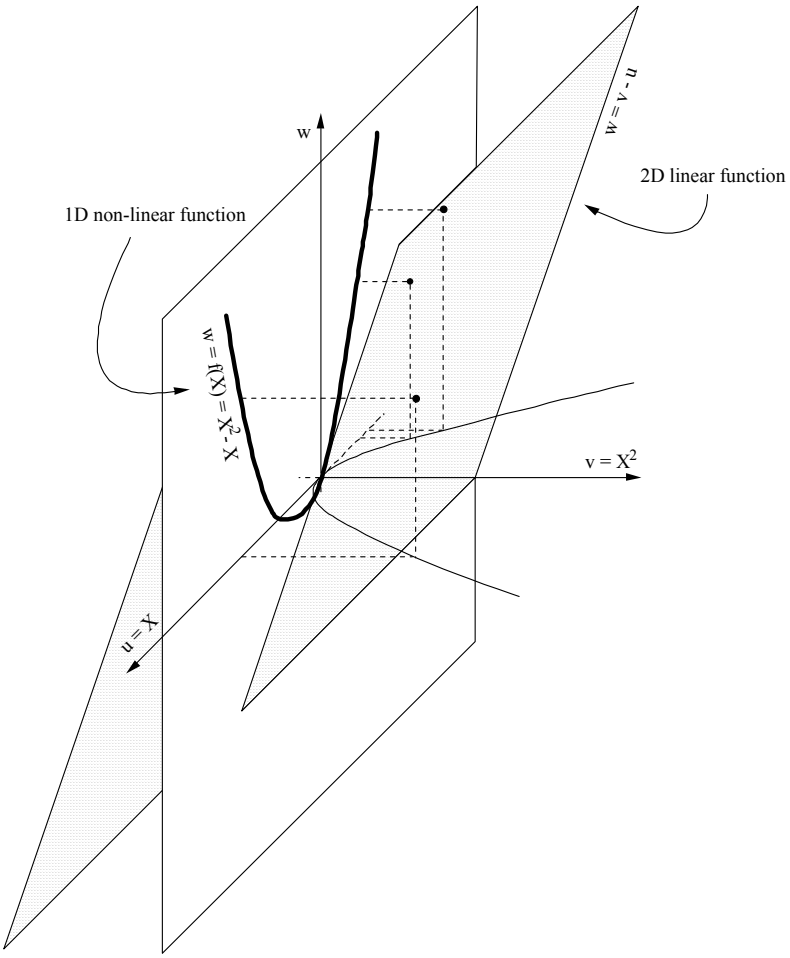


Fig. 5.3 - A 2D linear expression of a 1D non-linear function. This figure is a geometrical interpretation of the same function as linear combination of cross-products.

Let N be the number of observations we have in our training set and let's define the following objects:

$$\mathbf{A} = \begin{bmatrix} f_{d+1,1} & - & f_{d+1,M} \\ | & \dots & | \\ f_{N,1} & - & f_{N,M} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ | \\ x_M \end{bmatrix} \text{ and } \quad \mathbf{z} = \begin{bmatrix} z_{d+1} \\ | \\ z_N \end{bmatrix}$$

Our problem reduces to solving the linear equation $\mathbf{Ax}=\mathbf{z}$ for $\mathbf{x}$. Of course, chances are that the number of training data points N will be much bigger than M and therefore, this problem is ill-conditioned. At this point, one can think of pseudo-inversion and methods such as singular value decomposition. Indeed, a singular value decomposition would give us

$$\mathbf{A}_{(N-d)\times M} = \mathbf{U}_{(N-d)\times M}\, \Sigma_{M\times M}\, \mathbf{V}^{T}_{M\times M}$$

where Σ diagonal and $\mathbf{U}^{T}\mathbf{U} = \mathbf{V}\,\mathbf{V}^{T} = \mathbf{I}$

and we could then estimate **x** as follows:

$$\mathbf{x} = \mathbf{V}\,\tilde{\Sigma}^{-1}\,\mathbf{U}^{T}\,\mathbf{z} \quad , \text{ where } \left(\tilde{\Sigma}^{-1}\right)_{ii} = \begin{cases} 1/(\Sigma)_{ii} & \text{if } (\Sigma)_{ii} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

But given that N might in the order of 10,000 or more, we can foretell the potential heaviness of such an approach. These computations could be simplified by noticing that the rank of **A** can't be any bigger than M but even then the method wouldn't posses the flexibility of an adaptive algorithm (the estimation has to be done from scratch for each new set of observations). Instead, we will introduce a recursive method, namely a Kalman filter, that will solve our problem as we acquire new data.

Cross-products

Given a number of variables d and an order q, it might sound useful to compute the number M(d,q) of terms $(f_{n,1},...,f_{n,M})$ corresponding to the list of possible cross-products of order q or less. Having an idea concerning the number of these terms will tell us how the size of our problem grows with the dimension d and the order q.

We wish to count all the possible $z_1^{b_1}.z_2^{b_2}\ldots z_d^{b_d}$ such that $\forall k,\ b_k \in N$ and $\sum_{k=1}^{k=d} b_k \leq q$. This is equivalent to counting all the possible cross-products $1^{b_0-1} z_1^{b_1-1}.z_2^{b_2-1}\ldots z_d^{b_d-1}$ such that:

$$\forall k,\ b_k \in N^{*} \text{ and } \sum_{k=0}^{k=d}(b_k - 1) = q \quad \left(\text{i.e.} \sum_{k=0}^{k=d} b_k = q + d + 1\right)$$

At this point, we can recall that there are $\binom{n-1}{k-1}$ ways to choose k non-zero positive integers that sum to n. Therefore a simple expression for M(d,q) is the following:

$$M(d,q) = \binom{q+d}{d}$$

Pascal's famous triangle, based on the property: $\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}$, leads to:

$$M(d+1,q) + M(d,q+1) = M(d+1,q+1)\ ,$$

which allows us to build a table for M(d,q) very quickly.

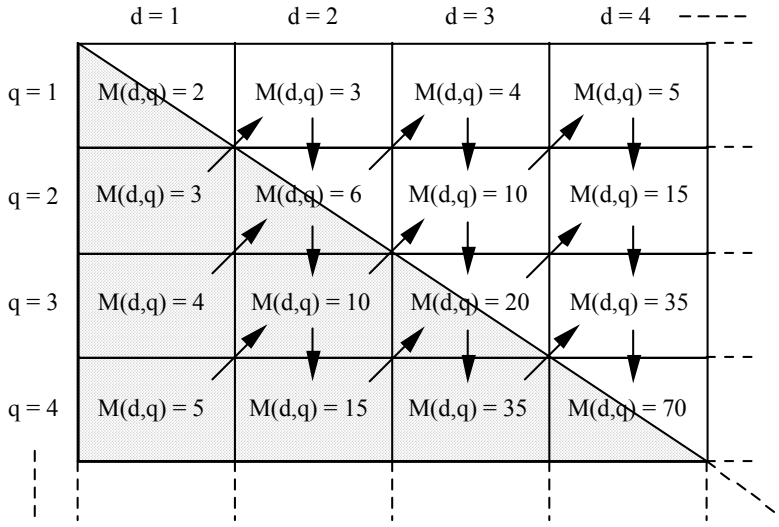


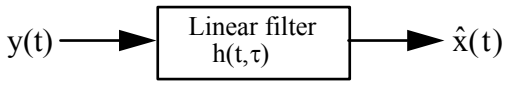
Fig 5.4 - M(d,q) table is Pascal's triangle.

Recursive estimation

Without pretending to be exhaustive, we will hereby introduce a standard Kalman filter by first tracing it back to its origins in detection and estimation theory and stating clearly the problem it solves. We will then work our way rapidly through its mechanism.

The origins

One of the original concerns of estimation theory was the estimation of the realizations of a stochastic process x(t) based on the observation (for $T_i<t<T_f$) of another related process y(t). In that context, and because of a lack of tools for non-linear systems, the best answer to that problem was a linear least square estimate which should lead to the description of a linear filter (impulse response $h(t,\tau)$):



The impulse response $h(t,\tau)$ is chosen such that the error is orthogonal to the observation:

$$E[(x(t)-\hat{x}(t))\,y(\tau)]=0$$ (Where E[.] refers to the expectation)

Rewriting this condition with covariances ($K_{xy}(t,\tau)$ and $K_{yy}(t,\tau)$) and the impulse response of the filter ($h(t,\tau)$) leads to the very famous *Wiener-Hopf equation*:

$$K_{xy}(t,\tau) = \int_{T_i}^{T_f} h(t,\sigma).K_{yy}(\sigma,\tau)d\sigma$$

Wiener gave his name to two versions of a filter that solves this equation in specific cases but Kalman suggested an alternative with an original recursive approach. The main originality of his approach was probably the introduction of a matrix (usually referred to as "**F**" - see below) which allows one to solve a wide variety of linear estimation problems. Later, extensions and modifications of this approach were used to solve non-linear estimation tasks.

The problem

Let's consider the following linear system:

$$\mathbf{x}(t+1) = \mathbf{F}\,\mathbf{x}(t) + \mathbf{v}(t) \quad (5.1) \qquad \text{(Evolution of parameters)}$$

and

$$\mathbf{z}(t) = \mathbf{H}(t)\,\mathbf{x}(t) + \mathbf{w}(t) \quad (5.2) \qquad \text{(Observation)}$$

where **F** is a pxp matrix, **H**(t) is a nxp matrix and **v**(t) and **w**(t) are white noises, usually assumed to be Gaussian. In our case, we would have:

p=M : the number of parameters "p" is the number of coefficients in the polynomial.

n=1 :our observation **z**(t) and the additive noise **w**(t) are scalars.

$\mathbf{H}(t) = \left[f_{t,1} \quad - \quad f_{t,M} \right]$: the polynomial is written as a linear combination of all the possible cross products.

F=I : the coefficients of the polynomial are constant.

We will estimate the parameters **x**() given our observations of z() through the following model:

$$\hat{\mathbf{x}}(t+1 \mid s) = \mathbf{F}\,\hat{\mathbf{x}}(t \mid s) \qquad (5.3)$$

and

$$\hat{\mathbf{z}}(t \mid s) = \mathbf{H}(t)\,\hat{\mathbf{x}}(t \mid s) \qquad (5.4)$$

(Note: $\hat{u}(t \mid s)$ refers to the estimation of $u(t)$ based on the observation of $z(\tau)$ for $\tau \leq s$)

Depending on the value of s with respect to t, solving that problem will accomplish different tasks:

- if s = t, we are filtering.
- if s < t , we are forecasting.
- if s > t, we are smoothing.

The solution that we're about to state relies on the assumption that **v**(t) and **w**(t) are centered white noises that are not correlated to **x**(t):

$$E[\mathbf{v}(t)\,\mathbf{v}^T(s)] = \delta_{ts}\,\mathbf{Q} \quad \text{and} \quad E[\mathbf{w}(t)\,\mathbf{w}^T(s)] = \delta_{ts}\,\mathbf{R}$$

(Note: δ_{ts} = 1 if t=s ; 0 otherwise)

One should keep in mind that even though it might be tempting to set **R** and **Q** to be diagonal, the only thing one we can say for sure is that **R** and **Q** will be symmetric.

As for any estimation problem, we need to state clearly what our criteria is. Let's define the error covariance matrix for t given s as:

$$\mathbf{P}(t|\,s) = E[\,(\mathbf{x}(t) - \hat{\mathbf{x}}(t|\,s))\,(\mathbf{x}(t) - \hat{\mathbf{x}}(t|\,s))^T\,]$$

Our criteria will be to minimize the trace of this matrix. One can notice that this minimization is equivalent to a least mean square criteria applied to the parameter set **x**().

The solution

The solution is known as a Kalman filter:

$$\hat{\mathbf{x}}(t|\,t) = \hat{\mathbf{x}}(t|\,t-1) + \mathbf{L}(t)\,(\mathbf{z}(t) - \hat{\mathbf{z}}(t|\,t-1)) \qquad (5.5)$$

It was rigorously demonstrated that this system was optimal in the case where the different white noises introduced earlier were Gaussian. It is not our intention in this paper to present the demonstration of this result but we can provide a few hints that should at least provide a intuitive understanding for the relation (5.5).

Hints:

We're trying to solve the Bayes least square estimate in the Gaussian case (which is equivalent to the linear least square estimate). Furthermore our system is linear:

$$\Rightarrow \quad \hat{\underline{X}}_L(\underline{Z}) = \underline{m}_x + \Lambda_{xz}\Lambda_z^{-1}(\underline{Z} - \underline{m}_z)$$

Let's write : $\hat{\underline{X}}_L\left(\begin{bmatrix} z_1 \\ | \\ z_M \end{bmatrix}\right) \stackrel{\Delta}{=} \hat{\underline{X}}(M)$

So rather than inverting the growing MxM matrix each time we get a new observation, we can use Gram-Schmidt to find a recursive relationship for the estimate:

$$\Lambda_z = \Gamma.\Lambda_e.\Gamma^T \quad \text{i.e.} \quad \begin{cases} e_1 = z_1 \\ e_2 = z_2 - \gamma_{21} e_1 \quad (\gamma_{21} = \dfrac{\langle z_2, e_1 \rangle}{\langle e_1, e_1 \rangle}) \\ | \\ e_M = z_M - \hat{z}(M|M-1) \end{cases}$$

which will end up giving us a recursive relationship on the estimate:

$$\hat{\underline{X}}(i+1) = \hat{\underline{X}}(i) + \Lambda_{x e_{i+1}} \lambda_{e_{i+1}}^{-1} e_{i+1} \ldots$$

which justifies the form of our solution in equation (5.5).

It will be the task of **L**(t) to make sure the trace of **P**(t|t) is minimized. Determining what this minimization implies on **L**(t) will give us its optimal expression.

From (5.1) and (5.3), the error of prediction has the following expression:

$$((\mathbf{x}(t) - \hat{\mathbf{x}}(t|t-1)) = \mathbf{F} \ (\mathbf{x}(t-1) - \hat{\mathbf{x}}(t-1|t-1)) + \mathbf{v}(t-1)$$

This relation leads to the following expression for the error covariance matrix:

$$\mathbf{P}(t|t-1) = \mathbf{F} \ \mathbf{P}(t-1|t-1) \ \mathbf{F}^T + \mathbf{Q} \tag{5.6}$$

In addition, combining (5.1) and (5.5) will lead to:

$$(\mathbf{x}(t) - \hat{\mathbf{x}}(t|t)) = (\mathbf{I} - \mathbf{L}(t) \ \mathbf{H}(t)) (\mathbf{x}(t) - \hat{\mathbf{x}}(t-1|t-1)) - \mathbf{L}(t) \ \mathbf{w}(t-1)$$

i.e.

$$\mathbf{P}(t|t) = (\mathbf{I} - \mathbf{L}(t)\mathbf{H}(t)) \ \mathbf{P}(t|t-1) \ (\mathbf{I} - \mathbf{L}(t)\mathbf{H}(t))^T + \mathbf{L}(t) \ \mathbf{R} \ \mathbf{L}^T(t) \tag{5.7}$$

We now wish to derive an expression for **L**(t) from the previous expression based on the minimization of the trace of this matrix. (5.7) is a quadratic equation (where the unknown is **L**(t), a nxp matrix). The minimization of Tr[**P**(t|t)] can sound complicated a priori. Yet, as for simple second degree equations, we can write (5.7) in its canonical form:

$$\mathbf{P}(t \mid t) = (\mathbf{L}\mathbf{S} - \mathbf{T})(\mathbf{L}\mathbf{S} - \mathbf{T})^T + \mathbf{M} \qquad (5.7\text{Bis})$$

As we know that **P**(t|t) is positive semi-definite, we are sure that **M** will also be positive semi-definite and the minimization of Tr[**P**(t|t)] falls into the zeroing of the term (**L S** - **T**).

By identification between (5.7) and (5.7Bis), we get:

$$\mathbf{S}\mathbf{S}^T = [\mathbf{R} + \mathbf{H}\mathbf{P}(t|t-1)\mathbf{H}^T] \qquad (5.8)$$

and

$$\mathbf{S}\mathbf{T}^T = [\mathbf{H}\mathbf{P}(t|t-1)] \qquad (5.8\text{bis})$$

$$(5.8\text{bis}) \Rightarrow \mathbf{T}^T = \mathbf{S}^{-1}[\mathbf{H}\mathbf{P}(t|t-1)] \quad \Rightarrow \mathbf{T} = \mathbf{P}(t|t-1)\mathbf{H}^T\mathbf{S}^{-T}$$

$$\Rightarrow \mathbf{L}(t) = \mathbf{T}\mathbf{S}^{-1} = \mathbf{P}(t|t-1)\mathbf{H}^T(\mathbf{S}\mathbf{S}^T)^{-1} \qquad (5.9)$$

So finally, injecting (5.8) in (5.9) will provide an expression for the desired value for **L**(t):

$$\boxed{\mathbf{L}(t) = \mathbf{P}(t|t-1)\mathbf{H}^T[\mathbf{R} + \mathbf{H}\mathbf{P}(t|t-1)\mathbf{H}^T]^{-1}} \qquad (5.10)$$

Recursive computation of P(t|t-1)

The last step we need to introduce is a recursive relationship which will update our estimate of the error covariance matrix. Lets write **P**(t|t-1) as **Σ**(t). As we will see, the equations (5.6), (5.7) and (5.10) give us a simple recursion on **Σ**(t).

From (5.7),

$$\mathbf{P}(t|t) = \Sigma(t) - \Sigma(t)\mathbf{H}^T\mathbf{L}^T - \mathbf{L}\mathbf{H}\Sigma(t) + \mathbf{L}\mathbf{H}\Sigma(t)\mathbf{H}^T\mathbf{L}^T + \mathbf{L}\mathbf{R}\mathbf{L}^T$$

and from (5.10),

$$\mathbf{P}(t|t) = \Sigma(t) - \mathbf{L}\mathbf{H}\Sigma(t) - \Sigma(t)\mathbf{H}\mathbf{A}^{-T}\mathbf{H}\Sigma(t) + \mathbf{L}\mathbf{A}\mathbf{L}^T$$

$$\text{where } \mathbf{A} = [\mathbf{R} + \mathbf{H}\Sigma(t)\mathbf{H}^T]$$

R and **Σ**(t) are symmetric and therefore, **A** is too. $\mathbf{A}^T = \mathbf{A}$, $\mathbf{A}^{-T} = \mathbf{A}^{-1}$ and $\Sigma^T(t) = \Sigma(t)$. So by applying the relation (5.10) once again, we finally get:

$$\mathbf{P}(t|t) = \Sigma(t) - \mathbf{L}\,\mathbf{H}\,\Sigma(t) - \Sigma(t)\,\mathbf{H}^T\mathbf{A}^{-1}\,\mathbf{H}\,\Sigma^T(t) + \Sigma(t)\,\mathbf{H}^T\mathbf{A}^{-1}\mathbf{A}\,\mathbf{A}^{-1}\mathbf{H}\,\Sigma^T(t)$$

which, after simplifications, leads to:

$$\mathbf{P}(t|t) = \Sigma(t) - \mathbf{L}\,\mathbf{H}\,\Sigma(t) \tag{5.11}$$

By injecting this expression for **P**(t|t) in the equation (5.6) we will finally get a recursion on **Σ**(t) (i.e. **P**(t|t-1)):

$$\boxed{\Sigma(t+1) = \mathbf{F}\,[\,\Sigma(t) - \mathbf{L}(t)\mathbf{H}(t)\Sigma(t)\,]\,\mathbf{F}^T + \mathbf{Q}} \tag{5.12}$$

The algorithm

By now, we have gathered all the pieces we need and by putting them back together, we will describe the recursive method that will solve our estimation problem. After having guessed some initial values, the algorithm implied by this method is provided by the expressions (5.3), (5.5), (5.10) and (5.12) we've just derived.

More specifically here is a recapitulation of the steps which constitute our algorithm:

(i) $\boxed{\mathbf{L}(t) = \mathbf{P}(t|t-1)\,\mathbf{H}^T\,[\,\mathbf{R} + \mathbf{H}\,\mathbf{P}(t|t-1)\,\mathbf{H}^T\,]^{-1}}$

(ii) $\hat{\mathbf{x}}(t|t) = \hat{\mathbf{x}}(t|t-1) + \mathbf{L}(t)\,(\mathbf{z}(t) - \hat{\mathbf{z}}(t|t-1))$

(iii) $\boxed{\Sigma(t+1) = \mathbf{F}\,[\,\Sigma(t) - \mathbf{L}(t)\mathbf{H}(t)\Sigma(t)\,]\,\mathbf{F}^T + \mathbf{Q}}$

and $\hat{\mathbf{x}}(t+1|t) = \mathbf{F}\,\hat{\mathbf{x}}(t|t)$

and update **H** to **H**(t+1) (i.e. compute the new values of the cross products)

This system will fit a polynomial function of arbitrary dimension and order to the data without requiring the construction of an orthonormal set of functions.

Implied Generalization in the Modeling Space

Rather than interpolating/extrapolating the original data set based on local structures, fitting a polynomial surface will imply a generalization scheme based on the data's global distribution. The maximum order of the polynomial function we wish to fit can be seen as a regularization term which will favor smoothness over out-of-sample error. The typically limited value for

this order compared to the size of a typical training data set suggests that this method will usually be safe from data over-fitting; however, one cannot say the same about eventual under-fitting. In a way, the regularization imposed by this method will tend to be predominant over the actual data-fitting unless the system itself fits a polynomial hypothesis or we choose an insanely large value for the maximum order.

A few words about the Criterion

As we can recall from our derivation of a standard Kalman filter, the entire method is centered around the minimization of the trace of the prediction error covariance matrix, which is equivalent to a least mean square estimation. In the general context of data fitting, such a criterion is justified and its outcome can be satisfying, however, the surface we are estimating is not just any type of surface, it's a prediction surface.

In the light of what we've discussed in Chapter 4 concerning non-locality, we are entitled to wonder if such a criterion is justified in our context. In other words, just because an area of the modeling (or lag) space hasn't been visited very frequently by our data doesn't necessarily mean it is less important than another area over which a lot more data has been observed. In fact, it would even sound reasonable to expect that the less populated areas of the state space correspond to places where the system's state moves more rapidly, suggesting that local Lyapunov exponents would tend to be larger in these areas. We recall that these Lyapunov exponents are derived from the eigenvalues of a local linearization of the system. These can be seen as a measurement of the rate at which a "volume" around this area will evolve in short term through the dynamics of the system. In many cases, this is actually taken to be these exponents' definition rather than a point of view. If instead of "volume" we were to think in terms of error of prediction, large Lyapunov exponents would lead to a more dramatic error propagation throughout the predictive performance of the estimated model. Of course, this line of thoughts is rather intuitive and open to counter-arguments, but it sounds reasonable enough to question the appropriateness of least mean square as a valid criterion in the context of prediction surfaces estimation.

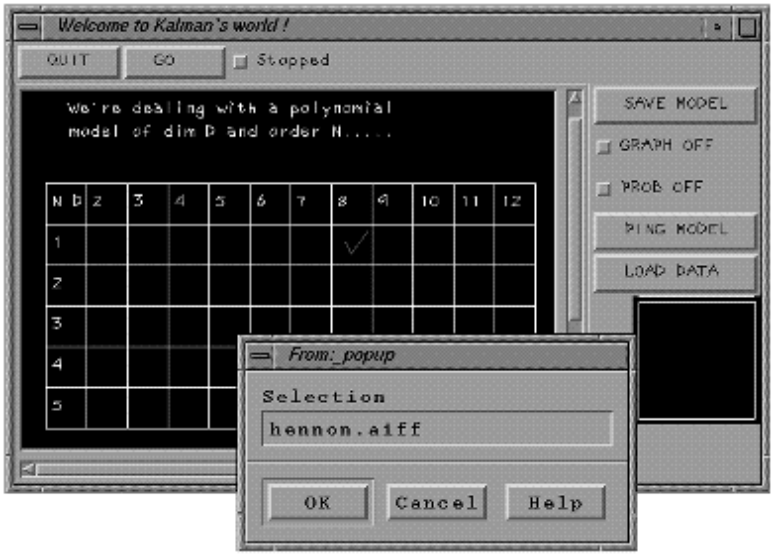
If we were to convince ourselves that every visited area of the state space is just as important to us regardless of its associated population, then an alternative approach would be to insure a uniform distribution of the original training data (at least over the support of our observation). The time constraint of the present work hasn't left the author much time to experiment with such an alternative but a possible path would be to pre-cluster the training data in order to identify uniformly distributed representative data prior to the surface estimation.

Implementation and evaluation

Software

The previous algorithm was implemented in C and tested on an SGI Indigo. It was incorporated along with an X/Motif interface with which the user can select the dimension of

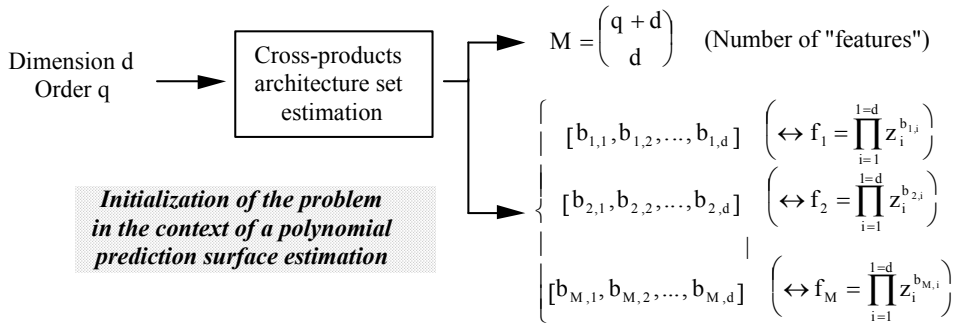
the modeling lag space, the maximum order for the polynomial fit and the file which will be used as training data.



The result of the estimation can be plotted in three dimensions using SGI's GL library. Along with these features, the system computes the out-of-sample error distribution associated with the estimation. It can also plot a preview of the forecast which may result from the estimated model. This software was never intended for anything more general than the evaluation of the proposed method and its set of features reflect the authors curiosity concerning the performance of a Kalman filter in the context of polynomial prediction surfaces estimation.

The core of this system is, of course, the Kalman filter itself. From the expressions that we derived earlier, the actual software implementation is fairly straightforward. The specificity of what one could call the "feature vector" **H** was intentionally kept out of the main Kalman filter's implementation.

We recall that in the context of our polynomial fit in d dimensions with a maximum order q, this feature vector is derived from d successive observations as the set of all the possible cross-products of order q of less. The number of features (i.e. length of **H**) and the architecture (in terms of exponents) of all the valid cross-products are computed once and for all based on the user's choices concerning (d,q).



After posing the problem in terms of these cross-products, we only need to initialize the polynomial surface's parameter with some arbitrary values and let the Kalman filter do the rest:

```
void Kalman(float *Observations, int NbObs)
{
  int k,l,n,m,nbObs;
  double TheObservation,tmp;
  float *data;

  nbObs = NbObs; data = Observations;
       ...
```

The first feature vector is computed from the first d observations by our problem-specific process. In our case, this process is derived from the cross-product architectures we found previously.

```
       ...
  MakeFeatureVector(Observations, H);
  NbObs += -Dimension;

  while(NbObs>0)       /* loop on the full set of
observations */
    {
      TheObservation            =            (double)
(Observations++)[Dimension];

      /* Step one: Compute L(t) */
      for(k=0;k<NbParam;k++)
       {
         Stemp = 0.0;
         for(l=0;l<NbParam;l++)  Stemp  +=  Sigma[k][l]
* H[l];
         Vtemp[k] = Stemp;
       }
      Stemp = R;
      for(k=0;k<NbParam;k++)    Stemp    +=    H[k]    *
Vtemp[k];
      for(k=0;k<NbParam;k++) L[k] = Vtemp[k] / Stemp;

      /* Step two: New Estimation */
      Prediction = 0.0;
```

```
      for(k=0;k<NbParam;k++)  Prediction  +=  H[k]  *
Estimate[k];
      Stemp = (TheObservation - Prediction);
      for(k=0;k<NbParam;k++)  Estimate[k]  +=  L[k]  *
Stemp;

      /* Step three: Update Sigma */
      for(k=0;k<NbParam;k++)
       {
         Stemp = 0.0;
         for(l=0;l<NbParam;l++)   Stemp   +=   H[l]   *
Sigma[l][k];
         Vtemp[k] = Stemp;
       }
      for(k=0;k<NbParam;k++)
      for(l=0;l<NbParam;l++)
         Sigma[l][k] += Q[l][k] - ( L[l] * Vtemp[k]
);

      /* Step four: Update H(t) */
      MakeFeatureVector(Observations, H);

      /* Update number of observation treated */
      NbObs += -1;
    }
       ...
```

After the previous loop, we are ready to return the result of our estimation (i.e. the array 'Estimate[.]'). In the context of this particular piece of code, the following are calls to some plotting function and self evaluation scheme:

```
       ...
  PlotKalmanResult(data, nbObs);
  Evaluation(data, nbObs);
}
```

Tests and Evaluation

Hennon Map

As a first test, the author wanted to analyze a simple system that is known to be deterministic, low dimension and non-linear. The Hennon map was a perfect candidate as its chaotic behavior was already an obstacle to any linear system approach. In addition, its dimension being only 2, this would allow a meaningful 3D plot of the resulting prediction surface.

Specifically, we chose the following system: $\begin{cases} x(t+1) = y(t) + 1 - 1.4\,x(t)y(t) \\ y(t+1) = 0.3\,x(t) \end{cases}$

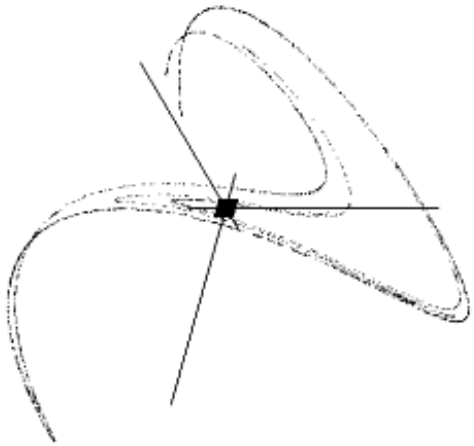


Fig. 5.5 - Plot of a Hennon map attractor from 2000 successive values of z(t) in a lag space.

From arbitrary initial values for x(0) and y(0), we iterate these difference equations in order to build our training data. Our ultimate time series z(t) (which we decide to save as an AIFF sound file) is a normalized version of the series x(t) between -1 and 1. Figure 5.5 is a plot of this data in a 3D lag space. Finally, a quick look at the previous system of difference equations should suffice to convince the reader of the fact that a polynomial fit should do a fairly good job at modeling this system. Indeed, the description of the system itself is a couple of simple polynomial expressions.

Although this system is known to exhibit a chaotic behavior, its (fractal) set of solutions in its state space has a very clear structure. In the light of the embedding theorem, we shouldn't be surprised when we observe the same characteristic structure (strange attractor) in a lag space reconstructed from successive observations of z(t) (see previous figure).

Choosing the maximum order of our polynomial surface to be equal to 1 is equivalent to fitting a linear model to our data. Given the non-linearity of this data, it is not surprising that such a fit is a dramatic mismatch. The following figure is the result of the fitting of a linear model of dimension 12.

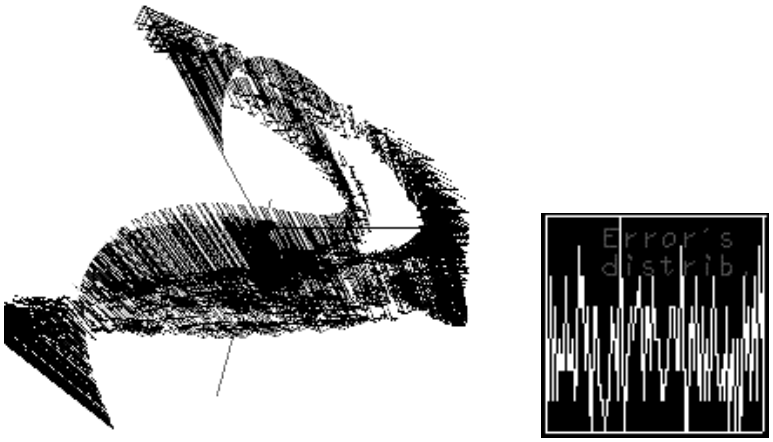


Fig. 5.6 - Attempt to fit a linear model to a Henon map.

Of course, we can't plot a 12-dimensional hyperplane and for these situations, the author chose to plot prediction errors as arrows in three dimensions. Each arrow is scaled appropriately with respect to the error value it represents.

If the previous plot doesn't convince us that we are facing a dramatic model mismatch problem, maybe a look at the estimation of the error's probability density will. This density only plots occurrences of the absolute value of the out-of-sample error between 0.0 and 0.1 but the error density is fairly uniform over the entire possible range. In this particular case, the program tells us that the mean of this absolute value is 0.433628 and the linear estimation doesn't capture much of the system's structure.

Now if instead of fitting hyperplanes we allow the system to use higher order terms (even as small as order 2) for a polynomial surface of two dimensions only, then we get much more encouraging results.

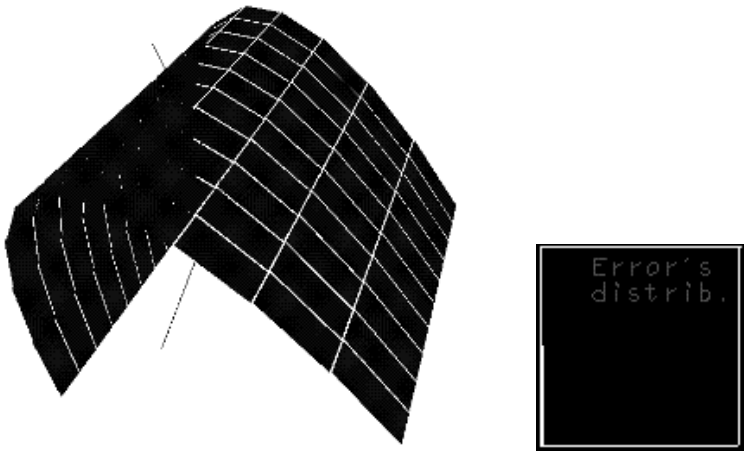


Fig. 5.7 - Fit of a Henon map by a 2-dimensional second order polynomial prediction surface.

Our Kalman filter fits very quickly (only once through the data set) and very accurately a second order polynomial surface to the observed data. The average for the absolute value of the resulting prediction error is only 0.000059 and it is barely perceptible in the previous plot of the error's density.

Even more importantly, using the two first samples from our observation z(t) and iterating the estimated polynomial model leads to an astonishing reconstruction. The previous plot overlays a time-series representation of the forecast on top of a plot in a 3D lag space. As one can notice, the shape of this set is barely distinguishable from the original data set.

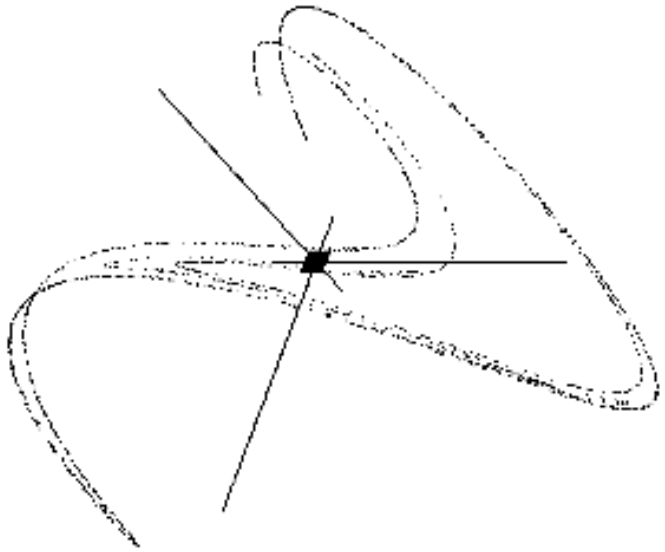


Fig. 5.8 - Data reconstruction by iterating the estimated polynomial model.

Audio Data

In sight of this very encouraging result, the author decided to test this scheme on an audio recording of a musical instrument. We chose the same normalized small quasi-stationary chunk of sampled sound (produced by the bowing of a violin string) that we chose earlier in Chapter 4. A previous entropy measurement suggested that this time series could be modeled via a non-linear system with only 4 or 5 degrees of freedom.

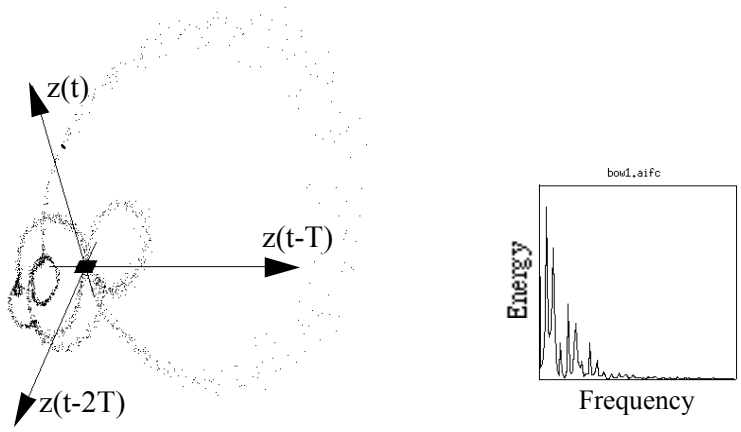


Fig. 5.9 - Lag space plot and spectrogram of a quasi-stationary chunk of a violin recording (same example as Chapter 4).

As we pointed out earlier, the algorithm requires the user to suggest a maximum order for the polynomial function to be fitted. The Kalman filter may require a few passes through the data set in order to converge and the observation of the out-of-sample error after each pass is an indicator of its state of convergence. The following figure plots this error after a single pass through the data (the dimension of the model was set to 4 and the order of the polynomial to 4). We recall that in this 3D lag space, the arrows go from the real data to their corresponding estimate through the model.

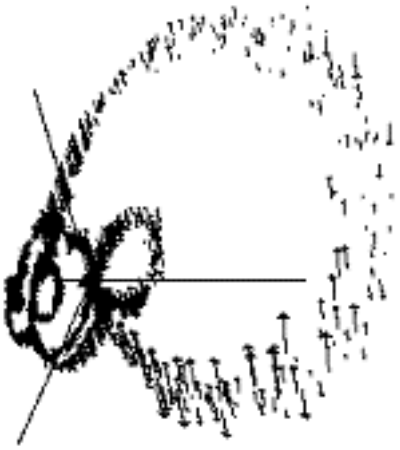


Fig. 5.10 - Out-of-sample error in a 3D lag space

The error mean is probably the most obvious quantity to observe when evaluating the performance of an estimator. The following figure shows this quantity as a function of the number of passes through our data set for different architectures for our model (in order to interpret the values for the error, we recall that the original data was normalized between -1 and 1).

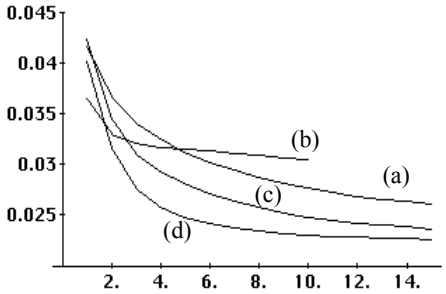


Fig. 5.11 - Evolution of the error mean as a function of the number of passes through the data: (a) Model of dimension 4 and order 4. (b) Model of dimension 5 and order 3. (c) Model of dimension 4 and order 5. (d) Model of dimension 5 and order 4.

A high error mean will definitely indicate a model mismatch but even if this mean is small, we are not guaranteed that the model doesn't miss some important structure in our data. As a sanity check, we might want to make sure the out-of-sample error distribution doesn't reveal any particular structure.

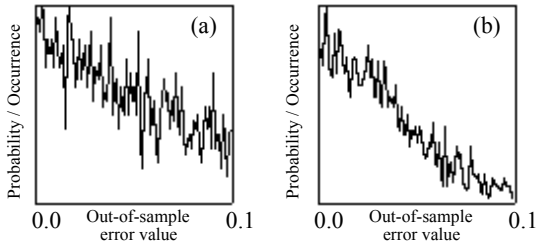


Fig. 5.12 - Out of sample error distribution for a polynomial model of dimension 4 and order 4. (a) after a single pass through the data; (b) after 15 passes through the data.

The previous and the following plots are histograms of the absolute value of the out-of-sample error for different model architectures (i.e. different orders) and at different stages of convergence.

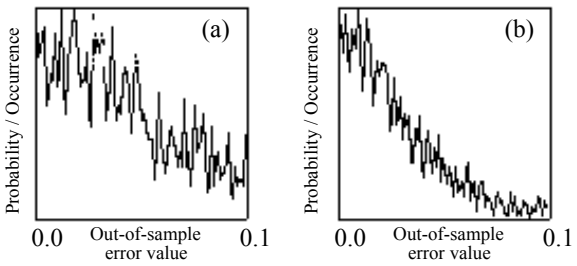


Fig. 5.13 - Out of sample error distribution for a polynomial model of dimension 4 and order 5. (a) after a single pass through the data; (b) after 15 passes through the data.

The shape of the resulting error distributions and the error mean definitely indicate that our polynomial fit captures some of the system's non-linearities. However, the resulting fit is not

nearly accurate enough to lead to any meaningful reconstruction of the entire training data from 4 initial conditions. Similar results were observed with chunks of sound recorded from other instruments. We are most likely facing another model mismatch problem and although this polynomial fit may sound satisfying in terms of out-of-sample error, it is not good enough to cope with non-locality, prohibiting any long-term prediction.

Expertise and Applications

Expertise

As illustrated by its application to a Hennon map, this polynomial approach has a lot of potential for the study of small and simple, yet non-linear systems. No linear system would be able to capture the structure of a non-linear (and eventually chaotic) system like this polynomial fit. Furthermore, this analysis is computationally cheap and its recursive mechanism makes it flexible and easy to use. In the case of larger (or more complex) non-linear systems, the size of the problem to solve quickly explodes with increasing values of the modeling space's dimension d and the maximum order q for the polynomial fit. Without even talking in terms of prohibitive sizes for the associated computation, the recursive algorithm derived from a Kalman filter will require a lot more time to converge.

However, even if the resulting model doesn't allow long term prediction, it will still capture the major non-linearities of the system in the global form of a polynomial function. A full-blown accurate model for the system would be ideal but a "fairly accurate" description of its mechanism is still a great analysis tool.

Possible Applications

The following applications have not been implemented as they don't necessarily fit in the context of the present document. They are a small set of the author's ideas concerning the potential of the modeling approach that was suggested in this chapter.

System Monitoring: The recursive modeling scheme we've introduced would be a perfect tool for the monitoring of a known non-linear system. It could detect and identify small variations in the system's behavior which may not appear through any linear system theory approach. Identifying the nature of these variations may require a detailed knowledge about the physical mechanism of the system under observation. However, even if we don't have any other information except for the fact that system is "fixed" or "time-invariant", this approach could still provide valuable cues concerning possible malfunctions or perturbations.

Physical Understanding: A multi-variable polynomial difference function may not appear as a very intuitive representation of a physical system. However, suppose this single and heavy difference equation was to be reduced to several lower-order difference equations. Even if the associated model isn't a perfect fit to the observed data, such a system of small difference equations could provide hints concerning the internal mechanism of the physical system that was observed, leading to plausible theories as to what this system really is in the physical world.

Data Compression: Polynomial models for a prediction surface are a clear extension of the notorious auto-regressive (AR) models from linear system theory. Although the accuracy of an AR model can only be worse than of a polynomial model, such models have been used extensively for compression purposes. Linear predictive coding (LPC) in the case of speech signals is based on the fact that it take less bits to encode a full set of parameters of an AR model along with a small energy prediction error, than it takes to encode the raw waveform. If the nature of the signal that we wish to encode is intrinsically non-linear, the approach suggested in this chapter could lead to a more appropriate encoding scheme. Our recursive estimation of a polynomial prediction function can be used for "Polynomial predictive coding".

Chapter Summary

In light of the modeling scheme which we've introduced in Chapter 4, we've suggested the use of multi-variable polynomial functions for the estimation of a global form for a prediction surface in a lag space of observations. This suggestion was obviously motivated by the generality of polynomial forms as well as a wish for a global description of a system's behavior. Rather than approaching this estimation problem in terms of the identification of an orthonormal basis of polynomials on which to project the observed data, we've chosen to consider all appropriate cross-products as observed features, essentially reducing our task to a familiar ill-conditioned linear problem $\mathbf{Ax}=\mathbf{z}$.

We've then suggested a Kalman filter as a recursive approach to solving this linear problem and took this opportunity to derive it rigorously with the specifics of our problem in mind. The resulting algorithm was implemented and applied to various test cases.

Although the results of these few tests don't seem to indicate that this approach may be suitable for sound synthesis, this method turns out to be very useful in other contexts. We've suggested only a few alternative uses for this approach (system monitoring, physical understanding, and data compression) but it is the author's belief that these don't nearly span the full potential of this approach.

As for sound synthesis, our results seem to indicate that we should look for alternative local approaches to the non-linear modeling of a time-series if we wish to reach a degree of accuracy that might cope with non-locality for long-term predictions.

Chapter 6

Cluster-Based PMF Models

Non-locality and stability issues require an accuracy that is very difficult to achieve while estimating a global model. If the major concern is the model's ability to resynthesize the original system's behavior, we are forced to turn to alternative approaches that will provide better accuracy in terms of data matching. In what follows, we will introduce a general approach towards the estimation of local models of the data's probability mass function, for which each sub-area of expertise results from a non-supervised clustering process applied to the observed data. The resulting process (Cluster-Weighted Modeling) unifies clustering and modeling in a single process without making drastic assumptions concerning the system's architecture.

Cluster-based Probability Distribution Estimation

As we already suggested in Chapter 4, the estimation of a probability distribution, and especially conditional probability distributions, of the observed data can provide very valuable information about the systems behavior for its characterization. It can also be taken literally as a model from which one can reproduce, interpolate and extrapolate the original system's behavior.

Justification of a Local Approach

There are many reasons why estimating a global form for a prediction function (or the model) of a system may sound attractive. After all, the system itself is an entity. Partitioning a state space doesn't have much of a chance to be physically meaningful. The general behavior of a

system (or its observation in lag spaces) is the information that we want to capture, and its local behavior in a sub-region of this already arbitrary lag space might not seem as relevant.

However, the complexity of a global approach to this modeling problem in terms of degrees of freedom is discouraging. An approach such as the previous recursive global polynomial may be a great way to acquire some information about the general architecture of the system, but the complexity of a global model's estimation will make any accuracy of modeling very difficult to achieve. If the major purpose of our model is its ability to resynthesize the original system's behavior, we are forced to turn to alternative approaches that will provide better accuracy in terms of data matching.

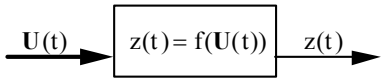
If a given problem is too complex, it is generally reasonable to attempt to break it down into several smaller problems of limited complexity. Viewed as a surface estimation task, it is natural to think of breaking down our modeling problem into several local surface estimations. Each one will have a constrained region of expertise within the system's lag space. Whether we are thinking of a radial basis functions (RBF) approach or something more general such as a probability distribution, the first step to consider will always be the identification of a set of relevant regions. At this point, non-supervised clustering becomes a natural process to think about.

Suggested General Form for the Model

A cluster-based estimated probability distribution is one whose form will reflect a collaboration between multiple local models whose areas of expertise resulted from some clustering of the observed data.

The Model

Let's consider the system that we wish to model in the general form of the following black box.



Here, z(t) stands for the output of the system while the vector **U**(t) is the set of the system's input. In the case of an autonomous deterministic system of finite dimension and in light of the Embedding Theorem, **U**(t) would be nothing but a set of lag values of the output z(t) (i.e. $\mathbf{U}(t) = \mathbf{U}(t,\tau) = \left(z(t-\tau),..,z(t-d\tau)\right)^T$).

Expressing the joint probability distribution of the output z and the input **U** based on some clustering procedure in the space (z,**U**) will eventually lead to a general form of the following type:

$$p(z,\mathbf{U}) = \sum_{m=1}^{M} p(z,\mathbf{U}|\,Cl_m)\,p(Cl_m)$$

(where M is the number of clusters)

In the previous expression, Cl_m (for m=1 to M) stands for the hypothesis associated with the expertise of the m^{th} cluster; the conditional $p(z, \mathbf{U} | Cl_m)$ is the expression of that particular expertise; and $p(Cl_m)$ is the prior of that hypothesis (or cluster, or class).

In order to turn this information into a form that exhibits the ability to synthesize further data z given some input **U**, this joint probability distribution needs to lead to an expression of the conditional probability distribution of the output given the input of our system. Of course, this is achieved by Bayes' rule:

$$p(z | \mathbf{U}) = \frac{p(z, \mathbf{U})}{p(\mathbf{U})} = \frac{\sum_{m=1}^{M} p(z, \mathbf{U} | Cl_m)\, p(Cl_m)}{\sum_{m=1}^{M} p(\mathbf{U} | Cl_m)\, p(Cl_m)}$$

Only then can we refer to our representation in terms of a model with the help of some decision rule such as maximum likelihood or Bayes least square.

Maximum Likelihood

From the joint probability density of two random variables (or vectors) p(x,y), the likelihood function of an observed instance x = X is defined as the function $y \rightarrow p(X, y)$. Of course, substituting instances of the random variable (or vector) x with an observation X in the expression of the joint probability density p(x,y) leads to the expression of the conditional probability distribution p(y|x=X) as they only differ by a constant multiplicative factor (p(x=X)). Maximum likelihood is a very general and useful decision rule which consists in maximizing the likelihood of the instances that were observed. In other words, it will identify the value $\hat{y}$ of y which maximizes p(X,y)=p(y|x=X)p(x=X). Rather than maximizing the likelihood function of the observation, it often turns out to be more convenient to maximize the logarithm of this function (referred to as the log likelihood). This is mainly due to the overwhelming omnipresence of Gaussian distributions. From the monotonicity of the logarithm, this slight variation doesn't influence the ultimate result of this decision rule. The validity of the maximum likelihood as a decision rule will not be discussed here but we'll simply say that it becomes questionable when the number of observations is small.

Back to our context, the choice of maximum likelihood would lead to the following decision rule (or process):

$$\hat{z}_{ML}(\mathbf{U}(t)) = \arg\max_z \; p(z | \mathbf{U} = \mathbf{U}(t)) = \arg\max_z \; \ln\left(p(z | \mathbf{U} = \mathbf{U}(t))\right)$$

"Given the input **U**(t), the output $\hat{z}_{ML}(\mathbf{U}(t))$ will be chosen such that it maximizes the conditional probability $p(z | \mathbf{U} = \mathbf{U}(t))$"

Bayes Least Squares

If maximum likelihood is the most popular decision rule, Bayes least squares has to be its main challenger. Instead of picking an instance for the random variable which maximizes locally a conditional probability, Bayes least squares suggests a decision based on a more global observation of this conditional. As suggested by the name of this decision rule, it is based on the minimization of the squared prediction error:

$$\varepsilon = E\left[\left(\hat{z} - z\right)^2 \middle| \mathbf{U}\right] = \hat{z}^2 - 2\,\hat{z}\,E\left[z \middle| \mathbf{U}\right] + E\left[z^2 \middle| \mathbf{U}\right]$$

and naturally, $$\frac{\partial \varepsilon}{\partial \hat{z}} = 0 \Rightarrow \hat{z} = E\left[z \middle| \mathbf{U}\right]$$

"Given the input **U**(t), the output $\hat{z}_{BLS}(\mathbf{U}(t))$ will be chosen to be the conditional expectation of the output given that particular instance for the input".

Clustering

Clustering expresses a wish for data summarization. While the complexity of the eventual model is intuitively related to the number of relevant classes that were identified by the clustering process, this complexity should reflect the original system's mechanism and not be the artificial product of the training data size. Summarizing the original data set implies two obvious properties: It should lead to a smaller description than the original data and yet capture the totality (in the ideal case) of the data's relevant information.

Issues

For that purpose, the basic philosophy behind clustering is to identify a limited number of "tendencies" for the data over each one of which some striking cohesion can be observed and eventually summarized accurately via some averaging or other short statistical information.

Identifying clusters or classes requires some measurement of similarity between the objects that constitute our training data. Clustering can be approached from a variety of abstract and global points of view with criteria such as the minimum description length (from information theory) or a minimization of free energy (from statistical mechanics), but it often involves an initial choice of a metric (or distance). In the simplest case, that metric could be a Euclidian distance in the original space where the training data lives, implying that similar objects are close one to another. In that case, a successful clustering procedure would lead to a set of spread classes that would spatially span the region in which the training data lives. It could also be that the chosen metric was derived from some more complex functional relationships, to the point where the spatial distribution of the original data is not all that relevant to the desired classification. In the latter case, the reference to the word "metric" can even become questionable and one might prefer a more implicit reference such as a particular choice for the

form of a probability distribution or a functional relationship. The particular choice of a metric (or probability distribution) will achieve a specific type of clustering and rather than being arbitrary, it should always be influenced by the ultimate use of the resulting clusters (or classes).

Even when the desired criterion can be expressed in closed form, its optimization (minimization or maximization) usually calls for a recursive process. On top of issues such as the choice of a metric and the properties of the clusters, this adds the question of the algorithm's convergence.

Proposed General Clustering-based Modeling Scheme

When clustering is desired in a context where the definition of a metric is meaningful and where clusters will indeed group data based on some notion of proximity, common approaches such as K-means, ISODATA or "softer" versions of the preceding are intuitively satisfying. However, as Professor Gershenfeld would suggest increasingly involved functional relationships and forms of probability distributions, we quickly reached a state of confusion where intuition had substituted understanding. Therefore, we decided to stick to our probability-based point of view and approach the estimation of cluster centroids with our notations in a straightforward but rigorous fashion. The following derivation resulted from a collaboration with Professor Gershenfeld and Bernd Schoner.

Suggested "General" Form

As the desired architecture of a model for the data's probability distribution becomes more sophisticated, it also becomes more and more involved in the clustering as well. One can no longer consider "clustering" and "modeling" as two separate stages towards the inference of the model. In order to ensure meaningful clusters, we need to know ahead of time what the main architecture of our model will be. We suggest the following form for the data's probability distribution.

We recall from earlier:

$$p(z,\mathbf{U}) = \sum_{m=1}^{M} p(z,\mathbf{U}|\,Cl_m)\,p(Cl_m)$$

We suggest to write the conditional of $(z,\mathbf{U})$ given the hypothesis Cl_m to be a generalized version of a separable Gaussian random vector as follows:

$$p(z,\mathbf{U}|\,Cl_m) = K_{m,z}\,e^{-(z-f(\mathbf{U},\beta_m))^2/\,2\sigma_{m,z}^2} \prod_{k=1}^{d} K_{m,k}\,e^{-(u_k-\mu_{m,k})^2/\,2\sigma_{m,k}^2} \qquad (6.1)$$

where of course $K_{m,z} = \dfrac{1}{\sqrt{2\pi\sigma_{m,z}^2}}$ and $K_{m,k} = \dfrac{1}{\sqrt{2\pi\sigma_{m,k}^2}}$ for normalization purposes;

and where $f(\mathbf{U}, \beta_m): \Re^d \rightarrow \Re$ can be seen as a local (possibly non-linear) model of the data restricted to the support of the cluster Cl_m. Over the expertise of the m^{th} cluster, this local model is parametrized via some set β_m.

This form is sufficiently general to include a large part of the approaches that one can encounter; but it obviously doesn't pretend to be universal.

Cluster Centroids Update

By letting μ_m refer to the centroid of the m^{th} cluster in the d-dimensional space **U**, by definition we have:

$$\mu_m = E[\mathbf{U}|\, Cl_m] = \int_{\mathbf{U}} \mathbf{U}\, p(\mathbf{U}|\, Cl_m)\, d\mathbf{U} = \iint_{z,\mathbf{U}} \mathbf{U}\, p(z, \mathbf{U}|\, Cl_m)\, d\mathbf{U}\, dz$$

Using Bayes' rule, the conditional distribution of (z,**U**) given a class (or cluster) can be restated as follows.

$$p(z, \mathbf{U}|\, Cl_m) = \frac{p(Cl_m|\, z, \mathbf{U})\, p(z, \mathbf{U})}{p(Cl_m)}$$

and by substitution, $\mu_m = \frac{1}{p(Cl_m)} \iint_{z,\mathbf{U}} \mathbf{U}\, p(Cl_m|\, z, \mathbf{U})\, p(z, \mathbf{U})\, d\mathbf{U}\, dz$,

which is equivalent to the following expression using expectations:

$$\mu_m = \frac{1}{p(Cl_m)} E[\mathbf{U}\, p(Cl_m|\, z, \mathbf{U})]_{p(z,\mathbf{U})}$$

This little exercise would seem vain if it weren't for the fact that all we have is a set of observed data. Because of this, the preceding expectation is nothing more than a sum over the observed data (each observation has probability 1/N where N is the total number of observations). In other words, a training data set carries implicitly the form of the data's probability distribution (an assumption which any Monte-Carlo process relies upon). Substituting our expectation with this sum will lead to the following.

$$\mu_m = \frac{1}{p(Cl_m)} \frac{1}{N} \sum_{i=1}^{N} \mathbf{U}^{(i)} p(Cl_m|\, z^{(i)}, \mathbf{U}^{(i)})$$

This relationship can be seen as the equation that needs to be solved in order to estimate the clusters' centroids. Of course, it is very unlikely that we could solve this equation analytically and we'll end up implementing the following recursion until we reach stability:

$$\mu_m \longrightarrow \mu_m^{new} = \frac{\sum_{i=1}^{N} \mathbf{U}^{(i)} p\left(Cl_m | z^{(i)}, \mathbf{U}^{(i)}\right)}{N\, p\left(Cl_m\right)} \tag{6.2}$$

In practice, when we choose a specific functional relationship upon which to base our clustering, the conditional $p\left(z, \mathbf{U} | Cl_m\right)$ is much easier to define then $p\left(Cl_m | z, \mathbf{U}\right)$ but Bayes' rule will rescue us once again:

$$\begin{aligned} p\left(Cl_m | z, \mathbf{U}\right) &= \frac{p\left(z, \mathbf{U} | Cl_m\right) p\left(Cl_m\right)}{p(z, \mathbf{U})} \\ &= \frac{p\left(z, \mathbf{U} | Cl_m\right) p\left(Cl_m\right)}{\sum_{j=1}^{M} p\left(z, \mathbf{U} | Cl_j\right) p\left(Cl_j\right)} \end{aligned} \tag{6.3}$$

As for the prior $p(Cl_m)$ of each cluster, they can be updated based on the following relationship (Note that we're using the same "Monte-Carlo" assumption in order to turn an expectation into an average over the observed data):

$$\begin{aligned} p\left(Cl_m\right) &= \iint_{z,\mathbf{U}} p\left(z, \mathbf{U}, Cl_m\right) dz\, d\mathbf{U} \\ &= \iint_{z,\mathbf{U}} p\left(Cl_m | z, \mathbf{U}\right) p(z, \mathbf{U})\, dz\, d\mathbf{U} \\ &= \frac{1}{N} \sum_{i=1}^{N} p\left(Cl_m | z^{(i)}, \mathbf{U}^{(i)}\right) \end{aligned} \tag{6.4}$$

We believe that this general scheme provides a "confusion free" (some would say "no nonsense") approach to clustering in a wide variety of problems where classes of behavior need to be identified. Any arbitrary decision or tweaking resides in the chosen form of a conditional or a joint probability where a confusing statement of "metric" or "distance" is not necessary.

Note: Also, the little exercise that enabled us to turn a conditional expectation into an average over the instances of our data will be encountered more than once. This led Pr. Neil Gershenfeld to define the resulting averaging as a ***"Cluster-weighted expectation"*** for which he uses the following notations:

$$\mu_m^{new} = \frac{\sum_{i=1}^{N} \mathbf{U}^{(i)} . p\left(Cl_m | z^{(i)}, \mathbf{U}^{(i)}\right)}{N . p\left(Cl_m\right)} \equiv \langle \mathbf{U} \rangle_m$$

We will encounter more of this object while deriving the input and output's conditional variances in what follows.

Conditional Variances Update

From the chosen form (6.1) for the data's conditional probability distributions, it is straightforward to derive:

$$p(u_k|\,Cl_m) = K_{m,k}\, e^{-(u_k-\mu_{m,k})^2/\,2\sigma_{m,k}^2}$$

from which our familiarity with Gaussian distributions leads to the following expressions for the various conditional variances for the input:

$$\begin{aligned}\sigma_{m,k}^2 &= \int (u_k-\mu_{m,k})^2\, p(u_k|\,Cl_m)\, du_k \\ &= E\left[(u_k-\mu_{m,k})^2|\,Cl_m\right]\end{aligned}$$

Using Bayes' rule like we did for the centroids will turn this expectation into an average and using our brand new notion of *cluster-weighted expectation*, this will lead to the following:

$$\sigma_{m,k}^{2,\,new} = E\left[(u_k-\mu_{m,k})^2|\,Cl_m\right] = \frac{\sum_{i=1}^{N}(u_k^{(i)}-\mu_{m,k})^2\, p(Cl_m|\,z^{(i)},\mathbf{U}^{(i)})}{N\, p(Cl_m)} = \left\langle (u_k-\mu_{m,k})^2\right\rangle_m \qquad (6.5)$$

As for the input's variance $\sigma_{m,z}^2$, we have to proceed with some caution. For instance, the first term of the product in the expression (6.1) should not be mistaken for the probability distribution of the output z given the hypothesis of the cluster Cl_m. This term is a function of **U** as well. What is certain however, is that:

$$\begin{aligned}p(\mathbf{U}|\,Cl_m) &= \int_z p(z,\mathbf{U}|\,Cl_m)\,dz \\ &= \int_z K_{m,z}\, e^{-(z-f(\mathbf{U},\beta_m))^2/\,2\sigma_{m,z}^2}\,dz \;\left(\prod_{k=1}^{d} K_{m,k}\, e^{-(u_k-\mu_{m,k})^2/\,2\sigma_{m,k}^2}\right)\end{aligned}$$

and as $$\int_z K_{m,z}\, e^{-(z-f(\mathbf{U},\beta_m))^2/\,2\sigma_{m,z}^2}\,dz = 1,$$

we get $$p(\mathbf{U}|\,Cl_m) = \left(\prod_{k=1}^{d} K_{m,k}\, e^{-(u_k-\mu_{m,k})^2/\,2\sigma_{m,k}^2}\right),$$

which leads us to: $$K_{m,z}\, e^{-(z-f(\mathbf{U},\beta_m))^2/\,2\sigma_{m,z}^2} = \frac{p(z,\mathbf{U}|\,Cl_m)}{p(\mathbf{U}|\,Cl_m)}. \qquad (6.6)$$

On the other hand, we also know (Gaussian distribution) that the desired value for $\sigma^2_{m,z}$ will verify:

$$\sigma^2_{m,z} = \int_z \left(z - f(\mathbf{U},\beta_m)\right)^2 K_{m,z} e^{-\left(z-f(\mathbf{U},\beta_m)\right)^2 / 2\sigma^2_{m,z}} dz$$

Its non-dependence upon **U** further leads to:

$$\sigma^2_{m,z} = \iint_{z,\mathbf{U}} K_{m,z} e^{-\left(z-f(\mathbf{U},\beta_m)\right)^2 / 2\sigma^2_{m,z}} p\left(\mathbf{U}|\, Cl_m\right) dz\ d\mathbf{U}$$

.

Using the identity (6.6) in this last expression, we finally get:

$$\begin{aligned}\sigma^2_{m,z} &= \iint_{z,\mathbf{U}} \left(z - f(\mathbf{U},\beta_m)\right)^2 \frac{p\left(z,\mathbf{U}|\, Cl_m\right)}{p\left(\mathbf{U}|\, Cl_m\right)} p\left(\mathbf{U}|\, Cl_m\right) dz\ d\mathbf{U} \\ &= \iint_{z,\mathbf{U}} \left(z - f(\mathbf{U},\beta_m)\right)^2 p\left(z,\mathbf{U}|\, Cl_m\right) dz\ d\mathbf{U}\end{aligned}$$

Again, we recognize another case where our *cluster-weighted expectation* becomes handy and leads finally to the following expression for the appropriate variance $\sigma^2_{m,z}$.

$$\begin{aligned}\sigma^{2,new}_{m,z} &= E\left[\left(z - f(\mathbf{U},\beta_m)\right)^2 |\, Cl_m\right] \\ &= \frac{\sum_{i=1}^{N} \left(z^{(i)} - f(\mathbf{U}^{(i)},\beta_m)\right)^2 p\left(Cl_m|\, z^{(i)},\mathbf{U}^{(i)}\right)}{N\ p\left(Cl_m\right)} = \left\langle \left(z - f(\mathbf{U},\beta_m)\right)^2 \right\rangle_m\end{aligned} \tag{6.7}$$

Local Models Update

As we stated earlier, the parametric functions $f\left(\mathbf{U},\beta_m\right): \Re^d \rightarrow \Re$ can be comprehended as local models over the expertise of their associated cluster. Given the initial choice of an architecture for these (simple) local models, the parameter set β_m will have to be tuned for each cluster accordingly to some criteria. These parameters can be derived by maximizing the likelihood (or log-likelihood) of the observed data over the restricted expertise of Cl_m as shown here:

$$0 = \frac{\partial}{\partial \beta_m} E\left[\log\ p\left(z^{obs}, \mathbf{U}^{obs} | Cl_m\right) |\ Cl_m\right] = \frac{\partial}{\partial \beta_m} \left\langle \log\ p\left(z, \mathbf{U} | Cl_m\right) \right\rangle_m$$
$$= \frac{\partial}{\partial \beta_m} E\left[\left(z^{obs} - f(\mathbf{U}^{obs}, \beta_m)\right)^2 |\ Cl_m\right] = \frac{\partial}{\partial \beta_m} \left\langle \left(z - f(\mathbf{U}, \beta_m)\right)^2 \right\rangle_m$$
$$= \left\langle \left(z - f(\mathbf{U}, \beta_m)\right) \frac{\partial f(\mathbf{U}, \beta_m)}{\partial \beta_m} \right\rangle_m$$

We can note that "gaussianly-biased" initial choice for the distribution at (6.1) results in an equivalence between maximum log-likelihood and error-mean. The previous expression will lead to an equation (or system of equations depending on the size of the set β_m) from which we will derive the optimal values for this set at any given stage of the clustering:

$$\beta_m^{new} \text{ is chosen such that } \left\langle \left(z - f(\mathbf{U}, \beta_m)\right) \frac{\partial f(\mathbf{U}, \beta_m)}{\partial \beta_m} \right\rangle_m \Bigg|_{\beta_m = \beta_m^{new}} = 0 \qquad (6.8)$$

"Cluster-Weighted Modeling"

As a summary of what we've just derived from our "general purpose" cluster-based modeling scheme: Given some initial choice concerning the form of the local models $f(\mathbf{U}, \beta_m) : \Re^d \rightarrow \Re$ and the number M of clusters that will be used, the clustering and modeling stages of the process have been merged. The expressions which we derived earlier can be reorganized in the form of an algorithm and the resulting method is referred to as *"Cluster weighted modeling"* by Professor Neil Gershenfeld.

	Initialization of $\left(M;\ p(Cl_m);\ \mu_m;\ \beta_m;\ \sigma^2_{m,k};\ \sigma^2_{m,z}\right)$
(i)	Estimate data's conditionals $p\left(z^{(i)}, \mathbf{U}^{(i)} |\ Cl_m\right)$ from (6.1).
(ii)	Estimate clusters' posterior probability $p\left(Cl_m |\ z^{(i)}, \mathbf{U}^{(i)}\right)$ from (6.3).
(iii)	Update clusters' priors $p\left(Cl_m\right)$ from (6.4).
(iv)	Update clusters' centroids μ_m from (6.2), input variances $\sigma^2_{m,k}$ from (6.5) and the local models' parameters β_m from (6.8).
(v)	Update the output's conditional variances $\sigma^2_{m,z}$ from (6.7).
(vi)	Go to (i) if needed (see below).

The preceding is the description of an algorithm that was derived from the previous investigation. Again, its being recursive comes from the fact that we cannot infer the optimal

set of our model's parameters analytically, and that therefore, we are reduced to implementing a step-by-step updating/optimizing process. Knowing when to stop looping can be based on a couple of observations. A first obvious clue will be to evaluate how much the resulting model has changed during the last iteration. If the change that we detect is insignificant, then we are entitled to consider that we did the best we could. The other clue comes directly from the value of the output's conditional variances $\sigma^2_{m,z}$.

Indeed , in the most likely case where we choose to base a prediction on Bayes' least square as a decision rule, we will have:

$$\hat{z}(\mathbf{U}) = z_{BLS}(\mathbf{U}) = E[z|\mathbf{U}] = \int_z z\, p(z|\mathbf{U})\, dz$$

$$= \sum_{m=1}^{M} p(Cl_m|\mathbf{U}) \int_z z\, p(z|\mathbf{U}, Cl_m)\, dz$$

which, given the expression (6.1), will eventually lead to the following:

$$\hat{z}(\mathbf{U}) = \sum_{m=1}^{M} p(Cl_m|\mathbf{U}) \int_z z\, K_{m,0} e^{-(z-f(\mathbf{U},\beta_m))^2/2\sigma^2_{m,0}}\, dz$$

$$= \sum_{m=1}^{M} p(Cl_m|\mathbf{U})\, f(\mathbf{U},\beta_m) = \frac{\sum_{m=1}^{M} p(\mathbf{U}|Cl_m)\, p(Cl_m)\, f(\mathbf{U},\beta_m)}{\sum_{m=1}^{M} p(\mathbf{U}|Cl_m)\, p(Cl_m)}$$

A natural way to estimate the accuracy of our estimate would be to compute the resulting mean square error $\varepsilon = E\left[(z - \hat{z}(\mathbf{U}))^2\right]$. Deriving this quantity analytically is not possible due to the "soft reconstruction" (or overlapping) or the Gaussian-shaped clusters. In spite of an initial intuition, it does not reduce to a combination of the output's conditional variances $\sigma^2_{m,z}$. However, these variances do measure similar quantities over the restricted areas of each local model (i.e. each cluster) and it sounds very reasonable to use these estimates (either the minimum, the maximum, or the average) in order to get an idea concerning how close a fit was achieved by the model. The author would like to stress the fact that these conditional variances should only be taken as indicators as there is no simple analytical relationship between these local fits and the overall performance of the resulting predictor. A further in-depth investigation of this relationship involves the study of the system's (or the model's) smoothness. At an intuitive level, we'll simply say the conditional variances $\sigma^2_{m,z}$ will quantify the predictor's accuracy in a satisfying manner if the resulting model doesn't exhibit any striking discontinuity or sharp transition.

Examples

Straightforward Separable Gaussians

Largely inspired by the works of Pr. Rosalind Picard and Kris Popat [PP93] on the characterization and modeling of visual textures, the author's first attempt to build a cluster-based probability mass function estimation of this sort was based around the following choice of separable Gaussians:

$$p\big(z(t), \mathbf{U}(t)\big) = p\big(z(t), z(t-\tau), \ldots, z(t-d\tau)\big) = \sum_{m=1}^{M} w_m \prod_{k=0}^{d} K_{m,k}\, e^{-(z(t-k\tau)-\mu_{m,k})^2 / 2\sigma_{m,k}^2}$$

In fact, to be chronologically accurate, it was this model that inspired the general form (6.1) which we've introduced earlier. Hence, it is not surprising that this model appears as a special case of our general model:

$$p\big(z, \mathbf{U} \mid Cl_m\big) = p\big(z(t), .. z(t-k\tau) \mid Cl_m\big) = \prod_{k=0}^{d} K_{m,k} e^{-\left(z(t-k\tau)-\mu_{m,k}\right)^2 / 2\sigma_{m,k}^2}$$

$$= K_{m,z} e^{-(z-f(\mathbf{U},\beta_m))^2 / 2\sigma_{m,z}^2} \prod_{k=1}^{d} K_{m,k} e^{-\left(u_k-\mu_{m,k}\right)^2 / 2\sigma_{m,k}^2}$$

$$\text{where } \begin{cases} z = z(t) \\ \mathbf{U} = \mathbf{U}(t) = \big(z(t-\tau), .., z(t-d\tau)\big)^T \\ f(\mathbf{U}, \beta_m) = \mu_{m,0} \in \Re \text{ (i.e. the local model is a constant)} \end{cases}$$

Under these circumstances, the relationships (6.7) and (6.8) reduce respectively to:

$$\sigma_{m,z}^{2,new} = \sigma_{m,0}^2 = \left\langle \big(z - \mu_{m,0}\big)^2 \right\rangle_m \text{ and } \mu_{m,0}^{new} = \langle z \rangle_m .$$

In fact, for those who are familiar with conventional clustering techniques, the resulting clustering technique could be seen as an implementation of "soft K-means".

Test Data

The data we chose to test this approach is the same digital recording of the quasi-periodic part of a bowed violin string we've discussed before. Again, the sampling frequency of the recording is 44.1 KHz and its resolution is 16 bits. A version of the PMF estimator presented earlier was implemented in C. We can explicitly tell the system the maximum number of clusters that it should use as well as the dimension of the lag space it should compute the PMF

for. Once this analysis is completed, we can use this parametric PMF in order to resynthesize the original data. At this point, we can choose between a deterministic and a stochastic approach by using either conditional expectation or conditional probabilities.

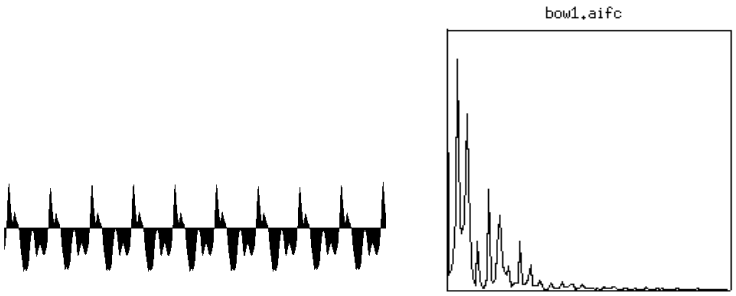


Fig. 6.1 - A chunk of the original data and its FFT-based spectrum estimation.

In the context of our data and by experimenting with various numbers of clusters to be used for the PMF estimation, it appeared that 200 clusters did a good job for dimensions up to 5.

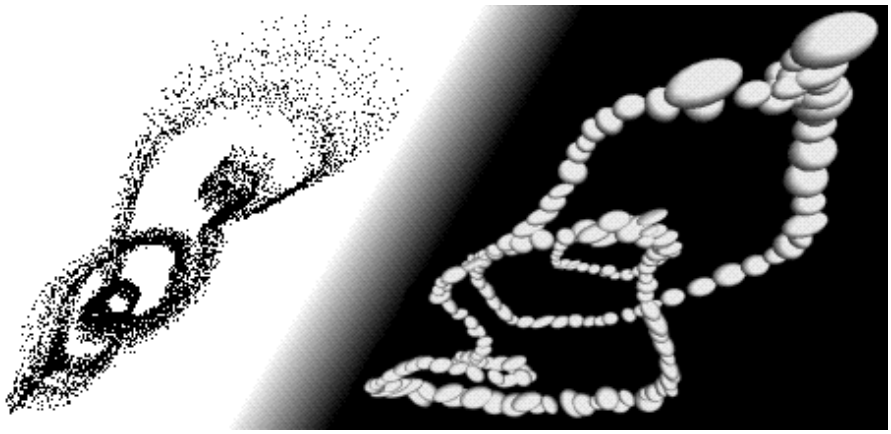


Fig. 6.2 - Observation of a chunk of violin sound in lag space and visualization (in 3D) of the estimated 200 clusters in a 5D lag space..

The following figure presents some conditional probability distributions derived from the estimated PMF in dimension 5. As our knowledge of the past increases (i.e. conditioning), we observe a dramatic reduction of the observation's apparent randomness. In other words, we observe a clear tendency for the model to become deterministic as we increase the dimension of the modeling lag space.

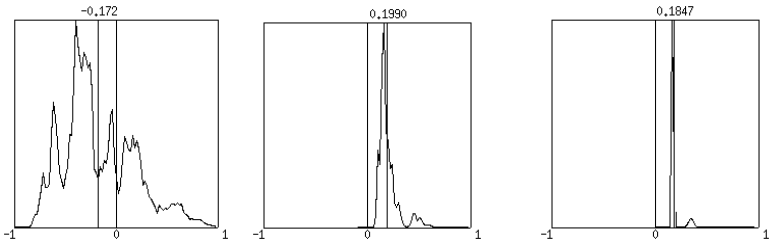


Fig. 6.3 - Estimated PMF with 200 clusters. From left to right: p(z), p(z|u_1), p(z|u_1,u_2) where u_1 and u_2 where chosen arbitrarily.

Test Data Reconstruction

Having an estimate of our data's probability mass function in dimension 5, we can now build a *deterministic model* of dimension 4 as: $\hat{z} = f(u_1, u_2, u_3, u_4) = E[z|\mathbf{U}]$

After feeding the first four samples of the original data as initial conditions, we can iterate this deterministic function f() and try to reconstruct the bowed string sound of the violin. The following figure is the result of this reconstruction.

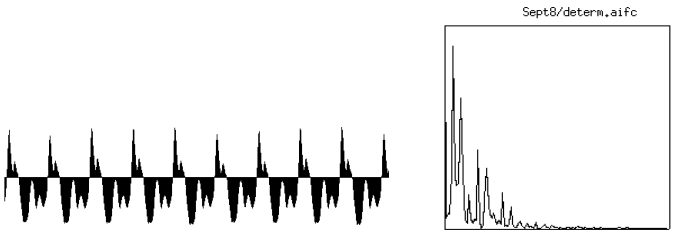


Fig. 6.4 - A chunk of the deterministic reconstruction and its FFT-based spectrum estimation.

The accuracy of this reconstruction is remarkable. Even after over 20000 iterations, the model is perfectly stable and accurate. The result of this reconstruction was saved in a sound file and it is very difficult to tell it apart from the original.

The same dimension 5 probability mass function can be seen as a stochastic model for our data. We can then synthesize an instance of this *stochastic process* by generating random numbers accordingly to the conditional probabilities:

$$p(z|u_1, u_2, u_3, u_4) = p(z|\mathbf{U})$$

Once again, the first four samples of the original data are fed as initial conditions and the rest of the reconstruction results from an iteration of the model. The following figure (Fig. 6.5) is the result of this process.

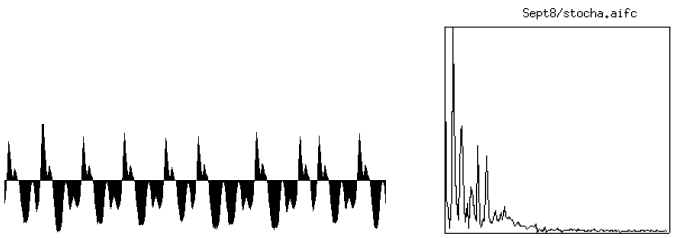


Fig. 6.5 - A chunk of the stochastic reconstruction and its FFT-based spectrum estimation.

In addition to being computationally very expensive, this approach doesn't lead to the accuracy we had with the deterministic approach. Ironically, this model could have been thought of as being more "complete" than the deterministic model which uses only the means of these conditionals. In the light of what we've referred to as the "pessimism" of a stochastic approach in Chapter 4, we shouldn't be shocked.

Relationship with a Radial Basis Functions approach

As we are about to see, there is a clear correspondence between the deterministic approach that we've just derived and a radial basis functions (RBF) approach. In order to illustrate this point, we will show that the previous deterministic approach is rigorously equivalent to a particular case of RBFs.

Let's imagine we wish to characterize the prediction surface of the same system as a linear combination of some radial basis functions. Let's also assume that we choose the anchor points of our RBFs through the same clustering method we've used for the PMF. Let's finally imagine that the d-dimensional RBFs we chose have the same form as the (d+1) dimensional Parzen windows we used.

This means that our model for f() is:

$$z = f(u_1, u_2, .., u_d) = \sum_{m=1}^{M} \alpha_m \cdot \mu_{m,0} \cdot \prod_{k=1}^{d} b_{m,k}\left((u_k - \mu_{m,k})^2\right)$$

where $b_{m,k}(x) = K_{m,k} e^{-x/2\sigma_{m,k}^2}$ and the α_m are such that:

$$\sum_{m=1}^{M} \alpha_m \prod_{k=1}^{d} b_{m,k}\left((u_k - \mu_{m,k})^2\right) = 1$$

We recall that the estimation of the data's probability distribution led to the following:

$$p(z, u_1, .., u_d) = \sum_{m=1}^{M} w_m b_{m,0}\left((z - \mu_{m,0})^2\right) \prod_{k=1}^{d} b_{m,k}\left((u_k - \mu_{m,k})^2\right),$$

Which, by using Bayes' rule, will lead to the following expression for the conditional probability distribution of z given the input **U** (which we recall are the past d lag values of the same observation in our case).

$$p(z \mid \mathbf{U}) = \frac{\sum_{m=1}^{M} w_m . b_{m,0}\left((z-\mu_{m,0})^2\right) \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right)}{\sum_{m=1}^{M} w_m \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right)}$$

The deterministic approach to the re-synthesis of our data was based on Bayes' least square, which is equivalent to the conditional expectation. This conditional expectation can be derived from the previous expression for the conditional probability distribution, leading to the following expression:

$$E[z \mid \mathbf{U}] = \int_z z\, p(z \mid \mathbf{U})\, dz = \frac{\sum_{m=1}^{M} w_m \mu_{m,0} \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right)}{\sum_{m=1}^{M} w_m \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right)}$$

The last step is to realize that if we were to define the coefficients α_m as:

$$\alpha_m = \frac{w_m}{\sum_{m=1}^{M} w_m \prod_{k=1}^{d} b_{m,k}\left((X_{n-k}-\mu_{m,k})^2\right)},$$

then these coefficients would verify $\sum_{m=1}^{M} \alpha_m \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right) = 1$,

and they would further lead to the same expression for the conditional expectation as the one for the radial basis functions approach:

$$E[z \mid \mathbf{U}] = \sum_{m=1}^{M} \alpha_m \mu_{m,0} \prod_{k=1}^{d} b_{m,k}\left((u_k-\mu_{m,k})^2\right) = f(u_1,..,u_d)$$

This proves that in the case of these separable Gaussians, the cluster-based probability mass function approach turns out to be rigorously equivalent to a RBF approach for which the basis functions match the kernels we've used for the PMF's estimation.

Evaluation

The model's ability to reproduce the original training data is an encouraging result; however, if we step back a little, the amount of computation that was involved and the final size of the resulting model are a very high price to pay for the resynthesis of a quasi-stationary wave form. Besides a conviction that the model's internal mechanism is more likely to reflect the original system's behavior, the ultimate goal of this type of modeling is to be able to generalize this behavior to a wider range of states than the ones observed in the training data.

In order to evaluate the generalization skill of these straight-forward separable Gaussians, let's substitute our training data with a familiar sine wave. We already know a sine wave is a two-dimensional linear system and that, therefore, one could plot its associated prediction surface as a plane in a three-dimensional modeling lag space. Figure 6.6 illustrates the outcome of the modeling of a sine wave through the previous weighted sum of separable Gaussian distributions. The ellipsoids represent the resulting clusters while the surface is a plot of the prediction surface that is implied by the estimated model.

This surface shown in Fig. 6.6 was derived from a grid of inputs $\left(u_1^{(j)}, u_2^{(j)}\right)$. For each point (j) of this grid, we use the estimated probability distribution to derive a predicted value for z as:

$$z^{(j)} = E\left[z \mid u_1^{(j)}, u_2^{(j)}\right] = \int_z z\, p\left(z \mid u_1^{(j)}, u_2^{(j)}\right) dz$$

$$= \frac{\sum_{m=1}^{M} w_m\, \mu_{m,0}\, b_{m,1}\left((u_1^{(j)} - \mu_{m,k})^2\right) b_{m,2}\left((u_2^{(j)} - \mu_{m,k})^2\right)}{\sum_{m=1}^{M} w_m\, b_{m,1}\left((u_1^{(j)} - \mu_{m,k})^2\right) b_{m,2}\left((u_2^{(j)} - \mu_{m,k})^2\right)}$$

where $b_{m,k}(.)$ refers to the same object as previously: $b_{m,k}(x) = K_{m,k} e^{-x/2\sigma_{m,k}^2}$.

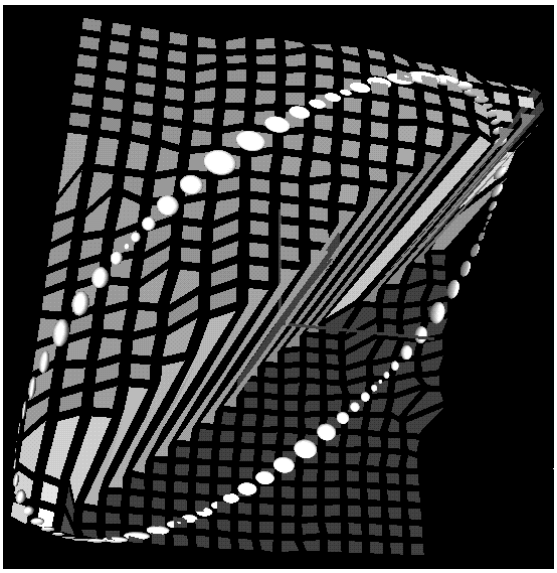


Fig. 6.6 - Visualization of the generalization implied by straightforward separable Gaussians in the simple case of a sine wave. The ellipsoids stand for the clusters that were identified. Each cluster is summarized by a separable Gaussian vector. The surface is the prediction function that is implied when using Bayes' least squares decision rule in conjunction with the cluster-based estimated probability mass function of the data.

Over the expertise of any given cluster, the local description of the prediction surface is reduced to a constant value (we recall that $f(\mathbf{U},\beta_m) = \mu_{m,0} \in \Re$). Therefore, we shouldn't be surprised by the sharp "step" exhibited by our model. We could have anticipated such a behavior by realizing that our separable Gaussian-based model can be considered a "soft" version of a nearest-neighbor lookup. Being limited to a simple mean (or average), the local characterization that we chose doesn't capture any structural information and the resulting model has missed completely the strong linear structure of a simple sine wave.

As a result, initializing our estimated model with any arbitrary state will eventually result in the exact same sine-wave the training was based upon (same amplitude). In other words, we've built a model which will force the system's behavior onto the support of the observation regardless of initial conditions, lacking any type of structural understanding of the original system that was observed.

Cluster-weighted Local Linear Models

In order to overcome the lack of structural understanding that was exhibited by the models that we can derive from the previous choice of separable Gaussian vectors, the first improvement that come to mind is to upgrade our local characterizations from being simple constants $f(\mathbf{U},\beta_m) = \mu_{m,0} \in \Re$ to local linear models:

$$\begin{cases} z = z(t) \\ \mathbf{U} = \mathbf{U}(t) = \left(z(t-\tau),..,z(t-d\tau)\right)^T \\ f\left(\mathbf{U},\beta_m\right) = p_m + \mathbf{q}_m\left(\mathbf{U} - \langle \mathbf{U} \rangle_m\right) \left(\text{where } p_m \in \Re \text{ and } \mathbf{q}_m \in \Re^d\right) \end{cases}$$

From (6.8), the tuning of the parameter set β_m will lead to the possibly familiar-looking linear regression relationships:

$$(6.8) \Rightarrow \begin{cases} p_m \rightarrow p_m^{new} = \langle z \rangle_m \\ \mathbf{q}_m \rightarrow \mathbf{q}_m^{new} = \Lambda_{\mathbf{U},m}^{-1}\left(\langle z\ \mathbf{U} \rangle_m - p_m \langle \mathbf{U} \rangle_m\right) \\ \text{where , } \Lambda_{\mathbf{U},m} = \left\langle \left(\mathbf{U} - \langle \mathbf{U} \rangle_m\right)\left(\mathbf{U} - \langle \mathbf{U} \rangle_m\right)^T \right\rangle_m \end{cases}$$

As a note, we can observe that the update of the parameter set involves the estimation of the input's conditional covariance matrix $\Lambda_{\mathbf{U},m}$, a matrix inversion, and the computation of the conditional input/output cross correlations:

$$\langle z\ \mathbf{U} \rangle_m = \left(\langle z\ u_1 \rangle_m ,..,\langle z\ u_d \rangle_m\right)^T,$$

As for the conditional means $\langle \mathbf{U} \rangle_m = \left(\mu_{m,1},..,\mu_{m,d}\right)^T$, these were already computed for (6.2).

At the time when this document is being written, the implementation of this method and the evaluation of its performance are current research topics for which Bernd Schoner and Professor Gershenfeld are the main investigators.

Chapter Summary

Overcoming non-locality for the estimation of a prediction surface pointed us towards local models. Such models have a better chance to fit the training data with more accuracy. In this chapter, we've considered models for the data's probability distribution, which can be expressed as a weighted sum of a set of local models. Because we felt that the complexity of the resulting model should be a function of the system's complexity rather than a function of the training data size, we have identified the area of expertise of each one of these local models with a notion of "classes of behaviors." This led us to consider non-supervised clustering as a means by which we could determine these sub-areas of the modeling space.

Non-supervised clustering has been addressed in a variety of ways throughout a wide range of fields of study. In a concern for rigor and clarity, we've approached the identification of these classes of behavior without any pre-conception or bias towards a particular method. This led

us to a fairly general scheme which merged the clustering and the modeling tasks in a unifying process which we refer to as *Cluster-weighted Modeling*. This process was derived around a specific form for the model's architecture, which can be seen as a generalization over separable Gaussians. However, we feel that this choice of architecture is general enough to address a very wide range of problems; Gaussian distributions are rarely inappropriate and the "separable" component limits the model's degrees of freedom to a number that is more likely to be manageable by some iterative optimization procedure.

As expected, the "local" component of these models answered our call for accuracy of data-matching. However, by walking through the example of straightforward separable Gaussians, we've illustrated the fact that the same "local" property can miss completely a global structure (even as simple as a striking linearity in the case of a sine wave); leading eventually to a catastrophic generalization of the system's behavior. Of course, the general form for the cluster-weighted models reflects the anticipation of this annoying side-effect by offering a scheme for the estimation of local structures over each one of the clusters' expertise. As we continue to investigate cluster-weighted modeling and investigate the performance of cluster-weighted linear models, we expect to see that the proposed scheme addresses both the issues of data matching and local structure capturing.

There is however a remaining issue that neither the global approach of Chapter 5, nor this general purpose cluster-based local approach have addressed. This issue concerns time scales. While deriving these approaches, we've secretly implied that the observed system was autonomous, or at least that the input vector **U** was made up of measurements that have a similar nature (in the examples we've shown, this input vector was nothing but a lag vector of the same observation). Fitting a prediction surface to the observed data describes short term behaviors for the system but we are entitled to wonder what exactly the meaning of "short term" is when the input measurements which constitute **U** have different natures, and more importantly different time scales of evolution. Another part of the same issue of time scales has to do with the fact that the long-term predictive expertise of our model relies on the faith we have in the accuracy of its short-term expertise. In other words, we can only hope that our model is accurate enough and that the system is regular enough in order to lead to a meaningful long-term behavior.

Even if the resulting model is not to be used for long-term prediction, it already provides a good characterization for the data that was observed. The prior assumptions that this method makes concerning the model's architecture are minor when we compare them to the ones that would be imposed by, let's say, a linear estimation technique. There is no doubt in the author's mind that a wide variety of problems can benefit greatly from the insights that such a characterization provides. Back to the specific context of this work however, musical instruments are only a very limited subset of all possible dynamical systems. Moreover, the long-term behavior of a time-series that was produced by a musical instrument is usually peculiar as it tends to be periodic. Although we've tried very hard to keep our discussion of embedding modeling as general as possible, the author's concern for applicability to musical sound synthesis will lead to a specialization which we will introduce in the following chapter.

Chapter 7

Modeling Strategies / Psymbesis

Musical instruments are only a very restricted subset of all the possible dynamical systems and the scheme that we implied so far by building models in a single lag space is general enough to take extreme behaviors such as chaos in account. As an applicability concern, one way to reduce the current size of this modeling problem is to choose a modeling space which reflects some of the expectations one has about the sound produced by a musical instrument. This state of mind will lead to a particular synthesis technique: Psymbesis (Pitch Synchronous Embedding Synthesis).

Representation / Modeling Space

As a result of the embedding theorem, we saw that considering lag values of observations as the state variables of a model is justified. Assuming that the lag space we build has an appropriate dimension and that the system we're studying is essentially deterministic, we are entitled to characterize the dynamical behavior of our model through the description of a prediction surface. Estimating and describing this prediction surface as a general purpose multi-variable function is exactly what we attempted with either global polynomials or the RBF/PMF approaches that we discussed earlier. Neither of these approaches take the nature of the modeling space in account, namely the fact that it was constructed from successive lag values of the same observations. Indeed, it doesn't seem to matter theoretically whether the system is aware of the modeling space's nature or not, and a concern for generality would even tend to prohibit any constraint in the description of the predictor. If we have faith in the accuracy of the surface's estimation and description, then the desired behavior should emerge naturally from the inferred model whether the original system was linear or not, chaotic or not. In order to overcome non-locality and stability issues, the estimated prediction surface will have to be remarkably accurate. As a result, it seems unlikely that one could get away with any short description of the predictor, and the approach will undoubtedly aim towards very

computationally expensive models. Even then, regardless of the flexibility we allow our predictor to have, it is not clear how we can guarantee the long term behavior of our model.

Some Specifics of Musical Sounds

Confronted with such applicability concerns, this chapter considers some "less general" alternatives for the description of a predictor in our particular context of musical instruments.

Music the way we know it would have never existed without our ability to perceive refined harmonic properties from slight air pressure variations. Although they are not sufficient to explain all the subtleties of musical expression, harmonic relationships and periodicities of wave forms are undoubtedly responsible for most of today's music. Hence, we shouldn't be surprised when we observe that over 90% of an arbitrary musical stream reveals a local quasi-periodic nature. In a lag space of successive sound samples, this is illustrated by peculiar cycles that evolve slowly. For a particular short time period, a snapshot of this cycle (and a scheme for generalization) describes the system's local dynamical behavior the same way a prediction surface would, while it carries some extra information about the long term behavior of the system (it is a cycle).

So far, no clustering or modeling we discussed took these specifics into account. In a concern for generality, we avoided religiously any notion of time or periodicity. This is about to change.

Suggestions and Assumptions

In view of the previous observations, the purpose of this chapter is to introduce *Pitch Synchronous Embedding Synthesis* (or *"Psymbesis"*) as a simplification over the more general approaches that we discussed earlier. The term of "simplification" here doesn't imply that *Psymbesis* is a trivial process, but rather that it is based on a few assumptions concerning musical sounds which will restrict the expertise of the suggested model to sound synthesis. We will now list these assumptions and suggestions. At this point, the author would like to emphasize the fact that the following are not convictions as much as a way to state what is implied by *Psymbesis* as a model for a musical instrument. If these suggestions sound as if they trivialize the nature of a musical instrument, considering alternatives such as FM, wave tables, additive synthesis or sampling should help reduce frustration.

Suggested architecture

Instead of defining the modeling space as a composite of lag values of both control parameters (inputs) and sound samples (output), we hereby suggest to break down the modeling space into two spaces. The first space will be called the *Control Space* and it will stand for the input's (or control's) state space. The second space will be called the *Dynamics Space* and it will be a lag space of sound sample values, just as for an autonomous dynamical system.

H1: Control space = space of perceptually musical gestures

In Chapter 3, we introduced a variety of perceptually meaningful musical gestures and means by which these can be estimated from an audio stream. We are now at the stage where we have to make a decision concerning the nature of the information that will eventually control our virtual instruments. Now that the nature of the instrument's physical interface is dissociated from its internal dynamics, it is the perfect opportunity to forget all about finger boards and mouth pieces. Instead, we will choose our general purpose perceptually meaningful musical gestures to stand for the virtual instrument's control parameters. Of course, nothing guarantees that this set will always be a complete set for any arbitrary instrument that we wish to model. This set is only as complete as the author's imagination could make it; any further extension of this set could very well be a source of future improvements.

The first assumption we'll make for now concerns the completeness of this set as we define the *Control Space* to be the space *(v,p,i,b)* where:

v: volume (or loudness);

p: pitch (or most likely pseudo-period);

i: intercorrelation max (related to a notion of noisiness);

b: brightness (defined as a measurement of energy distribution among the signal's harmonics).

H2: A static state for the instrument corresponds to a cycle in Dynamics Space

The observation of musical sounds in lag spaces reveals interesting looking cycles in slow mutations. Once again, these cycles reveal the quasi-periodic nature of musical sounds. In a lag space of sound samples, the state of a dynamical system can be defined as a vector of successive samples but it doesn't seem to make much sense to define the state of the entire instrument in terms of these fast changing vectors. In agreement with our perception of audio (pitch and harmonic properties), it makes much more sense to define the instantaneous state of an instrument as a snapshot of this slowly evolving cycle in lag space. Therefore, instead of associating a static state for the instrument with a fixed point in a sample lag space, we suggest here to associate it with a description of a *cycle* in lag space (or ***Dynamics Space***).

In most cases, this notion of a *cycle* shouldn't be taken too literally as the system can very well present some noise or other ambiguities that will contribute to the cycle's fuzziness. This cycle

should be understood as the representation of a stationary (in the stochastic sense) system with a strong tendency to periodicity.

H3: A static point in the control space maps to a static state for the instrument

Given the previous definition for an instrument's static state, the third assumption that we're making is that a static point in the control space maps to a static state for the instrument. This is to say that if the control set is frozen in *(v,p,i,b)* then the dynamics of the instrument in *Dynamics Space* should eventually settle down around a *cycle*.

Suggested representation for cycles in Dynamics Space

Let's represent the dynamical behavior of the system in a "pitch synchronous" and "normalized amplitude" way. This should allow a more uniform representation of the dynamics across all the possible states the instrument could be in. This also implies that the notions of pitch and amplitude will have to be "hard-coded" in the synthesis engine as well. In other words, the synthesis engine will have to be explicitly given the current pitch and volume of the control along with the estimated instantaneous state of the instrument in order to produce any sound.

A Complete Representation

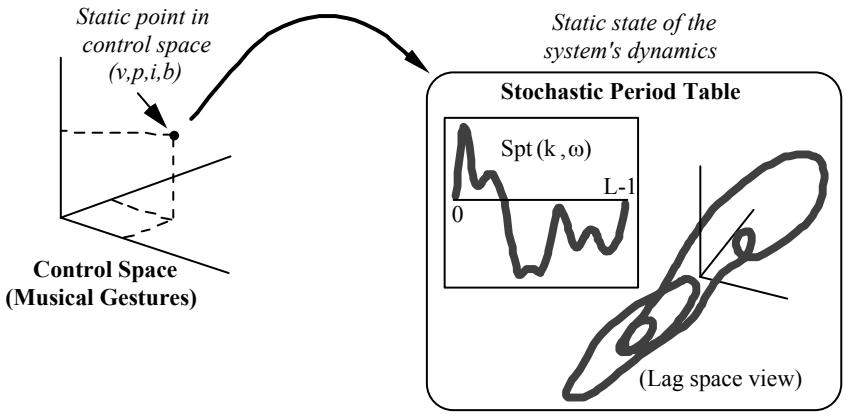


Fig. 7.1 - Suggested representation of a virtual instrument's behavior. (Recall H3: *"A static point in the control space maps to a static state for the instrument"*)

We will refer to ***stochastic period tables*** as the means by which we will represent a ***cycle*** in the dynamics space. A ***cycle*** has a notion of time built in and that is the purpose of these tables. However, the actual arrangement of the tables (beginning and end) don't pretend to carry any information. These tables are of fixed size and fixed amplitude which is in compliance with the "pitch synchronism" and "amplitude normalization" suggested earlier.

In other words, a ***cycle*** $\xi(s,\omega)$ is a stochastic process with periodic means and variances:

$$\exists\, T_\xi \in \Re \;\text{ s.t. }\; \forall\, s \in \Re \;\begin{cases} \sigma_\xi^2(s + T_\xi) = \sigma_\xi^2(s) \\ \text{and} \quad \bar{\xi}(s + T_\xi) = \bar{\xi}(s) \end{cases},$$

and it will be represented by a ***stochastic period table*** $\mathrm{Spt}(k,\omega)$ of fixed length L (i.e. $k \in [0, L-1]$) and normalized amplitude:

$$\xi(s,\omega) \rightarrow \mathrm{Spt}(k,\omega)$$

such that:

$$\begin{cases} E[\mathrm{Spt}(k,\omega)] = G\,\bar{\xi}\left(\text{offset} + \dfrac{kT_\xi}{L}\right) \\ \sigma^2_{\mathrm{Spt}}(k) = G^2\sigma^2_\xi\left(\text{offset} + \dfrac{kT_\xi}{L}\right) \end{cases},$$

where $k \in [0, L-1]$ and $G = \left(\max_s \left|\bar{\xi}(s)\right|\right)^{-1}$ and 'offset' stands for the "beginning" of the table and, as we said before, it doesn't pretend to carry any information relevant to the system's dynamics (it is the result of a fairly arbitrary decision concerning the arrangement of the table).

In order for the synthesis engine to produce any sound, it will have to be fed with the description of a ***cycle***. This will be achieved by providing the corresponding ***stochastic period table*** as well as the corresponding pitch and loudness (which were lost in the representation of the table).

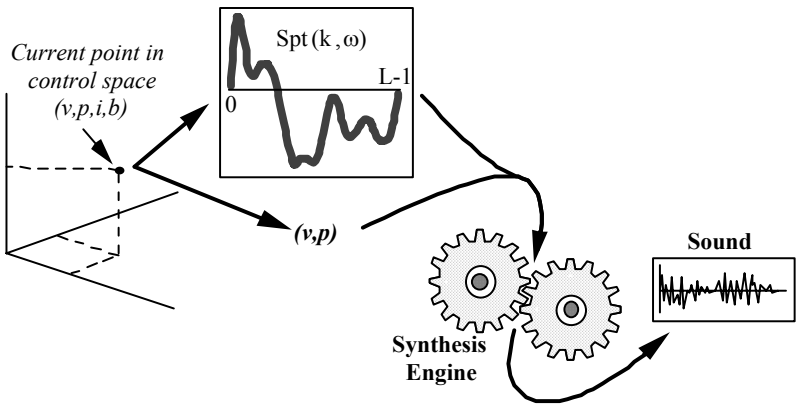


Fig. 7.2 - Control space, stochastic period tables and synthesis engine.

The previous figure illustrates the relationship between the various objects that we've defined in this part: the *control space* of perceptually meaningful *musical gestures*, the representation of the system's dynamics via a *stochastic period table* and finally the *synthesis engine*, which

we will study a little later in this chapter. The next step from here concerns the inference of this representation from training data.

Construction / Model Inferring

The inference of the suggested model falls naturally into three steps. First comes the pre-processing of the data which will lead to the identification of the control training data (musical gestures). Once the whole training set is identified and prepared, the second stage deals with the representation of the *control space*, which we will characterize via some clustering techniques. Finally, the last task concerns the construction of *stochastic period tables* that will be assigned to representative points in the control space.

Training Data

We start with an audio recording *s(t)* of the original instrument. This recording should explore the range of sonic possibilities of the instrument while staying monophonic. This recording is then passed through the various analysis tools described in Chapter 3 (Machine Listening), leading to the estimation of the following musical gestures: volume *v(t)*, pitch *p(t)*, noisiness *i(t)* and brightness *b(t)*.

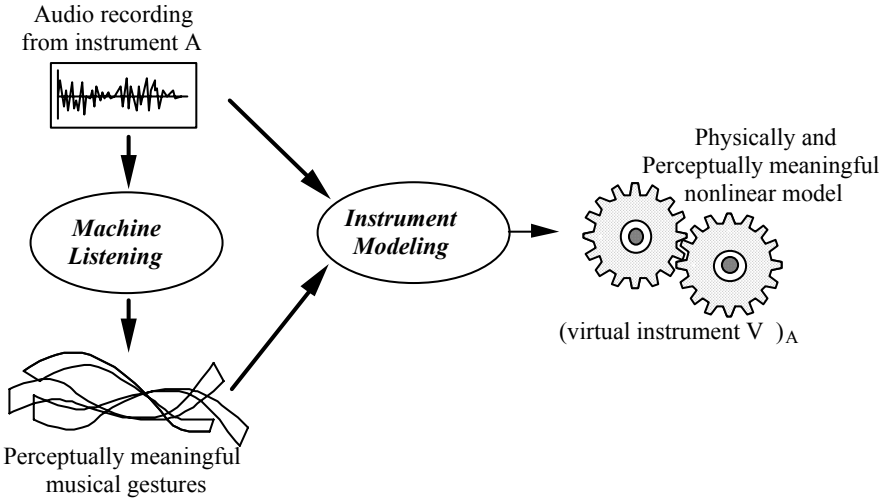


Fig. 7.3 - A synchronous recording of audio and estimated musical gestures constitute the training data from which the model will be inferred.

Our training data consists then of the five synchronous streams: *s(t), v(t), p(t), i(t), b(t)*. The last four streams (*v,p,i,b*) will describe the virtual instrument's control (or input) while *s(t)* obviously is the instrument's output. The model we are trying to infer will map control values to the description of a dynamical behavior, which will eventually lead to sound. As we saw

earlier, this dynamical behavior will be represented by stochastic period tables. These stochastic period tables will be estimated from the stream *s(t)*.

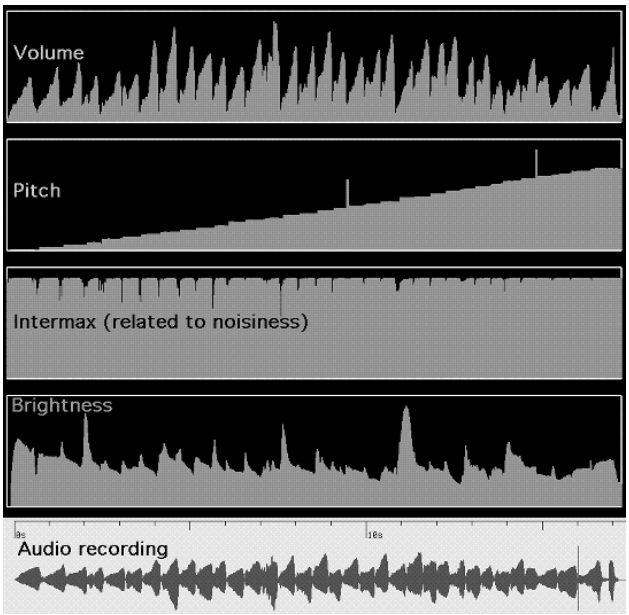


Fig. 7.4 - An example of training data (the original recording is a simple chromatic scale played on a cello)

Control Space (*v, p, i, b*)

The derived streams *(v,p,i,b)*, along with the audio recording characterize what will soon become the control space of our virtual instrument. An elegant representation of the control space should only be a function of the instrument's behavior and not a function of the size of the training data. This points us already towards the idea of clustering as a way to "summarize" the control part of our training data.

The wish for data summary is not the only reason for clustering the control space. It is also motivated by the fact that the only way for us to measure statistics (for the associated cycles) is by averaging observations. Therefore, we clearly need to identify representative classes of control to which multiple observations from our training set can be assigned. Only then will we be able to estimate our ***cycles*** (or stochastic period tables).

Clustering in a normalized space

The coordinates of the control space ***(v,p,i,b)*** have dramatically different natures and therefore they will not share the same range of values. The pitch ***p*** will typically span a range of values between 30 and 80 or so, while the maximum intercorrelation ***i*** will be restricted between 0.5 and 1.0 and the brightness ***b*** will wander in a 3.0 to 15.0 range. Clustering usually implies some metrics over the space and if we decided to use a Euclidean distance, these heterogeneous ranges of values would bias the resulting clusters to favor their discrimination

based on a particular coordinate over the others. In order to overcome this undesired phenomenon, we should consider normalizing the control space accordingly to the range of each one of the coordinates. This process will lead to the alternative control space of elements **x** defined as follows:

$$\mathbf{x} = \begin{vmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{vmatrix} = \begin{vmatrix} (v - v_{min})/(v_{max} - v_{min}) \\ (p - p_{min})/(p_{max} - p_{min}) \\ (i - i_{min})/(i_{max} - i_{min}) \\ (b - b_{min})/(b_{max} - b_{min}) \end{vmatrix} \in [0,1]^4$$

In this normalized space, we can then think of applying some standard clustering techniques such as K-means (hard clustering) or Melting.

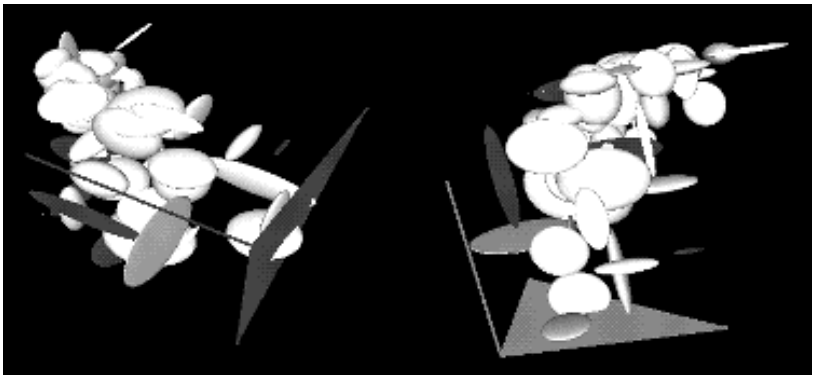


Fig. 7.5 - Two views of the outcome of a K-means hard clustering procedure applied to the normalized control space (views in ***(v,p,i)*** space).

The centroids of the estimated clusters will provide us with representative points in the control space, implying a summary for our control training data.

Summary of the control space

Either through hard or soft assignment of the training data to the estimated clusters (or classes), we can compute the means and variances for each one of these classes. The probability mass function of the normalized control data can then be summarized as:

$$p(x_1, x_2, x_3, x_4) = \sum_{m=1}^{M} w_m \prod_{k=1}^{4} K_{m,k}\, e^{-(x_k - \mu_{m,k})^2/(2\sigma_{m,k}^2)}$$

where M is the number of *representative* points (i.e. clusters or classes' centroids), $w_m>0$, and $\sum_{m=1}^{M} w_m = 1.$

Of course, this expression is not unique. Among other things, it implies that each cluster is modeled as a separable Gaussian distribution, which is also to say that we consider the various coordinates of the space to be uncorrelated. Given the nature of the space ***(v,p,i,b)***, this last assumption shouldn't be too far fetched.

Stochastic Period Tables

The final stage of the inference of the model consists in associating a single *stochastic period table* with each one of the representative points of the control space. These will be responsible for the description of the dynamical behavior of the virtual instrument when it is under the different classes of control represented by the centroids of our clusters.

Let's consider a single class (or cluster) in the control space and let the following be the set of data that belong to that class:

$$\mathbf{e}^{(n)} = \left(v^{(n)}, p^{(n)}, i^{(n)}, b^{(n)}, \tau^{(n)}\right)^T ,$$

where $n \in [0, N-1]$ and $\tau^{(n)}$ stands for the time pointer in the original sound stream s(t) that corresponds to this element. Note that we're implying hard cluster assignment here. Although hard clustering in not required for the estimation of the stochastic period tables, it will simplify our notations and illustrate the process in clearer terms. Once a clear understanding of this approach is acquired, an eventual generalization to a "softer" clustering scheme will be trivial.

Pseudo-periods extraction from the original sound

Let's recall that the ultimate goal is to build a pitch synchronous (fixed length L), normalized, stochastic representation of the cycle associated with that particular control class. The corresponding stochastic period table may result from some averaging over the observed pseudo-periods that the sound s(t) exhibited at the times $\tau^{(n)}$.

For each element $\mathbf{e}^{(n)}$ the corresponding pseudo-period table $\mathrm{Ppt}^{(n)}(k)$ ($k \in [0, L-1]$) is a re-sampled and normalized chunk of signal that we extract from the original s(t):

$$\mathrm{Ppt}^{(n)}(k) = G.s\left(\tau^{(n)} + \frac{kT^{(n)}}{L}\right),$$

where $T^{(n)}$ is the quasi-period $\left(T^{(n)} \propto 2^{-p^{(n)}/12}\right)$ and $G = \left(\max_{0<t<T^{(n)}} \left|s(\tau^{(n)} + t)\right|\right)^{-1}$.

Before we go ahead and average these normalized and pitch synchronous tables, there is one last trap we need to avoid. This is now the third time that we insist on the fact that the actual alignment of such tables (beginning and end) are arbitrary. The information they carry

concerns cyclic behaviors which have no beginning or end. Therefore any averaging or comparison we will apply to these tables shouldn't be influenced by the actual "alignment" of the tables. As an illustration, the following figure plots three pseudo-period tables which describe the exact same cyclic behavior while being out of alignment.

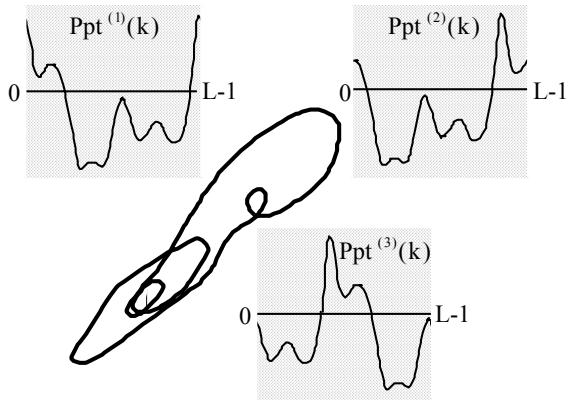


Fig. 7.6 - Example of three peuso-period tables $\mathrm{Ppt}^{(1)}(k)$, $\mathrm{Ppt}^{(2)}(k)$ and $\mathrm{Ppt}^{(3)}(k)$ that map to the same trajectory in a lag space.

Re-aligning the estimated pseudo-period tables with respect to each other is therefore an essential step.

Aligning and averaging

In order to align these tables with respect to each other, we will pick a reference table $\mathrm{Ppt}^{(\mathrm{REF})}(k)$ among all the $\mathrm{Ppt}^{(n)}(k)$ we extracted and align all the others with respect to that reference. The alignment procedure itself results from the following minimization:

$$\mathrm{shift}^{(n)} = \arg\min_{k} \sum_{l=0}^{L-1} \left\| \mathbf{u}^{(\mathrm{REF})}(l) - \mathbf{u}^{(n)}(k+l) \right\|^2 ,$$

where $\mathbf{u}^{(n)}(l) = \left(\mathrm{Ppt}^{(n)}(l_{\mathrm{mod}\ L}), \ldots, \mathrm{Ppt}^{(n)}((l+d-1)_{\mathrm{mod}\ L})\right)^T$ are lag vectors of dimension d that were built by reading $\mathrm{Ppt}^{(n)}(k)$ circularly. This is to say that we wish to minimize the average point-to-point distance in the lag space of dynamics. Of course, it doesn't take much calculus to realize the following:

$$\forall k,\ \sum_{l=0}^{L-1} \left\| \mathbf{u}^{(n)}(k+l) \right\|^2 = \sum_{l=0}^{L-1} \left\| \mathbf{u}^{(n)}(l) \right\|^2$$

and

$$\sum_{l=0}^{L-1}\left(\mathbf{u}^{(REF)}(l)\right)^{T}\mathbf{u}^{(n)}(k+l) = d\ \sum_{l=0}^{L-1}Ppt^{(REF)}(l_{\bmod L})\ Ppt^{(n)}\left((k+l)_{\bmod L}\right)$$

Therefore, the shift that we need to apply to the pseudo-period table $Ppt^{(n)}(k)$ will be given by:

$$shift^{(n)} = \arg\max_{k}\sum_{l=0}^{L-1}Ppt^{(REF)}(l_{\bmod L})\ Ppt^{(n)}\left((k+l)_{\bmod L}\right),$$

which invites us once again to find the maximum of a cross-correlation.

Once our extracted pseudo-period tables are aligned with respect to each other, we are finally ready to estimate the statistics of the stochastic period table associated with the current class 'Cl'. For instance, one way to do so would be through the following averaging:

$$\text{For } k \in [0, L-1],\ \begin{cases} E\left[Spt_{Cl}(k,\omega)\right] = \dfrac{1}{N}\displaystyle\sum_{n=0}^{N-1}Ppt^{(n)}(k) \\ \text{and}\quad \sigma^2_{Spt_{Cl}}(k) = \dfrac{1}{N}\displaystyle\sum_{n=0}^{N-1}\left(Ppt^{(n)}(k) - E\left[Spt_{Cl}(k,\omega)\right]\right)^2 \end{cases}$$

Synthesis Engine

Assuming that we possess a thorough description or characterization of a virtual instrument, the question remains as to how to produce an actual sound stream from it. This will be the task of the synthesis engine which we are about to describe. From the previous choices we made concerning the virtual instrument's characterization, we'll derive the model that is implied for this instrument. This model will be an expression of the probability distribution of sound samples. Choosing and applying a particular decision rule to this model will describe our full-blown synthesis engine.

General Form

Writing an explicit form for the samples' probability distribution is most likely more than we actually need in order to infer the mechanism of our synthesis engine. Depending on which decision rule we will decide to use, some of the choices that we are about to make may end up not influencing the outcome whatsoever. However, thinking about a model in terms of an expression for the data's probability distribution is something that we have done repeatedly in the context of this document and we found that it provides a clear and universal ground for thinking about modeling. Furthermore, in addition to providing some consistency with this document's previous chapters, the following derivation is the perfect opportunity to collect the

notions of cycles and state, as well as to introduce some other notions (such as phase), in a rigorous and hopefully clear package.

Representation

In the tradition of sound synthesis, a ***voice*** usually stands for a monophonic layer of sound. In our case, a ***voice*** will be a dynamical system with its own internal state. z(t) is its output (sound samples) and its internal state is represented as a lag space of dimension d (embedding dimension) by the following vector:

$$\mathbf{U}(t,\tau) = \left(Z(t-\tau), Z(t-2\tau), .., Z(t-d\tau)\right)^T$$

As we implied in what precedes, a ***cycle*** will stand for the representation of the voice's current stationary state. A cycle $\xi(s,\omega)$ is a stochastic process with periodic means and variances. Furthermore, having only access to second order statistics, we will simply adopt a Gaussian form for these cycles:

$$p_{\xi(s)}(X) = \frac{1}{\sqrt{2\pi\sigma_\xi^2(s)}} \exp\left(-\frac{\left(X-\bar{\xi}(s)\right)^2}{2\sigma_\xi^2(s)}\right)$$

and $$\exists\, T_\xi \in \Re \;\; s.t. \;\; \forall\, s \in \Re \;\; \begin{cases} \sigma_\xi^2(s+T_\xi) = \sigma_\xi^2(s) \\ \text{and} \quad \bar{\xi}(s+T_\xi) = \bar{\xi}(s) \end{cases}$$

In order to synthesize (or predict) a new sound sample z(t), the synthesis engine will have to be given both the current dynamical state of the voice $\mathbf{U}(t,\tau)$ and the stationary state of the instrument $\xi(s,\omega)$.

Suggested Architecture

Given $\mathbf{U}(t,\tau)$ and $\xi(s,\omega)$, we need to predict z(t). It is now time to explicitly write a fairly general form for our suggested model as a conditional probability distribution.

In order to do so, we need to introduce an additional object: the notion of ***phase*** in a cycle. We recall that we are not building a wave table-type synthesis engine or a sampler and although the ***phase*** φ corresponds to some pointer in the ***cycle*** $\xi(s,\omega)$, it is not governed by any external oscillator or rule (such as φ^{γ} constant in the case of sampling). The phase φ is another random variable that we need to introduce in order to write explicitly our probabilistic model.

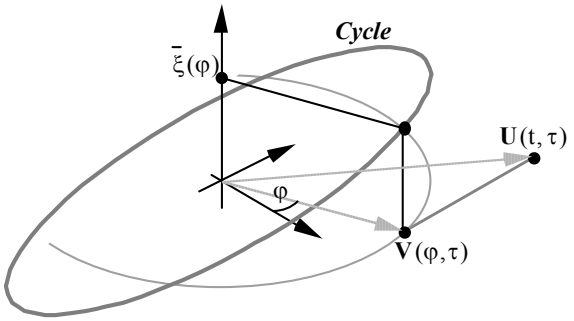
$$p\left(z(t)|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right)=\int_{\varphi=0}^{\varphi=T_\xi} p\left(z(t),\varphi|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right)d\varphi$$

$$=\int_{\varphi=0}^{\varphi=T_\xi} p\left(z(t)|\ \varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)p\left(\varphi|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right)d\varphi$$

Given the current dynamical state $\mathbf{U}(t,\tau)$ of the voice and the description $\xi(s,\omega)$ ($s\in[0,T_\xi]$) of the current cycle, the probability distribution of the current phase φ is related to the distance between the two d-dimensional vectors:

$$\mathbf{U}(t,\tau)=\left(Z(t-\tau),Z(t-2\tau),..,Z(t-d\tau)\right)^T$$

and $$\mathbf{V}(\varphi,\tau)=\left(\bar{\xi}(\varphi-\tau),\bar{\xi}(\varphi-2\tau),..,\bar{\xi}(\varphi-d\tau)\right)^T.$$



As a reasonable expression for the conditional probability distribution of this phase, we suggest the following:

$$p\left(\varphi|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right)=b_{\mathbf{U},\xi}\left(\left\|\mathbf{U}(t,\tau)-\mathbf{V}(\varphi,\tau)\right\|^2\right)=b_{\mathbf{U},\xi}\left(\sum_{n=1}^{d}\left(Z(t-n\tau)-\bar{\xi}(\varphi-n\tau)\right)^2\right),$$

where $b_{\mathbf{U},\xi}(.)\in[0,1]^{\Re_+}$ is some decreasing function such that:

$$\int_{\varphi=0}^{\varphi=T_\xi} p\left(\varphi|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right)d\varphi=1$$

Given the state $\mathbf{U}(t,\tau)$, the cycle $\xi(s,\omega)$ and the phase $\varphi\in[0,T_\xi]$, we finally suggest the following form for the conditional probability distribution of the new sound sample z(t):

$$p\left(z(t)|\ \varphi, \mathbf{U}(t,\tau), \xi(s,\omega)\right) = \frac{1}{\sqrt{2\pi\sigma_\xi^2(\varphi)}} \exp\left(-\frac{\left(z(t) - g\left(\varphi, \mathbf{U}(t,\tau), \xi(s,\omega)\right)\right)^2}{2\sigma_\xi^2(\varphi)} \right),$$

which is to say that this conditional is a Gaussian random variable with mean $g\left(\varphi, \mathbf{U}(t,\tau), \xi(s,\omega)\right)$ and variance $\sigma_\xi^2(\varphi)$.

The function g(.) is the means by which we will allow our model to generalize outside the support of the cycle $\xi(s,\omega)$.

Choices and Issues

From what precedes, the model we're suggesting has the following general form:

$$p\left(z(t)|\ \mathbf{U}(t,\tau), \xi(s,\omega)\right) = \int_{\varphi=0}^{\varphi=T_\xi} \frac{1}{\sqrt{2\pi\sigma_\xi^2(\varphi)}} \exp\left(-\frac{\left(z(t) - g\left(\varphi, \mathbf{U}(t,\tau), \xi(s,\omega)\right)\right)^2}{2\sigma_\xi^2(\varphi)} \right) b_{\mathbf{U},\xi}\left(\sum_{n=1}^{d} \left(Z(t - n\tau) - \bar{\xi}(\varphi - n\tau) \right)^2 \right) d\varphi$$

In order to make a decision concerning a particular instance of z(t), we can use this conditional probability distribution and compute the most likely value for z(t). This would be a "deterministic" approach. We can also think of producing an instance of z(t) from a random number generator that shares the same probability distribution. This would be a "stochastic" approach.

But before we start implementing either one of these approaches, we will need to make some decision concerning the form of the functions $b_{\mathbf{U},\xi}(.) \in [0,1]^{\Re_+}$,which describes the distribution of the ***phase*** , and g(.), which describes the means by which we generalize the ***cycle*** $\xi(s,\omega)$ to the entire ***dynamics space***.

Phase

In the context of this work, the previous full-blown expression for the conditional probability distribution of z(t) given the present state of the voice and the description of the cycle awakes some applicability concerns. Any choice concerning $b_{\mathbf{U},\xi}(.) \in [0,1]^{\Re_+}$ might be fairly arbitrary and as we will see later on, particular choices concerning the probability distribution of this phase may simplify this expression greatly.

Generalization in the Dynamics Space

We recall that $\xi(s,\omega)$ describes the dynamical behavior of the instrument in dynamics space only for a very restricted subset of the whole state space. In order to extend its prediction ability to the entire dynamics space, one needs a generalization scheme. Only then can we consider the cycle $\xi(s,\omega)$ to be the description of a prediction surface. In the previous

expression of the conditional probability distribution, we introduced the function g(.) which task is to achieve this generalization.

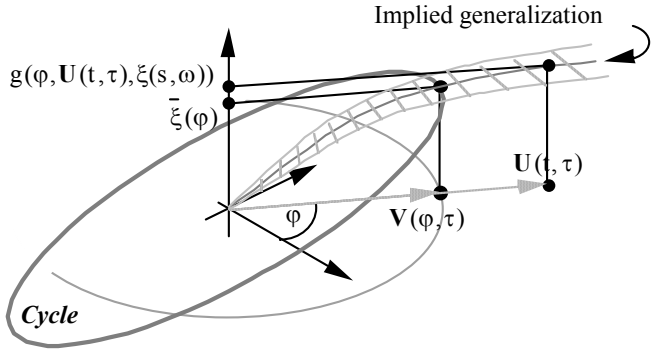


Fig. 7.7 - Illustration of a typical desirable generalization scheme. The cycle implies a description of a prediction surface via a radial generalization scheme. The figure illustrates a one sample prediction given an arbitrary state $\mathbf{U}(t,\tau)$ with respect to the closest state $\mathbf{V}(\varphi,\tau)$ that lives on the cycle. Other generalization schemes can be used and this figure is only an illustration of the basic ideas underpinning such a generalization.

Before suggesting any specific form for this generalizing function, let's list a few postulates it should verify.

(i) If the current state $\mathbf{U}(t,\tau)$ of the voice belongs to the support of the cycle $\xi(s,\omega)$ then we should find a phase φ such that $g\left(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)=\bar{\xi}(\varphi)$. In this case, one can find φ such that $\mathbf{U}(t,\tau)=\left(Z(t-\tau),Z(t-2\tau),..,Z(t-d\tau)\right)^T$ and $\mathbf{V}(\varphi,\tau)=\left(\bar{\xi}(\varphi-\tau),\bar{\xi}(\varphi-2\tau),..,\bar{\xi}(\varphi-d\tau)\right)^T$ are identical. This is to say that this function shouldn't have any influence when there is no need for generalization.

(ii) If there is a phase such that $\mathbf{U}(t,\tau)=\lambda.\mathbf{V}(\varphi,\tau)$ where $\lambda>1$ (i.e. the current state of the voice is wandering "outside" the cycle), then we should have $g\left(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)<\lambda.\bar{\xi}(\varphi)$. This is to say that the generalization scheme should attempt to pull the system's state towards the cycle. Let's recall that a linear generalization would imply $g\left(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)=\lambda.\bar{\xi}(\varphi)$.

(iii) Similarly, if there is a phase such that $\mathbf{U}(t,\tau)=\lambda.\mathbf{V}(\varphi,\tau)$ where $0<\lambda<1$ (i.e. the current state of the voice is wandering "inside" the cycle), then we should have $g\left(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)>\lambda.\bar{\xi}(\varphi)$.

Given these three postulates, a boundless variety of forms can be suggested for the function g(.) but we will hereby suggest one of the simplest forms, namely the following:

$$g\left(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)\right)=\left(\alpha+(1-\alpha)\frac{\mathbf{U}^T(t,\tau)\mathbf{V}(\varphi,\tau)}{\left|\mathbf{V}(\varphi,\tau)\right|^2}\right)\bar{\xi}(\varphi)\text{ , where }\begin{cases}0<\alpha<1\\ \mathbf{U}(t,\tau)=\left(Z(t-\tau),Z(t-2\tau),..,Z(t-d\tau)\right)^T\\ \mathbf{V}(\varphi,\tau)=\left(\bar{\xi}(\varphi-\tau),\bar{\xi}(\varphi-2\tau),..,\bar{\xi}(\varphi-d\tau)\right)^T\end{cases}$$

The parameter α will determine how fast the system will be pulled towards the cycle. The following illustration plots g(.) as a function of the normalized scalar product between the two lag vectors **U** and **V** for two different choices concerning α.

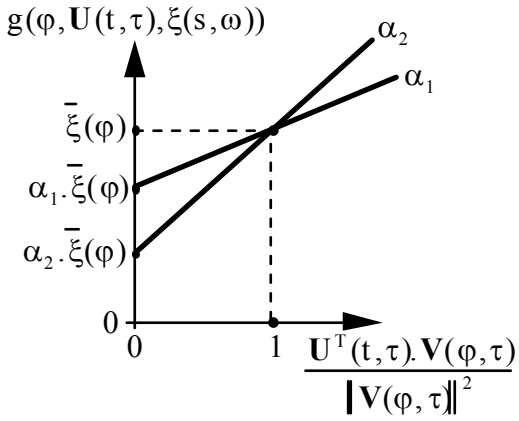


Fig. 7.8 - Suggested form for g(.) as a function of a normalized scalar product for two different values for α ($0 < \alpha_2 < \alpha_1 < 1$)

Taken as a function of this normalized scalar product, it is trivial to verify that such a form for g(.) verifies our three postulates given that α stays in the range between 0 and 1. The extreme case where α=0 corresponds to some sort of "radial linear" generalization which doesn't favor the cycle at all. The other extreme case where α=1 turns the cycle into a very strong attractor, forcing the dynamical state of the voice to follow that cycle almost immediately.

Virtual Instruments

By now, we have gathered most of the pieces that will constitute a virtual instrument. On the one hand, we have means by which we can describe classes of control and associated description of the dynamics; on the other hand we have an architecture for a synthesis engine which will use a description of some dynamics in order to produce an actual sound stream given some specific control. In what follows, we will list explicitly the set of objects which constitute a virtual instrument. We will then put these new notions to the test of the simplest possible interpretation.

Portrait of a Virtual Instrument

Much like its physical cousin, a virtual instrument's mission is to turn musical gestures into a sound stream. Working from this "black box" point of view and in the light of the assumptions and architectures that we've described so far, it is time to collect all the necessary objects that will constitute a full-blown virtual instrument.

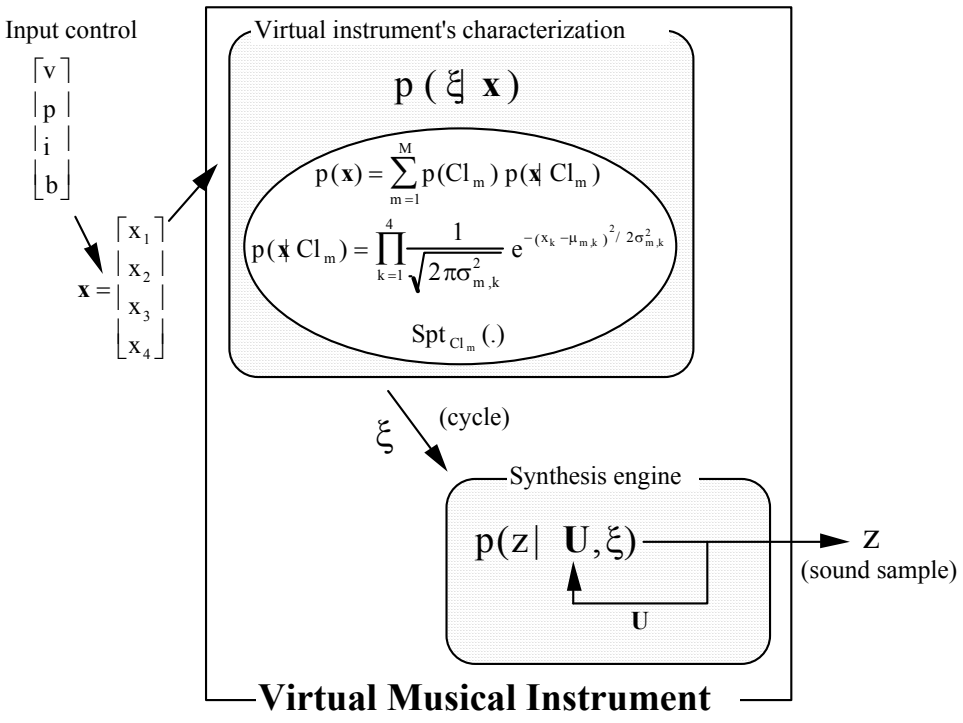


Fig. 7.9 - Portrait of a virtual instrument.

The major part of the virtual instrument's characterization resides in the set of clusters Cl_m that span the control space along with their associated stochastic period tables. As we implied earlier, each one of these clusters is taken as a "class of control" under which the dynamics of the system is described by a *cycle* for which the stochastic period table is an internal representation. After Chapter 6, we know that there is only a fine line between clustering and modeling and the inference of these clusters and cycles could be re-interpreted to characterize the probability distribution of the dynamics (i.e. the cycle) given the input control (i.e. normalized musical gestures **x**): $p(\xi | \mathbf{x})$.

Given the description (or at least a characterization) of this conditional distribution, a decision rule can be applied in order to estimate an appropriate cycle which will describe the instrument's dynamics. This cycle is then fed to the synthesis engine in order to produce some sound samples.

In Figure 7.9, as well as throughout the whole chapter, we've implied that the "instrument's characterization" and the "synthesis engine" were two separate entities. The author chose to do so for two obvious reasons. The first is the fact that it provides a more progressive and clearer view concerning the ideas underpinning *Psymbesis* as a modeling and synthesis method. The second reason is more pragmatic: it has to do with the modular and incremental coding of the method under these circumstances. Breaking the system into two fairly independent modules reduces the complexity of the model inference and the amount of computation required for the synthesis.

Example of an Interpretation and Implementation

Although we've been fairly specific in our description of notions such as *"control classes"*, *"phase"*, *"cycle"* and *"generalization"*, the scheme that we've presented so far is still open to various interpretations (and choices) which could lead to various implementations and techniques. As a way to evaluate it, we're about to describe a set of choices which could stand for the simplest possible interpretation of *Psymbesis*.

Classes of Control

Each class of control in the input space ***(v,p,i,b)*** is associated with a description of the system's dynamics via a stochastic period table. As we implied earlier, these classes will most likely result from some non-supervised clustering technique. We would be entitled to question the role played by the actual "pseudo-period tables" in this process of identifying these classes. Indeed, in the light of what we've discussed in Chapter 6, clustering should be seen as a modeling process and this process should probably take these periods into account if we are to assign stochastic period table to each of these classes.

In the context of this simplest interpretation however, we might be tempted to cut some corners for the sake of simplicity and clarity. Therefore, we suggest the use of a fairly general clustering technique which will only take the control values ***(v,p,i,b)*** (or rather their normalized versions **x**) in account. More specifically, we'll use a soft clustering technique known as "melting". This clustering technique is discussed in details in [Won92].

Control and Cycles

Keeping the control and the synthesis engine as two separate modules, we propose to infer the description of the current cycle based uniquely on the control parameters **x** (or ***(v,p,i,b)***). The clusters inferred in the control space of the instrument stand for "classes of control" and are associated with specific cycles. One way to infer the current cycle would be to estimate the most likely class of control the parameters **x** lead to, and to assign the appropriate description of a cycle based on this class. This is exactly what we are about to do in the context of this implementation:

Given an input set x, we will assign a cycle ξ which is derived from the representation $\mathrm{Spt}_{\mathrm{Cl}(\mathbf{x})}(u,\omega)$,

where $\mathrm{Cl}(\mathbf{x}) = \arg\max_{\mathrm{Cl}} p(\mathrm{Cl}|\ \mathbf{x}) = \arg\max_{\mathrm{Cl}} p(\mathrm{Cl})p(\mathbf{x}|\ \mathrm{Cl})$

Choice of the Phase's Probability Distribution

We recall the general model implied by the synthesis engine we've described:

$$p(z(t)|\ \mathbf{U}(t,\tau),\xi(s,\omega)) = \int_{\varphi=0}^{\varphi=T_\xi} \frac{1}{\sqrt{2\pi\sigma_\xi^2(\varphi)}} \exp\left(-\frac{(z(t)-g(\varphi,\mathbf{U}(t,\tau),\xi(s,\omega)))^2}{2\sigma_\xi^2(\varphi)} \right)\ b_{\mathbf{U},\xi}\left(\sum_{n=1}^{d} \left(Z(t-n\tau) - \bar{\xi}(\varphi-n\tau) \right)^2 \right) d\varphi$$

At this point, our main concern might be to simplify the model's expression as much as possible. What we're about to suggest here for the probability distribution of the phase may sound a little extreme but by no means do we commit ourselves to the simplifications it involves. We'll just consider the conditional probability distribution of the phase to be a Dirac distribution:

$$p\left(\varphi \middle|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right) = b_{\mathbf{U},\xi}\left(\left\|\mathbf{U}(t,\tau)-\mathbf{V}(\varphi,\tau)\right\|^2\right) = \delta\left(\varphi-\varphi_{ML}(\mathbf{U},\xi)\right)$$

In the previous expression, $\varphi_{ML}(\mathbf{U},\xi)$ is obviously the most likely value of the phase given the voice's state **U** and the cycle $\xi(s,\omega)$. This Dirac distribution will obviously allow us to get rid of the integral in the expression of the conditional of z(t) and we end up with the following:

$$p\left(z(t)\middle|\ \mathbf{U}(t,\tau),\xi(s,\omega)\right) = \frac{1}{\sqrt{2\pi\sigma_\xi^2\left(\varphi_{ML}(\mathbf{U},\xi)\right)}}\exp\left(-\frac{\left(z(t)-g\left(\varphi_{ML}(\mathbf{U},\xi),\mathbf{U}(t,\tau),\xi(s,\omega)\right)\right)^2}{2\sigma_\xi^2\left(\varphi_{ML}(\mathbf{U},\xi)\right)}\right)$$

$$\text{where}\quad \varphi_{ML}(\mathbf{U},\xi) = \arg\min_{\varphi}\left(\sum_{n=1}^{d}\left(Z(t-n\tau)-\bar{\xi}(\varphi-n\tau)\right)^2\right)$$

Resulting prediction surfaces

Given this choice concerning the probability distribution of the phase and the previous form of the generalizing function g(.), we can now implement the synthesis engine and view its implied predictions surfaces. In order to visualize any prediction surface, we have to pick a system that will appear to be deterministic in a lag space of dimension two. Let's revisit once again the very particular case of a sine wave.

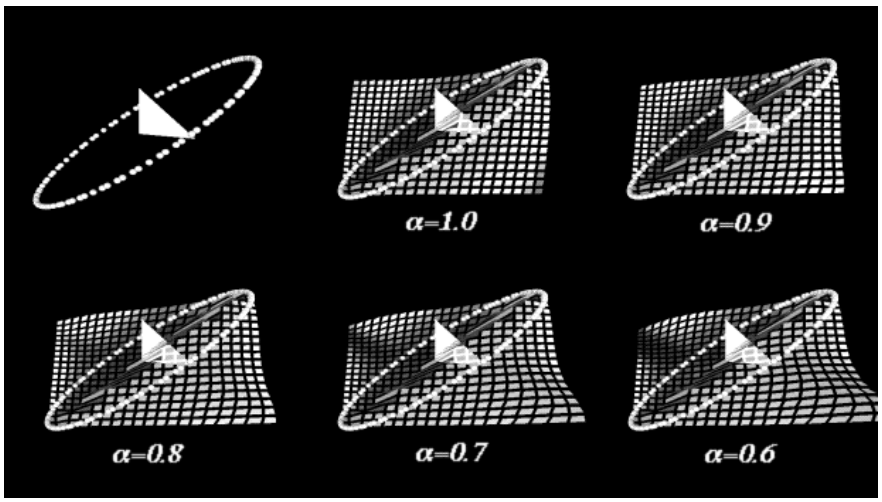
The cycle $\xi(s,\omega)$ will be represented by a stochastic period table that exhibits a period of a sine-wave. In a three-dimensional lag space, this cycle is a simple ellipse. In order to observe the prediction surface that is implied by the synthesis engine, we will feed the engine with a set of internal states $\mathbf{U}^{(i)}$ which span a two-dimensional grid:

$$\mathbf{U}^{(i)} = \left(Z^{(i)}(t-\tau),Z^{(i)}(t-2\tau)\right)^T \quad \in [-1,1]^2$$

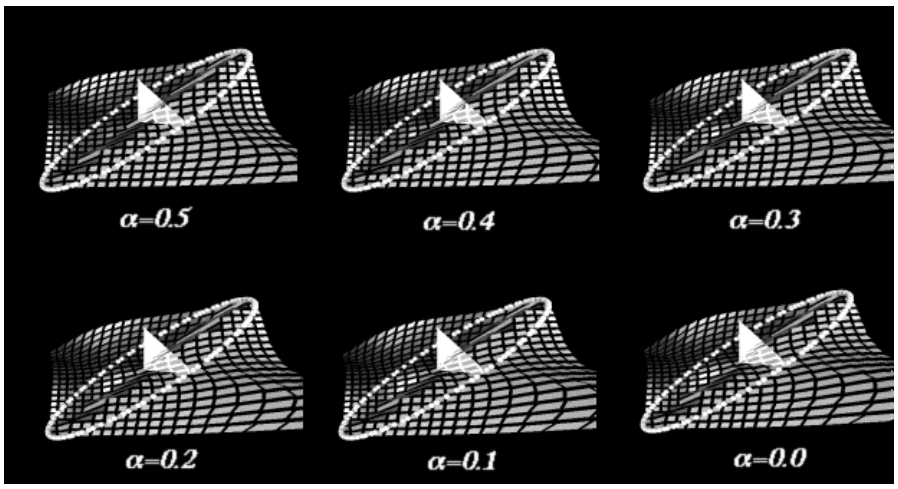
For each one of these internal states, the synthesis engine will compute the most likely phase $\varphi_{ML}^{(i)}$. Finally, in sight of the form of the conditional probability of z(t), the most likely value for z(t) (deterministic approach) will be given by:

$$E\left[z^{(i)}(t)\middle|\mathbf{U}^{(i)},\xi\right] = g\left(\varphi_{ML}^{(i)},\mathbf{U}^{(i)}(t,\tau),\xi(s,\omega)\right) = \left(\alpha+(1-\alpha)\frac{\mathbf{U}^{(i)T}(t,\tau)\mathbf{V}\left(\varphi_{ML}^{(i)},\tau\right)}{\left\|\mathbf{V}\left(\varphi_{ML}^{(i)},\tau\right)\right\|^2}\right)\bar{\xi}\left(\varphi_{ML}^{(i)}\right)$$

The following plots illustrate the various shapes of the resulting prediction surface associated with different choices concerning the value of α, which we introduced earlier.



For large values for α (i.e. α<=1.0), the synthesis engine will provide the "closest guess" based on what it sees in the description of the cycle. It won't extrapolate much outside the support of this cycle, strongly forcing the dynamical state of the system onto that support. As α decreases, a more meaningful generalization emerges from the synthesis engine's behavior.



One could very well imagine that the appropriate value for this generalization parameter α could be estimated from the original training data while the stochastic period tables are being estimated for each cluster in the control space. Once again, we'll go for the simplest interpretation of this modeling scheme and assume that α is constant throughout the entire model.

Application to a Virtual Cello

As an instrument to model, the author chose an electric cello (built in Canada by RAAD) which had been used previously at the Media Laboratory. The training data was taken to be the audio recording of a chromatic scale played inexpressibly on purpose. This ten second audio file was then passed through the machine listening tools which we discussed in Chapter 3. Loudness, pitch, noisiness and brightness were estimated at 10 millisecond intervals on the whole length of the recording. The resulting sequence of time-stamped musical gestures was then clustered using a melting technique inspired by [Won92].

For the assignment of stochastic period tables to each one of the identified clusters, we decided to try either an averaging technique (which was implied earlier), or an assignment based on the identification of a reference within each cluster. The later technique would simply consist of finding the training data point that is the closest to the cluster's centroid, and assigning the pseudo-period associated with that point to the mean of the stochastic period table.

Once the virtual cello is inferred (approximately 5 to 10 minutes), an extra set of control data (sequence of musical gestures) is needed in order to play the instrument and evaluate the result. A short melody was played on the same cello, but this time with much more expressiveness (with vibrato, envelope and other timbral modulations), by Ben Denckla, and recorded in another sound file. This sound file was then processed through the same musical gesture analysis in order to provide a sequence of input controls. The audio recording of this melody was discarded and this new set of musical gestures was finally fed to our virtual cello in order to produce a synthetic audio file of the melody. The resulting sound file exhibits a surprising degree of expressiveness considering that the training data was nothing but a ten second recording of a dull chromatic scale. As a further amusing exercise, we took advantage of the musical gestures representation of the melody by harmonizing the musical stream and adding layers. In order to edit freely a set of musical gestures prior to feeding it to our virtual instrument, a special editor had to be built. Armed with this graphic tool, one can literally sculpt the control sequence of the instrument and stress the inferred dynamics to their limits.

Strengths

At a macro level, the first striking result is the degree of expressiveness that this virtual instrument exhibits. Not only was the training data small, but the inference of the entire instrument was done in the absence of any human supervision. Our virtual cello exhibits subtle an smooth timbre variations which one can control arbitrarily with our perceptually meaningful musical gestures. This potential degree of expressiveness goes obviously to the credit of our control's nature and to the choice of a pitch-synchronous and normalized internal representation of the system's dynamics. Indeed, this choice of a representation enables the virtual instrument to generalize the instrument's timbre variations along a continuous range of pitches and loudness. Such a generalization in the control space becomes especially handy to the synthesis engine when it's required to function under a set of control that is distant from any class of control that was identified within the training data.

On a micro level, the representation of a voice in terms of a dynamical system whose dynamical behavior undergoes constant mutations leads to interesting transitional stages when these mutations are extreme. At the same time, describing the dynamics in terms of a cycle along with a generalization scheme keeps us safe from any major instability. In order to observe the behavior of Psymbesis as a synthesis algorithm more closely, the author decided to

"slow down" the synthesis process an to build a graphical representation of the voice's dynamics in a 3D lag space using SGI's graphic library. In spite of the simplistic nature of the generalization scheme that we suggested for this particular implementation, the visualization of the voice's behavior in this state space fulfills our expectations. The state of this voice is clearly attracted by the support of the cycle. As we recall, this support stands for the desired long-term behavior of the system. At the same time, the voice's state doesn't confine itself to this support as it clearly visits a wider area of the state space in the case of a sudden mutation of the cycle. In the context of more traditional synthesis techniques such as additive synthesis, wave shaping, wave tables or FM synthesis, such transitions would need to be explicit within the control set of the synthesis engine. In the case of Psymbesis, one could say that we get these transitions "for free" from modeling our virtual instrument as a dynamical system instead of a wave form generator.

Weaknesses

In a concern of simplicity and clarity in the context of this implementation, the characterization and clustering of the control space was done on the exclusive basis of the measured values of the training data's musical gestures. Because the pseudo-periods exhibited by the data were never taken in account for this clustering, the resulting assignment to a stochastic period table to each cluster (or class of control) was not as successful as we may have hoped. Averaging pseudo-periods that were "too different" within each cluster resulted in a clear loss of the original sound's granularity. Part of this loss can be minimized if we choose to assign stochastic period tables from the identification of a reference pseudo-period within each cluster rather than from an averaging process. Any further improvement would require to reconsider the clustering in the control space (see "pointers to alternative interpretations").

Pitch-synchronicity implies the inference of a monophonic instrument. Indeed, even though the inferred virtual instrument can be used to synthesize polyphonic audio streams, each voice is perceived as an isolated dynamical system. Any polyphonic stream will essentially be an audio mix of several monophonic streams which don't interact with each other. One could very well imagine implementing cross-talk channels between voices in a final implementation. After all, each voice is a dynamical system and it would be very easy to force some of a voice's energy to bleed onto the other voices. However, the current scheme of *Psymbesis* doesn't suggest any means by which the character of these cross-talk channels may be inferred from observed data. In other words, such a feature would require some human supervision and common sense.

Finally, setting the generalization parameter α by hand and as a constant was a very useful feature in order to observe closely the behavior of the associated synthesis engine. It validated our approach to generalization in the dynamics space. However, in the general context of the inference of a virtual instrument from a recorded audio stream, this parameter should be estimated from the observed data instead of resulting from an arbitrary decision. Such a task doesn't introduce any major problem but it is closely related to the assignment of the stochastic period tables. The author feels like this task should be merged within a more careful approach to the characterization of the virtual instrument's control space.

Pointers to Alternative Interpretations

The previous description of a simplistic interpretation of *Psymbesis* resulted in the validation of this approach to sound synthesis. By now, we know it works and we can confidently point the reader to more elaborate interpretations which will suggest potential improvements.

Merging Control and Dynamics

As we said earlier, the characterization of the control space (nature of the musical gestures and clustering) and the characterization of the system's dynamics (cycles and synthesis engine) were presented separately for clarity purposes. Once we understand the basic idea of associating dynamical descriptions to sub-areas of the control space and the notion of pitch synchronous representation of dynamics (cycle), we might decide to interpret them as being part of a unique system. It was the exact same line of thought that brought us from separable Gaussian probability mass function estimation to the more general description of *Cluster-Weighted Modeling*. We already raised the fact that an appropriate clustering for the control space should take cycle descriptions into account in addition to the values of the controlling musical gestures. Merging control and dynamics appears like a good ground to do so.

Alternative Description for a Cycle

We've defined a cycle as a stochastic process with periodic mean and variance. It could very well be that we don't care all that much about its stochasticity in the context of musical instruments. However, more important than this eventual randomness is the description of generalization in dynamics space. While implementing the previous proof of principal, we've chosen to set the generalization parameter α once and for all for a particular virtual instrument. A great deal of argument could be raised by this fairly arbitrary choice. It is likely that a more meaningful approach to this generalization could come from the integration of α (or some variant if we decide to use a different type of generalization function) in the description of a cycle.

Structure in the Control Space - Control Advisor

The degree of expressiveness that musical gestures such as volume contour, pitch contour, noisiness and brightness offer is so fine-grained that we've automatically assumed that they would be an ideal set of controls. This perspective is taken from the point of view of those of us who may be frustrated by a traditional synthesizer's lack of fine control. However, one can very well imagine cases where such a precise set of knobs may appear as a burden in the absence of any short-cuts. After all, these musical gestures, being perceptually meaningful, are easier to control than an instrument's physical interface but can still lead to a rather tedious representation of a musical stream when compared to, let's say MIDI. Chances are that in the case of a specific instrument which is played in a "normal" or "traditional" fashion, the musical gestures obey some extra rules which may relate them one to another, limit their jaggedness, or simply constrain their ranges. As it is, *Psymbesis* offers control without constraints and leaves it up to the user (or controller) to figure out how to make this instrument sound good the same way a real musician would have to.

Studying eventual relationships between observed musical gestures in the context of an instrument may reveal some structure in the control space. Such structures could then be used in order to build an appropriate *control advisor*. This way, if we felt like providing the system with more sketchy commands such as "play that note now", the control advisor may be able to suggest an appropriate way to get there based on its expertise. The development of such an *advising layer* is not directly involved in the development of the virtual instrument in the sense that it doesn't necessarily fit in the area of sound synthesis. The author felt like referring to such a possibility at this point in order to emphasize the fact that it wouldn't take much to scale *Psymbesis* back to a MIDI box. After all, backwards compatibility is a crucial issue and a new synthesis method shouldn't necessitate an entire industry to start from scratch.

Chapter Summary

As we continue to learn more about non-linear systems and develop better tools to cope with non-locality, a straightforward application of embedding modeling will eventually provide a satisfying approach to the modeling of a musical instrument. In the meantime however, the pressure of applicability concerns pushed the author to compensate the difficulty of these issues with a restriction of an approach's expertise. Once focused on the particular case of musical instrument, we took advantage of some basic expectations about the data in order to introduce a modeling scheme which is more likely to fulfill our purpose in the shorter term.

In order to cope with the variety of time-scales one may encounter, we drew a clear distinction between the control (low bandwidth/long time-scale) and the dynamics (high bandwidth/short time-scale). As for the long-term behavior of the dynamics, the omnipresence of periodicity among musical signals stimulated the notion of *cycles* and pitch-synchronous representations. This way, the representation of the system's dynamics, which is typically local information if we think in terms of prediction surfaces, implicitly carries the long term inclination of the system (which is to settle eventually as a periodic wave-form).

We stated clearly the assumptions we derived from these general observations of musical sounds. For instance, we assumed that a static set of controls should result in the dynamics settling down on a cycle. Although these assumptions are justified only by the author's common sense, they are based on the nature of the data. They attempt to qualify the approach's area of expertise and are not the artificial product of a specific modeling tool. Given these not-so-arbitrary choices, we then suggested a possible architecture for a virtual instrument's characterization and synthesis engine. While doing so, we derived a framework which was precise enough to be justified, but open enough to leave leeway to a wide range of interpretations at various levels. Some of these choices are at a very low level while others could dramatically influence an eventual implementation.

In order to validate our framework, we walked through the simplest possible interpretation of the ideas we gathered. The outcome of this interpretation stands for our proof of principle as it is itself a working system which already does much more than any other automated modeling system. It can infer a virtual instrument without supervision from any monophonic audio recording and synthesize any arbitrary musical phrase based on the inferred model. These primary results are a preview of Psymbesis's promising potential. In addition, we've provided pointers to more elaborate interpretations which, in addition to improving the system's performance, could link it back to *Cluster-Weighted Modeling*.

Conclusions

Recapitulation

Considering a general musical process, we've drawn distinctions between various levels of abstraction and areas of expertise that may be involved in the qualification of such a process. While standing clear from musicology and cognitive sciences, we identified a mid-level of abstraction with the notion of *musical gestures*. These gestures don't pretend to carry any understanding of music on a cognitive level and they merely stand for an alphabet if we were to take language as an analogy for music. Because of their purposely limited level of abstraction, they are more likely to enjoy clear definitions and a very wide range of musical applications. In the third chapter of this document, we've suggested means by which such musical gestures can be estimated in real-time from a monophonic musical sound stream. In a way, the rest of this document can be seen as an investigation of means by which one could go the other way, turning streams of musical gestures into sound; a task traditionally know as sound synthesis.

Given the context of musical instrument modeling, rather than blindly applying traditional modeling tools, we've traced the problem back to its source. While doing so, we've brought some shaded areas of linear system theory back to the surface, and illustrated the misconceptions that could result from a model mismatch. In fact, any modeling technique which starts by making some strong assumption concerning the architecture of an eventual model before even looking at the data is bound to have the same fate. This "destructive" process would have left us with very little ground to build up from if it hadn't been for the application of Floris Takens' embedding theorem to non-linear modeling. Introducing non-linearities in an inferred model is not necessarily the answer to all the problems we might encounter, but not ruling it out is a good first step. By the end of the fourth chapter, we turned the modeling of a dynamical system into the estimation of a *prediction surface* from observed data, without introducing any assumption that wouldn't be derived from the observation itself. Rather than referring to our observations in terms of musical sounds, we've purposely kept our discussion in terms of time-series and dynamical systems, emphasizing the wide range of *Embedding Modeling*'s applicability.

Two general approaches to the characterization of prediction surfaces were then presented in the following two chapters. Far from spanning the entire range of possibilities that one might

think of, these two approaches illustrate two very distinct points of view. First, estimating a global polynomial form for the prediction surface was motivated by a wish for a unifying global description of the system's dynamics. For this purpose, we stated this polynomial fit in terms of a linear estimation for which we've suggested the usage of a Kalman filter. Second, a cluster-based local description of the data's probability distribution was motivated by accuracy concerns as well as system's characterization it provides. Again, our concern for clarity led us to realize that clustering and characterization are not necessarily two separate tasks. This led us to introduce *Cluster-Weighted Modeling* as a general scheme which merges the two in a unifying set of relationships. In addition to providing valid schemes for the inference of non-linear models, these general purpose approaches were an opportunity to build a deeper understanding of the issues that one may encounter in the specific case of sound synthesis.

In light of what these attempts taught us, we then focused back on the modeling of a musical instrument and suggested an approach which is more likely to give immediate results for musical sound synthesis. The resulting method, *Psymbesis*, can be seen as a set of constraints and hypotheses concerning the nature of a musical instrument, that benefits from the insights we gained during the investigations of the preceding general purpose approaches. It leads to the automatic inference of a virtual instrument for which the control parameters are the perceptually meaningful musical gestures we identified in the third chapter of this document. The nature of the original instrument's observation upon which this inference is based is nothing but a simple audio recording.

Major Contributions

Although the author can't possibly claim the originality of an intercorrelation-based pitch extraction method or a pitch synchronous timbre analysis, the *Machine Listening* part of this study led to an original collection of real-time sound analysis tools. Their performance and flexibility motivated their use in the context of various projects throughout the past three years and they don't seem to be ready for retirement yet. These tools were developed and implemented with a strong emphasis on their expertise. In a way, one could say that their major strength resides in their lack of pretension rather than in their actual mechanism.

Sound synthesis is not a new task, but the "bottom-up" approach we've followed throughout this document makes it stand out from the more traditional works in that domain. Instead of inventing a new abstract synthesis technique from oscillators, filters or wave-tables, we've looked at synthesis as the inference of a physically meaningful system from the observation of an existing instrument. To the author's knowledge, such a task had never been performed before through an automated (and justified) process without manual tweaking. We've elevated sound synthesis from the state of magical recipes to a clearly stated modeling problem. Furthermore, the task of inferring a model itself was approached from a radically new perspective. Based on the application of a ten-year-old differential topology theorem, *Embedding Modeling* is barely starting to make its first steps in the context of non-linear dynamics, and its application to music was still a far fetched marginal idea only a couple of years ago.

Fitting polynomial surfaces to a prediction function had been attempted before in terms of the identification of an orthogonal basis of polynomial functions. On the other hand, Kalman filters have been used for all kinds of linear and non-linear estimation (and control) tasks. However, the mixture of these two concepts along with the "linearization" of the polynomial estimation problem awards an originality to the technique we've introduced in the fifth chapter. The novelty of *Cluster-Weighted Modeling* is even less arguable. The first step was largely

inspired by Prof. Rosalind Picard and Kris Popat works on the characterization of visual textures, and it led to the first decent audio reconstruction a couple of years ago as one of the author's general examination projects. The generalization of separable Gaussians was then mainly due to Prof. Neil Gershenfeld's wish for finer structure modeling. The explicit form of *Cluster-Weighted Modeling* and the realization that clustering and modeling had to merge in a single process in this context came as the recent answer to an increasing degree of confusion that surrounded the suggestion of increasingly complex models.

As for *Psymbesis* as a scheme for musical sound synthesis, its introduction was primarily motivated by applicability concerns. Although it may not appear to be a straightforward application of *Embedding Modeling* in the sense that it makes a series of assumptions concerning the nature of the system that is to be modeled, it attempts to address the major issues that we encountered throughout this study. In its first generation, it stills inherits some baggage from traditional sound synthesis techniques, but it uses it in a completely original fashion. Although notions such as *tables* and *phase* may remind us of wave-table synthesis or even sampling, the derivation of phase from the internal dynamical state of the system is absolutely unique.

Future Directions

While deriving the collection *(volume, pitch, noisiness, brightness)* of musical gestures, we never pretended to be exhaustive and an obvious direction of improvement would be to complete (or at least expand) this set of perceptually meaningful musical gestures. The actual implementations of these extraction modules could also be a source of future improvement in terms of software packaging and performance optimization. At this point, the software implementation of these tools are the result of a compromise between flexibility (which implies software modularity) and performance (which tends to result in bulky and compact code).

The estimation of a global polynomial form for a system's prediction surface doesn't seem to be appropriate for the inference of a musical virtual instrument (at least in the brute force version that was presented in this document). However, this process has a wide range of potential applications ranging from system monitoring to data compression. These applications spread out of this study's main concern with sound but this doesn't make them any less worthwhile for further investigations. A generalization of the suggested approach to a set of low-order polynomial relationship sounds particularly interesting to the author as it could be seen as an attempt to implement a *physical understanding* on a machine. Indeed, a set a low-order polynomial difference equations can be interpreted as a guess concerning the observed system's physical mechanism.

Our investigation of cluster-based, probabilistic modeling for a system is so general that a list of all possible applications and further developments would be prohibitive.*Cluster-Weighted Modeling* as we introduced it is only the first generation of its kind. Some further perfection and modifications will keep on pouring from Prof. Neil Gershenfeld's research after this work is presented. After having addressed issues such as accuracy and local structure in its current form, some possible directions might be to address time scales and long-term behavior (non-locality).

Finally, *Psymbesis* is a single suggestion concerning the specifics of musical instruments. The assumptions it is based upon are open to mutations as we learn new ways to overcome non-locality and multiple time-scales. Even in the form it was presented in this document, it offers

enough leeway to various interpretations and implementations. For instance, the nature of the control set could expand to additional musical gestures; the process of clustering the control space and assigning cycles could merge into a single process whose flavor would recall *Cluster-Weighted Modeling*; the definition, or representation, of the notion of *Cycle* could itself mutate to include a richer topology of primitives; the generalization scheme that is implied by the architecture of the synthesis engine could become part of the description of a cycle and be estimated from the data. As for its use as a synthesis engine, it may turn out that a further study of the control's behavior would lead to interesting short-cuts if we were to decide that we didn't systematically want such a fine grain of control over the resulting sound stream.

The Last Word

Rather than describing a particular setup or even a specific synthesis algorithm, we've presented a new set of ideas and hopefully a philosophy which promises to lead to a new generation of modeling schemes for a wide range of applications. Throughout our investigation of musical sounds characterization, we've spanned a variety of issues ranging from signal processing, dynamical systems, modeling, clustering, to even information theory. While doing so, we've expressed a wish for rigor and provided new ideas which enjoy both solid justifications and wide ranges of expertise. We've tried to illustrate the boundless range of interpretations that may result from this philosophy and the author would like to end with a few convictions concerning the future of sound synthesis and how embedding modeling fits in this context.

It seems clear that synthesizers are bound to evolve towards high speed general computers. The author shares this conviction with the vast majority of researchers who are involved with sound synthesis in universities and industries. As an analogy, computer graphics has elevated animation from a succession of two-dimensional frames to the description of virtual worlds populated by objects that carry specific properties and appropriate behaviors. There is no reason why sound synthesis shouldn't have the same fate. The author believes that any virtual instrument should be described in terms of a sound object's properties and behaviors instead of a succession of abstract parameters that reflect the limited expertise of a synthesis algorithm. *Embedding modeling* appear to be a valid means by which one may extract these properties from the observation of an instrument. It makes no doubt that the interpretations of this philosophy are called to mutate and lead to multiple and diverse approaches, and this reinforces the need to an algorithmic representation of sound within synthesizers. Even if simulating an existing instrument is not every one's main concern within the music community, "surreal" virtual instruments are more likely to be interesting and fun to play with if they exhibit peculiar but coherent behaviors. Sound synthesis patents are running out and synthesizers have been based on the same old ideas for the past twenty years. The time is right for a drastic change in the computer music industry, both in terms of a more open architecture of the hardware, and in terms of sound representation. As premises of this change, the recent appearance of so-called "lead-synthesizers" (Yamaha's *VL1* and Korg's *Prophecy*) offer to compromise something as precious as polyphony for the sake of interesting behaviors. It is unclear as to whether or not the architecture of this hardware is more open but it reveals a clear interest in the dynamical property of a sound object.

References

[ABST93] Henry D. Abarbanel, Reggie Brown, John J. Sidorowich, and Lev Sh. Tsimring. *The analysis of observed chaotic data in physical systems*. Reviews of Modern Physics, Vol.65, No.4, October 1993.]

[Arf79] Daniel Arfib. *Digital synthesis of complex spectra by means of multiplication of non-linear distorted sine waves*. Journal of the Audio Engineering Society 27: 757-768.

[Bar91] J. B. Barrière. *Le Timbre, Métaphore pour la Composition*. Ircam, Christian Bourgois éditeur, 1991.

[BH92] Robert G. Brown and Patrick Y.C. Hwang. *Introduction to Random Signals and applied Kalman Filtering* (second edition). John Wiley & Sons, 1992, (selected chapters).

[Bla73] John Blacking. *How musical is man?* University of Washington Press, Seattle and London, 1973.

[BPB85] J. B. Barrière, Y. Potard, and P. F. Baisnée. *Models of continuity between synthesis and processing for the elaboration and control of timbre structures*. ICMC '85 Proceedings, p.193-198, 1985.

[Bro94] Reggie Brown. *Orthonormal Polynomials as Prediction Functions in Arbitrary Phase Space Dimensions.* Institute for Nonlinear Science, University of California, San Diego, 1994.

[Cas92] Martin Casdagli. *A Dynamical Systems Approach to Modeling Input-Output Systems.* Nonlinear Modeling and Forcasting, SFI Studies in the Sciences of Complexity, Proc. Vol.XII, Eds M. Casdagli & S. Eubank, Addison-Wesley, 1992.

[Cha81] Gérard R. Charbonneau. *Timbre and the Perceptual Effects of Three Types of Data Reduction.* Computer Music Journal 5(2) p.10-19, 1981.

[Cho73] John M. Chowning. *The Synthesis of Comples Audio spectra by Means of Frequency Modulation*. Journal of the audio Engineering Society 21: 561-534. Reprinted in C. Roads and

J. Strawn editions 1985 "Foundations of Computer Music". Cambridge, Massachusetts: MIT Press.

[Duf88] Hugues Dufourt. *Hauteur et timbre.* From "Inharmoniques", 3, Musique et perception. Ircam - Christian Bourgois éditeur, 1988.

[Ell96] Daniel P. W. Ellis. *Prediction-driven computational auditory scene analysis*. PhD dissertation, Department of Electrical Engineering and Computer Science, MIT, June 1996.

[Emm86] Simon Emmerson. *The Language of Electroacoustic Music.* Macmillan Press, 1986.

[ER85] J.-P. Eckmann and D. Ruelle. *Ergodic theory of chaos and strange attractors.* Reviews of Modern Physics, Vol.57, No.3, Part I, July 1985.

[Ger88] Neil A. Gershenfeld. *An Experimentalist's Introduction to the Observation of Dynamical Systems.* Directions in Chaos, Vol.II, Hao Bai-lin ed., World Scientific, 1988.

[Ger92] Neil A. Gershenfeld. *Information in Dynamics.* Workshop on Physics and Computation, PhysComp'92. IEEE Computer Society & ACM/SIGGRAPH.

[Gre75] John M. Grey. *An Exploration of Musical Timbre*. PhD dissertation, department of Psychology, Stanford University, 1975.

[GW93] Neil A. Gershenfeld and Andreas S. Weigend. *The Future of Time Series.* Time Series Prediction: Forecasting the Future and Understanding the Past, p.1-70, Addison-Wesley, 1993.

[Han89] Stephan Handel. *Listening*. MIT press, Cambridge, Massachusetts, 1989.

[Har85] W. M. Hartmann. *The frequency-domain grating.* Journal of the Acoustical Society of America 78(4), p.1421-1425, 1985.

[Mac92] Tod Machover. *Hyperinstruments: a Progress Report*. Available from the Media Laboratory of the Massachusetts Institute of Technology, 1992.

[Mca88] Stephen McAdams. *Perception et intuition: calculs tacites.* From "Inharmoniques", 3, Musique et perception. Ircam - Christian Bourgois éditeur, 1988.

[Min85] Marvin Minsky. *The Society of Mind*. Simon & Schuster 1985.

[Moo77] James A. Moorer. *Signal Processing Aspects of Computer Music - A Survey.* Computer Music Journal 1(1), p.4-37, 1977.

[Mur84] Tristan Murail. *Spectres et lutins.* From "L'IRCAM une pensée musicale" , Tod Machover. Collection "musique de notre temps" éditions des archives contemporaines, 1984.

[MYC91] Yoav Medan, Eyal Yair and Dan Chazan. *Super resolution pitch determination of speech signals*. IEEE Transactions on signal processing. vol 39, n°1 , January 1991.

[OS89] Alan V. Oppenheim and Ronald W. Shafer. *Discrete-Time Signal Processing.* Prentice Hall, 1989. (Sampling, LCCDE, spectrum analysis)

[PL94] Rosalind W. Picard and Fang Liu. *A new Wold ordering for image similarity.* Proceedings of IEEE Conference on Acoustics, Speech, and Signal Processing, Adelaide, Australia, April 1994.

[Por76] Michael R. Portnoff. *Implementation of the Digital Phase Vocoder Using the Fast Fourier Transform*. IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol ASSP-24, #3, June 1976, pp243-248.

[PP95] Kris Popat and Rosalind W. Picard. *Cluster-based probability model and its application to image and texture processing*. To appear in IEEE Transaction on Image Processing.

[PPR91] Giovanni De Poli, Aldo Piccialli, and Curtis Roads. *Representations of Musical Signals.* MIT Press, 1991, (selected chapters).

[Ris88] Jean-Claude Risset. *Perception, environnement, musiques.* From "Inharmoniques", 3, Musique et perception. Ircam - Christian Bourgois éditeur, 1988.

[Rod93] Xavier Rodet. *Flexible yet controllable physical models: a nonlinear dynamics approach.* Proceedings of the 1993 International Computer Music Conference.

[RW82] Jean-Claude Risset and David L. Wessel. *Exploration of Timbre by Analysis and Synthesis.* The Psychology of Music, Deutsch eds. Orlando: Academics, 1982.

[Sch77] Pierre Schaeffer. *Traité des objets musicaux*. Paris - Editions du Seuil, 1977.

[Sla68] A. W. Slawson. *Vowel Quality and Musical Timbre as Functions of Spectrum Envelope and Fundamental Frequency*. The Journal of the Acoustic Society of America, vol 43, #1, p.87-101, 1968.

[SS90] Xavier Serra and Julius O. Smith III. *Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition.* Computer Music Journal 14(4), p.12-24, 1990.

[SR70] Ronald W. Schafer and Lawrence R. Rabiner. System for Automatic Formant Analysis of Voiced Speech. The Journal of the Acoustical Society of America, vol 47, #2(Part 2), p.634-648, 1970.

[Str94] Steven H. Strogatz. *Nonlinear Dynamics and Chaos*. Addison Wesley, 1994.

[Tak81] Floris Takens. *Detecting strange attractors in turbulence.* Lecture Notes in Mathematics vol.898 (Springer, Berlin) p.366-381, 1981.

[The89] Charles W. Therien. *Decision Estimation and Classification.* Wiley&Sons, 1989.

[The92] Charles W. Therien. *Discrete Random Signals and Statistical Signal Processing.* Prentice Hall, 1992. (Stochastic modeling, Wold decomposition, spectrum analysis)

[Wes79] David L. Wessel. *Timbre Space as a Musical Control Structure.* Originally published in Computer Music Journal 3(2): p.45-52, 1979. Republished in "Foundations of Computer Music", Curtis Roads Eds. MIT Press p.640-657.

[Won92] Yiu-fai Wong. *Clustering Data by Melting*. Neural Computation 5(1), MIT, p.89-104, 1992.