Detection and modeling of transient regions in musical signals

Gouyon Fabien
fgouyon@ccrma.stanford.edu

September 1999

"Talking about music is like dancing about architecture."

Martin Mull

Abstract

It is well-known that regions corresponding to onsets and decays of notes are essential in the perception of a sound in order to determine its source recognition and its natural or synthetic character. Some important work has been and is being done regarding the detection of onsets in audio and the segmentation of signals, as well as regarding the characterization of musical events according to meaningful and practical features. Although many methods for signals representation exist, most methods do not represent certain problem regions coherently (attacks or decays of notes for example, but also other regions), and therefore can neither modify nor reproduce these regions in a meaningful way.

Any accurate sound content description and analysis/synthesis application (pitch and time scaling, compression, hybridization...) calls for a *unification of the characterization and segmentation issues*. Thus, in this work, we consider parallel and interacting schemes for detection and modeling of *nonstationarities*, or transient regions in general, which arise from the polyphonic nature of the signal as well as from each instrument's separate performance. We present an analysis/synthesis framework involving an explicit parametric model for transients based on Prony's technique. Explicit modeling of transients should be deeply embedded with an accurate segmentation of the signal. Segmentation, here, is based on the detection of abrupt changes in signal statistics; this generalizes well-known techniques based solely on the detection of energy-jumps as the discriminant statistical parameter.

Keywords

Parametric modeling, Prony model, Transient detection, Segmentation, Additive synthesis, "Sines + Noise + Transients" characterization of musical signals.

General information

This thesis gives an overview of my research at the Center for Computer Research in Music and Acoustics (CCRMA[1]) in Stanford University between March and September 1999. It serves as the final report required for my French degree in signal processing: the DEA SIC[2] (Diplome d'Etudes Approfondies, Signal, Images et Communications) in Toulouse, France.

This document summarizes my part of the project Harvey Thornburg and I began to work on, and will hopefully be able to pursue together.

My advisors for this work were:

- Dr Julius Orion Smith III at CCRMA, Stanford University, USA
- M Yannick Deville in LAMI (Laboratoire d'Acoustique, de Metrologie et d'Instrumentation), University Paul Sabatier, Toulouse, France
- M Francis Castanie in ENSEEIHT (Ecole Nationale Superieure d'Electronique, d'Electrotechnique, d'Informatique et d'Hydraulique de Toulouse), France

For information and comments:

Mail: 16, rue Maurice Alet 31400 Toulouse FRANCE
email: fgouyon@ccrma.stanford.edu
URL: `http://www-ccrma.stanford.edu/~fgouyon`

[1] See `http://www-ccrma.stanford.edu`
[2] See `http://www.enseeiht.fr/Recherche/LEN7/deasia/index.html`

Acknowledgements

There are so many people without whom this work would not exist, I have trouble thanking them all...

First of all, I would like to thank my advisors: Julius Smith who allowed me to work at CCRMA under his guidance; Yannick Deville and Francis Castanie who agreed for me to do my DEA training period abroad.

My next thanks goes to Harvey Thornburg who I enjoyed working with. There doesn't seem to be any digital signal processing or statistics issue he could not work out.

To my friends in Stanford, it truly has been enjoyable and insightful to meet you guys. I really enjoyed my stay thanks to: Hendrik Purwins, Stefania Serafin, Tamara Smyth, Bob Sturm, Stefan Bilbao, Juan Pampin, Katsuhiko Sakamoto, Michael Janus, David Berners, Fernando Lopez-Lezcano, Dan Levitin, Caroline Traube and the others Karmalites. Thanks to Harvey, Stefan and Bob who spent time rectifying some "frenchicisms" in this document.

I would also like to thank all those who helped me and gave me precious advice during this period: Xavier Serra, Daniel Arfib, Jean-Yves Tourneret, Corinne Mailhes, Regine Andre-Obrecht, Jean-Claude Risset, Jonathan Berger.

For her patience and efforts during the preparations for my arrival I also would like to thank Heidi Kugler.

And last, but obviously not least, I would like to thank my family (Valerie, both my parents and theirs, and so on), as well as my French friends, for always being here for me (virtually or even physically).

Contents

List of Figures

Chapter 1

Introduction and review

This chapter introduces the topics of interest of this work: the detection of transients and the modeling of the signal in the transient portions. We also propose a review of onset detection in the context of analysis/synthesis of digital musical signals. The main purpose of this introductory chapter is to justify the philosophy of considering jointly the characterization (modeling) and the segmentation (detection) issues.

1.1 Definition of "Transient"

Born from the interaction between several scientific and artistic fields amongst which are signal processing, music and psychoacoustics, some computer music definitions lack rigor in their definitions. In fact, the central term of this work ("transient") doesn't have a well defined meaning in the computer music community; however, from a signal processing point of view, the following features serve as a definition: A transient is localized over a very *short* time region of signal. A transient can be considered either as a *deterministic* signal or a *stochastic and highly non-stationary* signal. A stochastic and stationary signal cannot be a transient.

- Considering the signal as *deterministic*, a transient would differ from the other regions by the fact that the parameters that would describe it would have important fluctuations. As Serra wrote in [Serra/Bon.]: "Attack and release regions are identified by the way their instantaneous attributes change in time and the steady state regions are defined by the stability of the same attributes". Parameters to monitor can be very numerous,[1] and the fluctuation criterion is imprecise by virtue of the fact that all these parameters may not vary exactly at the same time or in the same way.

- Considering the signal as a *realization of a stochastic process*, a transient stands as the short transition between two stationary regions of the signal, and it differs from them for its important *non-stationarity*.

Obviously, the "shortness" of a signal's region and the "fluctuations" of its descriptive parameters (that are not yet rigorously defined) are very vague characteristics. In order to maintain the rigor of the subsequent work, we choose to interpret the musical signals as *realizations of stochastic processes* rather than deterministic signals; in this framework that accounts for the statistical behavior of signals, the notions of stationarity and non-stationarity that are essential in the characterization of transients have precise meanings (see Appendix 5 page 102).

Attacks of instruments fulfill the definition of transients; however, the opposite isn't satisfying, transients in musical signals should be considered more generally. Mainly, people have been looking for better ways to detect "onsets" in a signal (see subsection 1.1.1 and [Klapuri], [Schloss], [Shahwan],...). But it should be interesting to focus on any region of a musical signal in which, within a short time, the stationary attributes evolve in a significant way (the apprehension of this yields to the detection issue -see section 1.2). There is still a need for a model of signal that would deal with any transient region of the signal, beginning by onsets and ends of instruments tones.

1.1.1 State of the art in onset detection

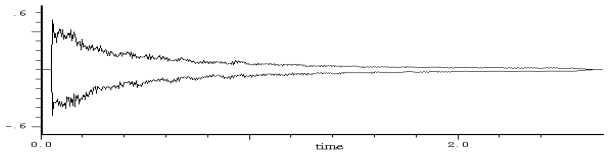
As one can see in figure 1.1, an important part of a tone issued from a musical instrument looks very much like an impulse signal. Experiments in the early days of computer music showed that cutting (literally) the signal after its attack region (the one isolated in (b) with the software SMS[2]) and listening to its leftover would lead one astray regarding the type of instrument that produced that sound. Thus, the timbre recognition of a sound is very much related to its attack. Transients in musical signals have been assimilated to attacks of instruments. This has to do with the fact that one of the very first and important goal in computer music research has been the automatic determination of rhythm in music. Looking for rhythmic clues within the signal naturally requires finding the boundaries of notes played by instruments, which stands as a primary step for fascinating (and very fashionable) topics in computer music: automatic transcription of music[3] and sound source separation[4].

[1] In Serra's case the attributes are: frequency, amplitude and phase of each partial, spectral characteristics of the analysis' residual, fundamental frequency, amplitude and spectral shape of the sinusoidal component and the residual component, noisiness, harmonic distortion, spectral centroid, and finally spectral tilt.

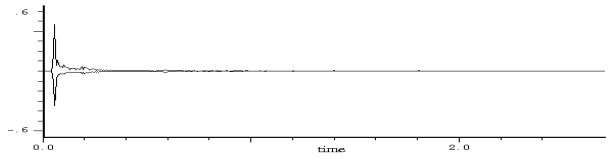
[2] See page 21, for precisions on SMS.

[3] see [Schloss] and [Klapuri]

[4] Those are not the goals of this thesis.



(a) Original bell sound



(b) residual

Figure 1.1: Original sound and SMS analysis residual: single tone (bell)

The criteria for the detection of onset have evolved. They were first based on slope detection, as explained in [Schloss]: an amplitude envelope of the signal is obtained, and a slope detector is applied to it. Decisions are made according to thresholds. Others, more recent, respond directly to sudden changes in energy levels. A simple method is to compute a short-term average energy and look for places where the energy quickly exceeds a preset threshold. A more advanced method is to make the threshold adaptive to the level of energy in the signal so that the threshold value tracks the recent activity level of the signal. One could also think about decomposing the signal into many different frequency bands that would be processed independently and eventually combined. According to KLAPURI (see [Klapuri]), if one wanted to focus on the perceptual onsets of sounds, the application of psychoacoustic knowledge (psychoacoustic models of intensity coding) could also permit the determination of beginnings of sounds that exhibit onset imperfections (i.e. sounds for which duration can be long and amplitude envelope does not rise monotonically). See also LEVINE's work in subsection 1.2.3.

These methods currently yield good results (see [Klapuri]) concerning the detection of onsets. However, they are mainly based on the detection of energy-jumps and focus on this particular statistical parameter. Thus, they usually fail when faced with detecting other problem regions of signals. This inadequacy increases with the signal's complexity (from monophonic to polyphonic). Moreover, these methods are not related to any modeling scheme.

1.2 Detection and modeling

At the outset, these are two different topics. This section gives the reasons we intend to cope with them simultaneously.

Suppose one is interested in extracting features from a musical signal and using them in order to represent, resynthesize or transform the sound. This could be done using a certain analysis scheme, an assumed type of model would eventually yield parameters that are supposed to suitably characterize the signal. When a region of the signal breaks the model (i.e. is not sufficiently well characterized by it), then one knows that this region corresponds to an event that needs a different description. Thus, from a logical point of view, the modeling scheme and the detection scheme should share information. Moreover, the detection scheme should be sensitive to the evolution of the stationary attributes of the signal, and the evolutionary Prony model we intend to build is meant to characterize certain types of non-stationarities (see section 4.2). Therefore, they both should be driven by a criterion representative of the evolution of the degree of non-stationarity in the signal.

Segmentation and description of sounds should be considered in parallel. Indeed, one basically wants to deal with the issue: "What are we describing?" In order for the description scheme to make sense, one should know what are the boundaries of the object of the description. Likewise, in order for the segmentation scheme to make sense, one should know which changes in the signal's very nature one is looking for.

Thus, our work is not so much based on a first extraction and a subsequent characterization of a certain portion of note of a single instrument. It is based on the following general (but not so musical) starting point: The signal should be processed sequentially, one should focus on the evolution of the degree of non-stationarity in a musical signal, assuming that this criterion is relevant to define accurate and meaningful boundaries as well as to give clues on the description scheme needed.

1.2.1 Segmentation issues

Segmentation of audio recordings facilitates editing operations and the synchronization of audio and video.[5] A features extraction of the sounds contents achieved by an automatic analysis is an major goal in any indexation purpose.

Different levels of segmentation can be defined. According to Lepain (see [Lep./Obrecht]), a first scale could be defined that accounts for changes in dynamic, frequency, timbre or rhythm. A larger scale would account for regions within which several changes of the latter parameters could occur but that would

[5] Ongoing work on audio segmentation in terms of speech, music and noise is being done in IRIT (see `http://www.irit.fr/ACTIVITES/EQ_IHMPT/participants/obrecht/regine.html`) and IRCAM (see [Rossignol]).

still be relevant as a unit, an entire chord in a polyphonic orchestra for example. Other levels can be defined; for example, one could segment a sound according to patterns (rhythmic or melodic), or according to large temporal regions (piano chorus, regions with presence of voice, etc...). At this level, the type of indexation is mainly up to the user.

Lots of work concerning the segmentation of speech has been done over the last thirty years, and several algorithms have been developed. (see in section 6.2)

1.2.2 Analysis/synthesis issues

The basic goal in analysis/synthesis concerns stands in characterizing musical signals by *meaningful* and *flexible* parameters. Finding a way to describe the signal efficiently would lead to a possible resynthesis that would be perceived as the original signal; moreover, it could generate a powerful and meaningful sound creation tool. For example, analysis and synthesis of audio oriented toward time and pitch scale modifications, as well as data compression, are current research topics.[6] But the accomplishment of a meaningful parameterization for sound transformation applications is a difficult task.

Although many methods for musical signal parameterization and modifications exist, most methods do not represent transients coherently, for they are Fourier-based methods. Indeed, these approaches are optimally suited for stationary signals such as steady-state sinusoids and filtered noise. However, musical signals contain many non-stationary, or transient regions. Transients have been defined above by two features: short duration, and unstability/non-stationarity. Intuitively those features call for a descriptor that would have both good time precision and frequency precision. Thus algorithms based on Fourier techniques and other signal representation techniques (see section 2.1) are bound to fail.

Historically, sinusoidal modeling techniques (i.e. based on a Fourier analysis) have been used in the fields of both speech data compression (see [Quat./McAul.]) and musical data analysis/synthesis.[7] An example of analysis in terms of frequency components is given in figure 1.2.

In the computer music world, sinusoids alone are not found to be sufficient to yield high quality analysis/synthesis schemes, recent improvements involve adding a noise model to those schemes. Serra made the contribution of a residual noise model that models the non-sinusoidal part of the input as a time-varying noise source. These systems are referred to as sines + noise algorithms.

Spectral Modeling Synthesis (SMS) is a software developed by Serra that performs a sines + noise analysis,[8] and permits different sorts of synthesis (resyn-

[6] Time-scale modifications alter the playback speed of audio without changing the pitch. Similarly, pitch-scale modifications alter the pitch of the audio without changing the playback speed.

[7] In the latter field, the references are very numerous: see for example work by Risset, Serra, Smith, Rodet,..

[8] See `http://www.iua.upf.es/~sms/`

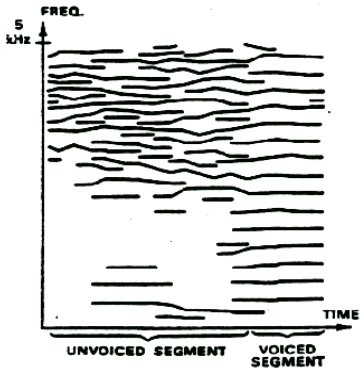


Figure 1.2: Example of a sinusoidal analysis of a sung melody (analysis performed with SMS)

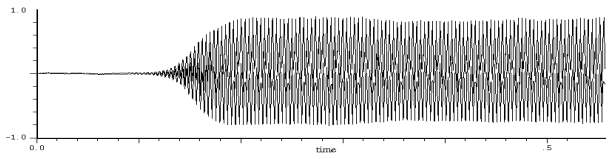
thesis, time-scaling, hybridization,..). It processes the signal as follows: analyze it in terms of sinusoidal components, compute the residual (i.e. non-sinusoidal part), and parameterize it as a time-varying noise source. Figure 1.3 shows an original clarinet sound, and the residual part left from the sinusoidal analysis. One can see that a precise region of the residual obviously does not correspond to a stationary noise component.

NB: For other software achieving similar tasks, see for example James Beauchamp's program,[9] the one produced at CNMAT Berkeley,[10] or the software "additive" developed at IRCAM.[11]

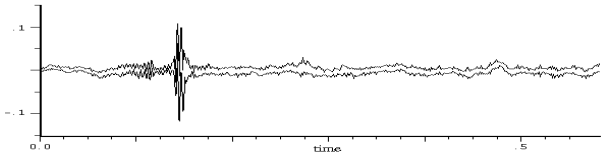
The observation that some regions of the signals (mainly onsets and ends of instruments' notes) aren't well modeled naturally calls for extending the sines + noise models to sines + noise + transients. Moreover, a well-known artifact of the sines + noise modeling of transients is the pre-echo problem: an instrument with sharp attacks would suffer a smearing in time of its attacks.[12] A solution is to simply remove the transient areas (assuming those can be found!), store it as samples, and add it back to the sines + noise components in a resynthesis stage (here it is vital to focus on phase-matching). But this method obviously lacks the purpose of flexibility.

In [Verma2], Verma and Levine suggest that the transients should be handled explicitly in order to provide a more realistic and robust model.

[9] `http://cmp-rs.music.uiuc.edu/cmp/software/sndan.html`
[10] `http://cnmat.CNMAT.Berkeley.EDU/CAST/`
[11] `http://www.ircam.fr/equipes/analyse-synthese/additive/index-e.html`
[12] Roughly, the reason is that due to the Fourier analysis, a windowing of the signal in the time domain corresponds to a convolution in the dual domain; see [Levine], on page 30 for details



(a) original



(b) residual (note different Y-axis scale)

Figure 1.3: Original sound and SMS analysis residual: single tone (clarinet)

1.2.3 Work by Scott Levine and Tony Verma at CCRMA

A common goal in the work of these two past PhD students at Stanford University was to improve the analysis/synthesis methods and extend them to sines + noise + transients schemes.

One of Verma's goals was to devise new ways to synthesize realistic attacks of instruments.[13] His Transient Modeling Synthesis (TMS) algorithm is the frequency dual to sinusoidal modeling. While sinusoidal analysis finds well-developed sinusoids by tracking spectral peaks over time, TMS finds transients by tracking the well-developed spectral peaks of a frequency domain signal. Impulse-like signals in the time domain are periodic in the frequency domain. A sinusoidal modeling (processed by SMS) of the frequency domain signal yields information on both location and characteristics of an impulse-like signal in the time domain. The mapping from the time domain to frequency domain is provided by the Discrete Cosine Transform (DCT). The frequency of the cosine (in the DCT domain) indicates the location of the impulse (see [Verma2]). Roughly speaking, TMS detection scheme is based on the fact that the DCT gives a "nice" representation of an impulse-like signal (here, "nice" stands for "easy to apply SMS on").

Levine[14] (see [Levine]) devised a sines + noise + transients representation of the signal in order to achieve both data compression and modifications. He

[13] `http://www.stanford.edu/~darkstar/`
[14] `http://www-ccrma.stanford.edu/~scottl/`

detects attack-transient regions and models them using transform coding techniques. The non-transient regions are modeled by a mixture of multi-resolution sinusoidal modeling and noise modeling. Regarding the handling of the transients, here is a very basic summarized version of his algorithm:

1. Detection: The transient detector uses two combined methods to find the onset times. The first method looks at the rising edges of the signal's short-time energy. If the current computed short-time energy is much larger than the average of the previous, then the current frame is tested by the second method. In the subsequent method, one looks at the short-time energy of the *residual* between the original signal and its synthesized version using multiresolution sinusoidal modeling.

   A decomposition of the signal into different frequency bands can also be processed, this would be a more efficient way to deal with what LEVINE calls "microtransients": sounds that are not broadband enough to be found by the basic detection algorithm (like closing hi-hat for example). Eventually, a thresholding decision is made. When an onset is detected in a frame, a region covering 66 ms of the surrounding signal is labeled as a transient.

2. Modeling: Time regions segmented by the detection scheme are not covered by the sinusoidal analysis, they are covered by a serie of short (2.9 ms) windows that are coded with a *transform coding technique* using a Modified Discrete Cosine Transform (MDCT).

   According to him, drawbacks of a transform coding technique are the following: First, it requires a relatively large number of bits per second, when compared to sinusoidal and noise modeling. Second, it is currently *not possible* to perform modifications on transform-coded MDCT coefficients.

LEVINE's method lacks flexibility for it doesn't yield a practical set of parameters. Moreover, it doesn't link the detection scheme and the modeling scheme.[15]

VERMA's algorithm is more flexible for it eventually uses an inherently flexible existing software (SMS) to model impulse-like signals (or attacks, those stands as transients definition in his work). Thus, each parameter of SMS (they are numerous) can act as a control parameter of the synthesized transient. Resynthesis of an attack is possible, as well as manipulation of the model's parameters; and new sound textures can be explored.

Looking precisely in his work, one could figure out that VERMA's TMS implicitly model transients as sums of *exponentially decaying sinewaves*. Indeed, the Discrete Cosine Transform of such sinewaves correspond to the type of signals SMS is geared towards (see [Verma1]). Exponentially decaying sinewaves correspond to the Prony model definition (as we will see in details in subsection

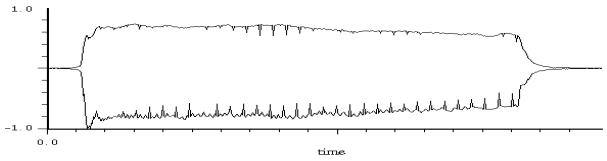
[15] Here, I must emphasize again that it was not its purpose.

3.1.1). Thus, *TMS implicitly makes an asumption on the nature of transients.* This asumption is the *same* we will make in modeling the transients as Prony-like signals. However, the difference between these two approaches is that TMS provides model parameters that are less meaningful than the direct Prony approach. Indeed, TMS parameters are inherently the same as SMS, but are used in a dual domain, what corrupts their initial meanings. Prony's parameters (i.e. frequencies, amplitudes, phases and damping factors) are more meaningful and relevant for transformations.

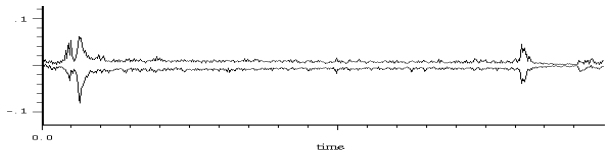
Together these observations motivate the search for a new transient model.

1.3 General overview of the project "Chicken and egg"

Sinusoidal analyses are very powerful and flexible but are also known to miss certain features of the signals. From the case of a single tone of a monophonic instrument, to the case of a monophonic musical phrase, and to an entire polyphonic signal, one can see (in figures 1.4 and 1.5) that attacks and decays of notes, as well as other regions of a signal (see figure 1.6) call for new modeling and detection schemes.



(a) original

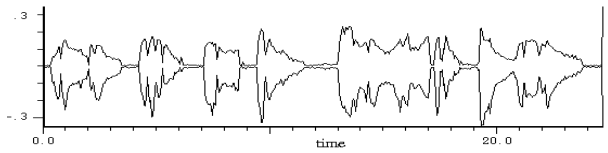


(b) residual (note different Y-axis scale)

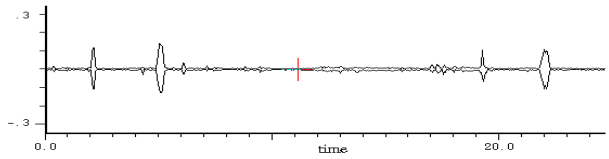
Figure 1.4: Original sound and SMS analysis residual: single tone (sax)

This observation motivated the following project:

In our work, musical signals are considered as the following of stationary regions between which are located transient regions. These regions evolve from very



(a) original



(b) residual

Figure 1.5: Original sound and SMS analysis residual: monophonic musical phrase (Elvis singing "Love me tender...")

non-stationary to quite steady (i.e. what is considered stationary[16]). The more a polyphonic signal is complex, in terms of number of events and number of sources (instruments) making the signal, the more non-stationary regions are numerous and complex to describe. It may make sense to analyze as *one non-stationarity* a precise problem region of the signal resulting from the addition of different transient events (issued from separate instruments).

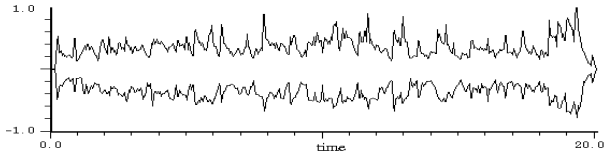
The main idea is to develop a transient model which takes advantage of segmentation information and which "fuses" well with a sinusoidal model for the steady state portion, SMS for example (see subsection 1.2.2). Applications should be as diverse as processing of transient sounds without echo and phase artifacts, to rhythm detection, to classification of timbres, plus others (see chapter 5).

Two kind of processing are sequentially applied to the signal:

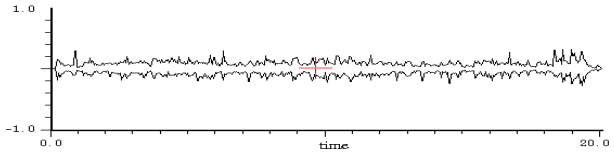
- Chicken: A method of *accurate* temporal detection of a transient. As the number of segments isn't known *a priori*, one may want to identify *when* and *if* changes occur by a sequential approach.
- Egg: A *flexible* modeling of transient regions using Prony's model (a region of the signal is modeled by a sum of damped sinusoids which parameters are determined from the original signal's samples).

The Chicken gives a time at which an abrupt change occurs in the stationary attributes of the signal. Then the Egg runs (sic): the signal is supposed to

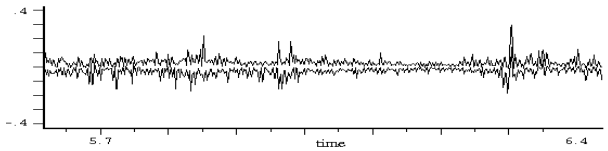
[16] See the definition in appendix 5 on page 102.



(a) original



(b) residual



(c) residual (detail - note different scales)

Figure 1.6: Original sound and SMS analysis residual: polyphonic musical signal

become progressively stationary, so features of the model of transient must give an indication of the non-stationarity decreasing (i.e. of the signal becoming progressively stationary again). Depending on this indication, the Chicken starts again, and so on....

1.4 Before proceeding...

The reader should be aware of the fact that we solely tested our algorithms over synthetic signals; this in order to obtain more meaningful results for the improvement of the method.

Likewise, the asumption regarding the generality of this project, and its relevance for polyphonic signals has not been tested yet.

Also, the detection of transient regions won't be discussed in this thesis. However, the junction between the modeling and the detection schemes remains a future goal.

As said a few lines above, the activation of the modeling scheme might eventually be monitored by an indication regarding the presence of a non-stationary region. Likewise this scheme must give an indication on the non-stationarity decreasing. In order to fit in the general philosophy introduced in this first chapter, this feature is perceived to be vital to the modeling scheme. Accordingly, we will focus on it in the subsequent work.

However, we consider worthwhile introducing in a very basic way the theoretical framework regarding the detection issue; the best place to introduce it is logically in the section regarding future work (section 6.2).

Part I

Modeling

The goal of this part is the modeling of transient regions. We discuss different aspects of this goal in three chapters:

- Chapter 2 gives justifications to the use of a Prony model.
- The basic Prony model is detailed in chapter 3 and applied to short regions of the signal considered stationary.
- Several adaptation of this model can be tried in order to handle the non-stationarity, this is the purpose of chapter 4. As it has been shown to be worthwhile considering jointly the modeling and detection issues, the eventual modeling method should be chosen so that it would give an indication on the rate of non-stationarity of the signal it is modeling, so that we can take the decision to run the detection algorithm again. This concern is highlighted in paragraph 4.4.2.

Chapter 2

Why Prony?

2.1 Parametric methods versus signal representation methods

In this very short section, we intend to focus on the basic differences between two types of spectral analysis techniques; this within a general framework.

Spectral analysis methods can be divided according to two categories: signal representation methods and parametric methods. The former don't make any asumption regarding the nature of the signal, their goal is to represent the signal in a practical way; they correspond to ways to focus on a certain kind of information contained in the signal, without loosing any of its contents. For the latter type of methods, the signal is assumed to correspond at the very outset to a certain structure defined by a set of parameters which values have to be found (or estimated); in this framework, one looses information regarding the true nature of the signal, but it stands as information that one implicitely chooses to be not important in his analysis of the signal.

The following techniques are examples of signal representation methods:

- Time/Frequency representations: Fourier transform, short time Fourier transform, Wigner-Ville, Rihaczek-Kirkwood
- Time/Scale representations: continuous wavelet transform, discrete wavelet transform

As for parametric methods, hese are some examples:

- ARMA, AR, Prony

Concerning signals that can be perfectly modeled by some parametric process, one can estimate the parameters of the model by estimation methods that use samples from the signal. In practical cases, the samples of the signal feed the same estimation methods, but the novelty is that one wants to determine the process which is the *closest* to the actual signal, accepting he is making an approximation. One could always try to find (and eventually achieving it) the best approximation of a given signal with respect to any parametric method, but the relevance of the use of this method or any other is still an open problem.

Methods that would determine how to best represent any signal don't exist, therefore one must make asumptions. Even when using non-parametric methods.

Indeed, a spectral representation based on Fourier allows one to visualize both sinusoidal and non-sinusoidal parts of the signal. Its output stands as a continuum of spectral data, unlike any modeling scheme. It yields what in the signal will be considered purely sinusoidal (and parameterized in consequence), as well as the non-sinusoidal part of the signal. In a musical analysis/synthesis concern, using a Fourier-based transform is a first step one currently makes. Subsequently, one usually processes a peak detection scheme in order to parameterize the signal in terms of meaningful and practical triplets {magnitude, frequency, phase}. In this step, an implicit asumption is made, which is basically the same as in Prony modeling: one wants to focus solely on the purely sinusoidal part of the signal. One could think about using a method that accounts for this asumption in its very structure: Prony's method.

2.2 Practical motivations

Eventually, this work must be linked to other existing sound representation tools that efficiently handle stationary regions of the signal. In this respect, a transient model based on Prony's method would inherently fuse well with a sines + noise model applied to the stationary regions. Indeed, it would yield a characterization of the transients based on parameters very well suited for an *additive synthesis* reconstruction of the signal.[1]

Finally, another motivation for the use of Prony stands in the fact that transients are short by definition; and the efficiency of Prony's method is known to be remarkable on very short windows.

NB: One may want to look at section 3.2 for a detailed comparison between Prony's method and Fourier.

[1] As we will see next, those parameters are: frequencies, amplitudes, phases and damping factors

Chapter 3

Prony in stationary cases

The present chapter introduces the Prony parametric modeling of a very short frame of the signal, considered stationary. The Prony model can handle the case of stationary stochastic processes, but is not able to deal with highly non-stationary ones, such as the ones described in section 1.1. The topic of the next chapter (on page 63) will be to show the different ways one could extend the stationary case to the non-stationary one; i.e. how one could deal with the entire region of interest (the actual transient). But let's first focus on the first step: the stationary case.

The next illustration stands for a general view of this chapter's contents.

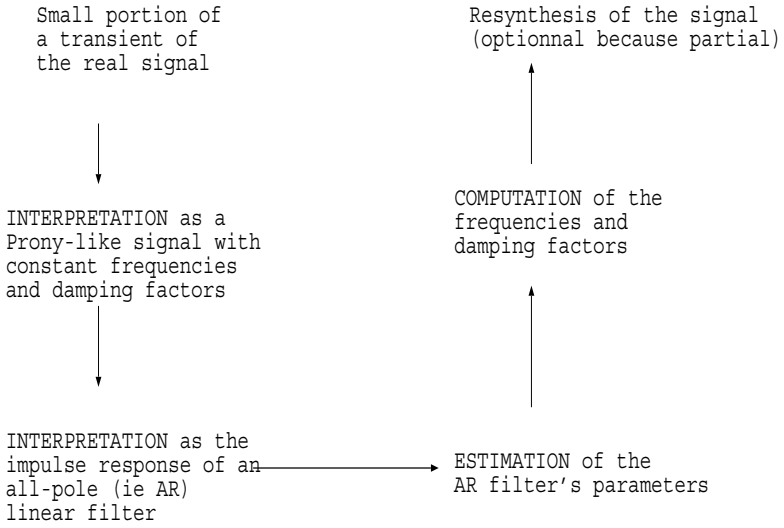


Figure 3.1: Illustration of the Prony algorithm in stationary case

3.1 Algorithm, Presentation of the Prony model

Here are our algorithm's progressive steps, they will obviously make more sense to the reader within the development of the next paragraphs, but it seems important to summarize the algorithm at the very beginning as a section one could refer to at any time of the reading:

- Interpretation of the musical signal $x(n)$ as an assumed model {Prony: sum of damped sinusoids + observation noise} + a modeling error
- Estimate the order $2L$ of the Prony model
- Choose the parameters p and N and form the matrix M from the signal originals' samples
- Reduce the observation noise, so that $x(n)$ be interpreted as a sum of damped sinusoids (the basic Prony model) + a modeling error
- Minimize the modeling error so that $x(n)$ be as close as possible to a sum of damped sinusoids. (i.e. compute the vector $\underline{a}$ such as $\|M \times \underline{a}\|^2$ be minimized)
- Form the polynomial $A(Z)$ with the knowledge of $\underline{a}$, compute its roots and select among them the ones corresponding to the L sinusoids in the signal

We will also discuss the different theoretical approaches and possible interpretations of the modeling error and the observation noise.

3.1.1 Introduction to the Prony model

In the Prony model, the idea is to model one frame of N samples of the signal by a sum of L exponentially damped sinusoids (i.e. $2L$ complex exponentials) and to determine their optimal parameters from the original signal's samples. But the Prony model is basically a deterministic model, as we will want to deal with real signals (that can be considered either as deterministic signals or realizations of stochastic processes), we introduce a version of the Prony model that accounts for an observation noise ($w(n)$); the addition of a latter disruptive term $e(n)$ is not part of the model, but stands precisely as the modeling error. The interpretations of these additive terms are important issues in this work. We will see that the term $w(n)$ (observation noise) can be considered as a stochastic white noise ; and the term $e(n)$ (modeling error) can be apprehended within a deterministic framework or a stochastic one.

Eventually, a Prony-like signal will be interpreted as the impulse response of an Auto Regressive (AR) filter.

$$x(n) \equiv \left(\sum_{m=1}^{2L} B_m * Z_m^n \right) + e(n) + w(n) \tag{3.1}$$

where $B_m = A_m * e^{i\theta_m}$ and $Z_m = e^{-\alpha_m} * e^{i2\pi\widetilde{f_m}}$

Thus the parameters of the Prony model are:

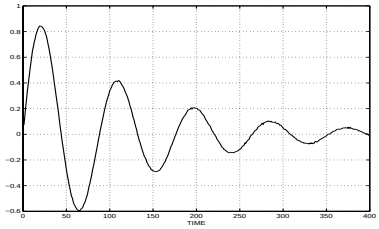
- $\widetilde{f_m}$: the normalized frequency
- α_m: the damping factor
- A_m: the amplitude
- θ_m: the phase

NB: The link between the normalized frequency and the frequency is $\widetilde{f_m} = \frac{f_m}{F_s}$ where F_s is the sampling frequency.

Figure 3.2 is an example of a very straightforward Prony-like signal, composed by only one exponentially damped sinusoid added to a white noise. Features are: $SNR = -60dB$, $F_S = 44.1kHz$, $N = 400$ samples, $f_m = 500Hz$, $\alpha_m = 0.008$, $A_m = 1$, $\theta_m = 0$.

As it is a synthetic signal, there is no modeling error.

Figure 3.2: Example of a straightforward Prony-like signal



Here is what one should keep in mind in reading what is detailed in section 3.1.1:

1. The theoretical case of a pure L-sinusoidal signal without observation noise, handled by the basic deterministic Prony model (without disruptive terms), permits us to introduce (on page 38) how one can estimate the values of the Prony model's parameters.

2. In the case of a real signal (issued from actual musical instruments), whether one chooses to interpret it as a deterministic or a stochastic process, the *estimation method basically stays the same* but is adapted

so that the real signal may be approximated by another one corresponding to the {deterministic + observation noise} Prony model, i.e. $\widehat{x}(n) = \left(\sum_{m=1}^{2L} B_m * Z_m^n\right) + w(n)$. (see page 40)

Thus, the underlying goal in this work is to *estimate the values of a non-noisy L-sinusoidal signal's parameters, so that this signal be as close as possible to the actual signal which observation noise has been removed.*

3.1.1.1 Approach without the observation noise

3.1.1.1.1 Basic deterministic Prony model Say the signal is solely made of damped sinusoids. We have:

$x(n) = \left(\sum_{m=1}^{2L} B_m * Z_m^n\right)\ \forall n \geq 0$ where Z_m are defined as on page 37.

One can show that there exists a vector $\underline{a} = (a_0, \cdots, a_p)$ where $a_0 = 1$, such as

$$\begin{aligned}\prod_{m=1}^{p} (Z^{-1} - Z_m) &= \sum_{k=0}^{p} a_k * Z^{-k} \\ &= A(Z)\end{aligned}$$

$$\Longrightarrow \sum_{k=0}^{p} a_k * x(n-k) = 0\ (\forall n \geq p+1) \tag{3.2}$$

(NB: $a_0 = 1$)

Therefore, the model is entirely determined if we know the p first samples of the signal.

let's define a matrix called the "signal matrix":

$$M = \begin{vmatrix} x(p) & \cdots & x(0) \\ \vdots & & \\ x(N) & \cdots & x(N-p) \end{vmatrix} \tag{3.3}$$

$$\Longrightarrow M \times \underline{a} = \underline{0}$$

Here, the resolution of this equation is quite straightforward. We are looking for a vector $\underline{a}$ in the null-space of the matrix M. Provided that $p > 2L$, one can show (see [Henderson] for more details) that this matrix is singular, and that its rank is $2L$ (L being the number of sinusoids). M and $M^t \times M$ (which is square)

have the same null-space, so one can perform a Singular Value Decomposition (SVD) over either one of those matrices, and thus obtains its null-space.

NB: It is important to emphasize the previous classical linear algebra result: for any matrix A, A and $A^t \times A$ have the *same null-space*.

Eventually, *any vector from the null-space of these matrices is a solution* to our problem, and the computation of the roots of the polynomial defined by its elements yields to the frequencies and the damping factors of our sinusoids.

The equation (3.2) also permits us to introduce the basic statement of this work: a Prony-like signal can be interpreted as the impulse response of an Auto Regressive (AR) filter:

Let's interpret our signal $x(n)$ as a realization of a stochastic process $X(n)$. One can show that starting from the equation (3.2), the explicit handling of the autocorrelation function $R_{XX}(m) \equiv E\left[X(n) * X(n-m)\right]$ leads to:

$$R_{XX}(m) = -\sum_{k=1}^{p} a_k * R_{XX}(m+k) \ (\forall m \geq p+1) \tag{3.4}$$

Which are known as the Yule-Walker equations (see Appendix 1 page 90) one has to deal with when one wants to estimate the parameters of an AR(p) model which can be illustated as in figure 4.5.

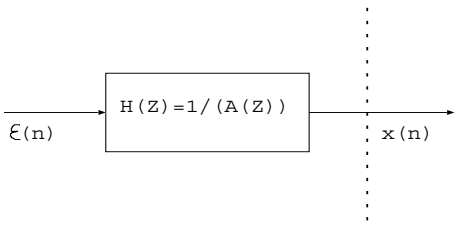


Figure 3.3: AR model illustration

In the linear systems framework, $\epsilon(n)$ is the innovation of the AR process. In our case, $\epsilon(n)$ is an impulse.

Eventually, in the theoretical case of a basic deterministic Prony model, we introduced the AR processes framework; which has first been done by Grenier in [Grenier2]: "$x(n)$ can be considered as the output of a linear system with input zero stochastic initial conditions". In the following of this work, we will make the asumption that *a Prony-like signal can be interpreted as the impulse response of an Auto Regressive (AR) filter.*

NB: For more details on this asumption, see Appendix 4 page 96.

3.1.1.1.2 Deterministic case + modeling error Let's now focus on how we can adapt the SVD method introduced above in the case of real signals that are not made of a precise number L of damped sinusoids.

In the case of real signals without observation noise (or from which it has been removed as detailed in paragraph 3.1.1.2), we have:

$x(n) \equiv \left(\sum_{m=1}^{2L} B_m * Z_m^n\right) + e(n)$

One could make two different interpretations of the modeling error $e(n)$, and have a stochastic approach or a deterministic approach. In any ways, we will see that these two approaches lead to a single common resolution, that intend to compute the vector $\underline{a}$ such as $\|M \times \underline{a}\|_2$ be minimized, in this step, the error of model is minimized so that $x(n)$ be as close as possible as a pure sum of damped sinusoids , this is where we *fit our model to the actual signal.*

Deterministic approach For real signals, the matrices M and $M^t \times M$ are always non-singular. Thus the previous method (page 38) that could easily determine their null-space cannot hold without further information.

The signal is interpreted as a sum of L sinusoids added to a disruptive term. Provided that an estimation scheme can determine the number L (see subsection 3.1.4.1), one can highlight what would have been the null-space in the previous case (see Appendix 3 about truncated SVD on page 94).

Then comes the problem of the choice of a vector in this null-space (let's remind the reader that in the theoretical case, any vector in the null-space was convenient). Several methods can be tried (see section 3.1.2), what we did implement is the truncated SVD method introduced in [Kum./Tufts] (see Appendix page 94) that, given the order L, minimizes the norm of the vector $\underline{a} \equiv (a_0 \cdots a_p)$.

Let's go back to the equations, here we have $x(n) = \left(\sum_{m=1}^{2L} B_m * Z_m^n\right) + e(n)$

A derivation similar as the one previously introduced (in equation (3.2)) leads to:

$$\Longrightarrow \sum_{k=0}^{p} a_k * x(n-k) \begin{array}{l} = \sum_{k=0}^{p} a_k * e(n-k) \ (\forall n \geq p+1) \\ \equiv \epsilon(n) \end{array} \tag{3.5}$$

$$\Longrightarrow x(n) = -\sum_{k=1}^{p} a_k * x(n-k) + \epsilon(n)$$

Starting from this equation, the basic idea in Linear Prediction Error (LPE) method is to predict an estimator $\widehat{x}(n)$ of $x(n)$ as a linear function of the "past of the signal", i.e. $x(n-1), x(n-2), \cdots$

$\widehat{x}(n) = -\sum_{k=1}^{p} a_k * x(n-k)$ NB : k starts at the index 1

The goal is to *minimize a distance* between both. This distance can be interpreted as:

$$\epsilon(n) = \|x(n) - \widehat{x}(n)\|$$

$\Longrightarrow \epsilon(n) = \sum_{k=0}^{p} a_k * x(n-k)$ NB : k starts at the index 0

One must define a criterion to mesure this distance:

$J_\epsilon = E\left[\epsilon^2(n)\right]$ or $J_\epsilon = \sum_{n=p}^{N} \epsilon^2(n)$ for example. N is the number of samples taken, its choice is explained farther.

Minimizing the distance according to a chosen criterion would be equivalent to look for the vector $\underline{a} \equiv (a_0 \cdots a_p)$ that would minimize the criterion J_ϵ.

If one makes the asumption of ergodicity of our stationary process (for a definition, see Appendix 5 page 102), and then chooses the second criterion, then,

$$\begin{aligned}
\text{Minimizing } J_\epsilon \equiv \sum_{n=p}^{N} \epsilon^2(n) &\Leftrightarrow \sum_{n=p}^{N} \frac{\partial}{\partial \underline{a}} \left(\epsilon^2(n)\right) = 0 \\
&\Leftrightarrow 2 * \sum_{n=p}^{N} \epsilon(n) \frac{\partial}{\partial \underline{a}} \left(\epsilon(n)\right) = 0 \\
&\Leftrightarrow \sum_{n=p}^{N} \epsilon(n) * x(n-j) = 0 \ \forall j = 0 \cdots p \\
&\Leftrightarrow \sum_{k=0}^{p} a_{k*} \left(\sum_{n=p}^{N} x(n-k) * x(n-j)\right) = 0 \ \forall j = 0 \cdots p
\end{aligned}$$

$$\Longleftrightarrow \text{ Minimizing } \|C \times \underline{a}\|^2$$

where C is the matrix defined by the elements:

$c_{i,j} = \sum_{n=0}^{N-p} x(n+p-i) * x(n+p-j)$ $(\forall i,\ j = 0 \cdots p)$.

This matrix is an *estimation of the covariance matrix* . One can show that this matrix is equal to $M^t \times M$, where M is the signal matrix defined on page 38.

As we saw on page 38, C and M have the same null-space, we eventually try to *find the minimum-norm solution to* $\|M \times \underline{a}\|^2$

Stochastic approach As stated by GRENIER in [Grenier2], "the prediction error can be viewed as the innovation of an AR process". Indeed, if one chooses to minimize the first criterion $J_\epsilon = E\left[\epsilon^2(n)\right]$, it can be shown that $\frac{\partial J_\epsilon}{\partial \underline{a}} = 0$ leads to the Yule-Walker equations we introduced above (see equation (3.4)).

But obviously, as one will never have at his disposal all the realizations of a stochastic process but only one of its realizations (the actual signal), the criterion defined by expectations will not be taken in the practical cases. If one assumes the *stationarity* of the signal, then one uses the ergodicity[1] of the process in order to estimate the statistical data, like the correlation coefficients, through time averages. The correlation function can't be computed, thus we won't have to deal with the Yule-Walker equations; but it can be estimated, and one can show that C is asymptotically the best estimator of the autocorrelation matrix.

[1] See definitions in appendix 5 page 102.

3.1.1.2 Dealing with the observation noise

As one will see in subsection 3.2.2.2, the performances of the Prony analysis depend very much on the level of noise corrupting the signal. Therefore, it is relevant to focus on ways to attenuate the effect of the noise.

The signal is assumed to be the output of a linear system with input zero-stochastic initial conditions, plus an additive white noise, this can be illustrated by figure 4.4:

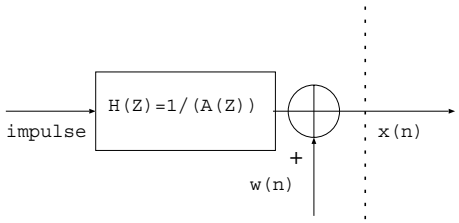


Figure 3.4: Prony model illustration in the deterministic+noise case

3.1.1.2.1 Noise compensation In the following paragraph, the method tested is the one that intend to reduce the effect of the noise by substracting an estimation of the white noise variance from the diagonal of the covariance matrix.

Experiment[2]:

- Generate a non-noisy Prony-like signal, estimate the covariance matrix and then the poles
- Display the estimations of the frequencies and damping factors (circles)
- Add white noise to the same signal, compute another estimation of the poles
- Display the new estimations of the frequencies and damping factors (dots)
- Estimate the variance of the white noise
- Remove the variance of noise from the diagonal of the covariance matrix' estimation, estimate the poles
- Display the last estimation of the frequencies and damping factors (crosses)

[2]Several plots corresponding to several realizations of the white noise are dispayed in figures 3.5 and 3.7.

With this experiment, we can compare the results of the estimation algorithm when the signal is non-noisy and when it's noisy. We can also test the goodness of our noise removal algorithm.

The original signal is made of 5 exponentially damped sinusoids.

Here are their parameters $[f, \alpha]$: [243Hz, 0.006], [500Hz, 0.008], [540Hz, 0.008], [1323Hz, 0.006], [2000Hz, 0.004]. One can see that two of them have close frequencies. (see figure 3.5)[3]

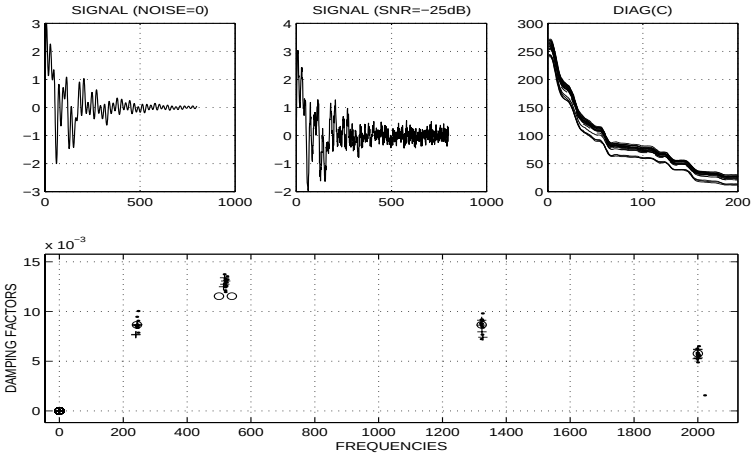


Figure 3.5: Observation noise issue: Sum of 5 damped sinusoids, estimated parameters.

In order to estimate the variance of the noise, we can try the method proposed by Laroche in [Laroche1]:

Say we have $x(n) = s(n)+w(n)$ where $x(n)$ is the noisy signal, $s(n) = \left(\sum_{m=1}^{2L} B_m * Z_m^n\right)$ the corresponding non-noisy signal and $w(n)$ the white noise which variance is σ_w^2. As the noise and the signal are not correlated, we can write $C_x = C_s + \sigma_w^2 \times I$.

One can show that the singular vectors of C_s and $C_s + \sigma_w^2 \times I$ are the same, thus if one could remove the effect of the noise on C_x (i.e. theoretically on its diagonal components), the SVD algorithm would yield the right estimation of the wanted parameters.

When the signal is a sum of sines + white noise, it is easy to determine the level of noise (look at figure 3.6).[4]

When the signal is natural, it doesn't correspond exactly to the model, there is no obvious boundary between the eigenvalues equal to σ_w^2 and the other

[3] The parameters are estimated over the signal without noise (o), the signal with noise (.) without any correction and with a correction on the estimated covariance matrix' diagonal (+).

[4] Y axis in dB

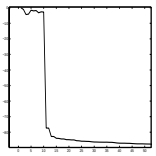


Figure 3.6: Observation noise issue: Eigenvalues of the estimated covariance matrix of a signal made of 5 sinusoids + a white noise.

eigenvalues. One could observe a regular decrease of the eigenvalues. One can choose (as in [Laroche1]) to estimate σ_w^2 as the smallest eigenvalue of C_x, besides it is the only option for C_s to stay positive definite.

Conclusions from figure 3.5:

- When there is no noise, the frequencies are very well estimated, and the damping factors are a little bit overestimated.
- When the noise is added, it is interesting to see that achieving the estimations over the noisy covariance matrix and the corrected covariance matrix seems to have the same performances, they both miss the close frequencies and replace them by a single intermediary one.

In the theoretical case, the covariance matrix C_x is not estimated and its elements are actual expectations, in that case, adding noise to the signal only has an effect on the diagonal of C_x. In the real case, the covariance matrix is estimated (expectations are estimated by averages over time), thus not only the diagonal but all the elements of C_x are effected by the noise. This effect is proportional to the inverse of the number of samples used for the averages. Thus we must face two problems:

1. We need a very good estimation of the noise's variance so that we can effectively remove its effect on the diagonal of C_x
2. The goodness of the estimation of C_x depends directly on the number of samples.

We estimate the expectations by averages over time, hence the non-diagonal terms are a little bit affected by the noise. Figure 3.7 shows us that even when the expectations estimations are very good (i.e. N is big) and when the effect of the noise on the diagonal is *exactly* removed (which was possible here but is not in real cases), the computation of the poles fail in the case of close frequencies.

As the method shows drawbacks in the case of large data (N=2000) and perfect knowledge of the noise variance, we can conclude that in real case where one has to estimate this variance (for example with the method explained above) and

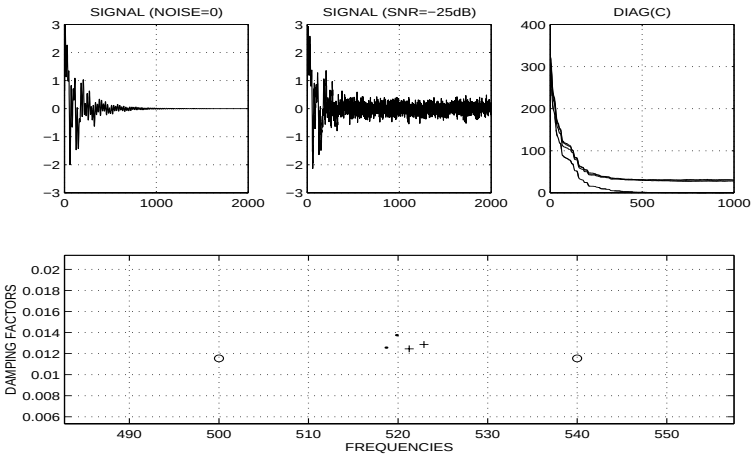


Figure 3.7: Observation noise issue: 5 sinusoids + a white noise

has constraints regarding the amount of data, the results won't be satisfying enough.

We have seen that the cases of close frequencies embedded in measurement noise call for new ways to reduce the effect of the noise, these are presented in the next paragraph. However, all frequencies that are spaced enough (see paragraph 3.1.4.4) can still be well estimated even for meaningful levels of noise. One can also show that increasing the order p has a good effect regarding the noise compensation (see paragraph 3.1.4.2). Finally, as it is shown in [Kum./Tufts]: "using an order $p \gg 2L+1$, the truncated SVD algorithm applied on moderately noisy data yields valid results".

3.1.1.2.2 Applying general ARMA estimators This approach detailed in [Kay] starts from the fact that the AR model is inconsistent for a noise corrupted AR process, and recognizes that *the true model for an AR(p) process embedded in white noise is an ARMA(p,p) model*. The AR filter parameters of the ARMA(p,p) model are identical to the ones of the AR(p) process. In respect with the spectral estimation theory developed in [Kay], in the linear system illustrated in figure 3.8, we have $Var(\epsilon(n)) = \sigma^2$ for the input and $Var(w(n)) = \sigma_w^2$ for the white noise added to $s(n)$. One can write the expressions of the Power Spectral Densities (PSD):

$$P_{SS}(Z) \equiv H(Z) * H^*(1/Z^*) * \sigma^2$$

$$P_{XX}(Z) \equiv P_{SS}(Z) + \sigma_w^2 \Longrightarrow P_{XX}(Z) = \frac{\sigma^2}{A(Z)*A^*(1/Z^*)} + \sigma_w^2 = \frac{[\sigma^2 + \sigma_w^2 * A(Z) * A^*(1/Z^*)]}{A(Z)*A^*(1/Z^*)}$$

Thus the PSD of $x(n)$ is characterized by zeros as well as poles. That is, the appropriate model for the observed process $x(n)$ is an ARMA(p,p) process.

One could use any ARMA estimation method to estimate the poles and the zeros of this model.

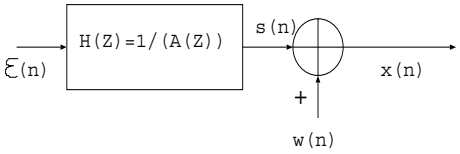


Figure 3.8: Observation noise issue: AR(p) process embedded in white noise

Thus before the addition, one can write:

$TZ\left(R_{SS}(n)\right) = H(1/Z^*) * TZ\left(\sigma^2 * \delta(n)\right) * H(Z) \Leftrightarrow s(n) = h(n) \star \epsilon(n)$

$\Longrightarrow P_{SS}(Z) = \frac{1}{A(1/Z^\star)*A(Z)}$

$s(n)$ is added to a stationary white noise $w(n)$ (standing as a realization of the stationary process W). One can define[5] the PSD of W:

$$\begin{aligned} P_{WW}(Z) &\equiv TZ\left(R_{WW}(n)\right) \\ &= \sigma_w^2 \end{aligned}$$

Thus one can write:

$$\begin{aligned} P_{XX}(Z) &= \frac{1}{A(Z)*A^\star(1/Z^\star)} + \sigma_w^2 \\ &= \frac{[1+\sigma_w^2 * A(Z)*A^\star(1/Z^\star)]}{A(Z)*A^\star(1/Z^\star)} \end{aligned}$$

One can see in this last equation that using an AR estimator is inherently non sufficient to get the parameters of $A(Z)$. But using an ARMA estimator would permit us to estimate the true AR parameters.

We did implement the Two-stage and Three-stage Least Squares (LS) methods detailed in [Kay] (see Appendix page 92 for the algorithm). This ARMA estimation algorithm rely on the estimation of the input.

Let's remind the reader that the final goal is to build an algorithm that would permit to estimate the parameters of the Prony model embedded in measurement noise (with very low SNR). However, as we are using an ARMA estimator for this task; thus we must first focus on estimation of ARMA parameters , then AR parameters of AR model embedded in measurement noise, and eventually Prony parameters of noise corrupted Prony signals.

ARMA estimation The Three-stage LS method yields good results for estimation of ARMA processes. An illustration can be seen in figure 3.9.[6]

[5] where $R_{WW}(n)$ is the autocorrelation function of the process W.

[6] Original (.- line), reconstructed ARMA using the 3-stage LS method (plain line).

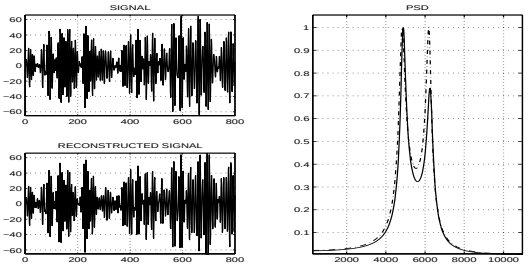


Figure 3.9: Observation noise issue: Temporal realizations and Power Spectral Densities of an ARMA(4,2) process.

AR + measurement noise estimation It is well known that the estimation of AR processes parameters is very sensitive to the addition of noise (we already saw that phenomenon in the example given in figure 3.7). In the case of two relatively close frequencies, at a relatively high SNR (eg SNR=-40 dB), it is seen (in figure 3.10) that even if the noisy signal look alike the non-noisy one, the addition of noise to the process yields a lost of one of the spectral peaks, moreover the remaining peak is displaced from its true position and broadened.[7]

One can see in our example that the Three-stage LS method yields a process that recovered the two spectral peaks. One could notice that its realizations are weaker than the original process' ones, but the important thing is that it permits us to *keep track of the correct AR parameters* and thus *the correct poles*.

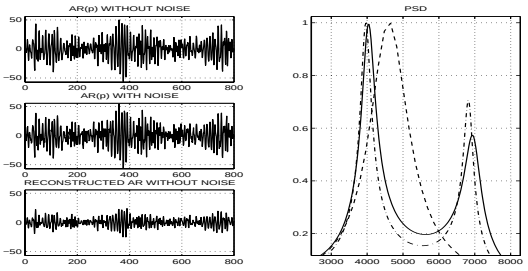


Figure 3.10: Observation noise issue: Temporal realizations and Power Spectral Densities of an AR(4) process.

In the cases of very low SNR (typically below -10dB), this method doesn't seem to be effective enough for the issue of close frequencies. However, one can see in figure 3.11 that it permits to recover the global shape of the PSD[8], and it

[7] Without noise (.- line), with noise SNR=-40dB (− line), and reconstructed AR(4) process using the 3-stage LS method (plain line).

[8] Without noise (.- line), with noise SNR=-10dB (− line), and reconstructed AR(4) process using the 3-stage LS method (plain line).

yields relevant AR parameters which, if not perfectly accurate, are still much better than the ones obtained by a classic AR parameters estimation over noise corrupted AR processes.

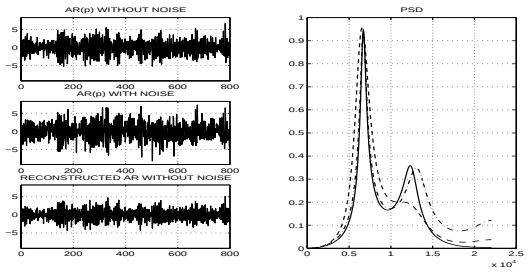


Figure 3.11: Observation noise issue: Temporal realizations and Power Spectral Densities of an AR(4) process.

The experiments on the Three-stage LS method (illustrated by figures 3.10 and 3.11) show that it seems to be better than a simple noise compensation algorithm (like the one detailed on page 42) for keeping track of AR parameters in AR + measurement noise cases. Thus, as a Prony-like signal can be interpreted as the impulse response of an Auto Regressive (AR) filter, one could think that this method would give good results for the estimation of Prony parameters in important measurement noise cases.

Prony estimation In the case where the Three-stage LS is tested on Prony-like signals (or impulse responses of AR filters) added with noise, we did get results much worse than the ones yielded by noise compensation algorithms. As this method showed to be actually better than the others, this should not be the case, further work must be done regarding this issue.

3.1.1.3 Conclusions

We've seen that the model fitting issue should account for a reduction of the effect of the observation noise $w(n)$, so that the signal $x(n)$ be interpreted as a sum of damped sinusoids (the Prony model) + a modeling error $e(n)$.

We've also seen that it is possible to consider the signal one wants to model in two different ways: as a deterministic process or as a realization of a stochastic process, that yields to different interpretations of the disruptive term $e(n)$.

Whether one chooses an interpretation or the other, it is shown that the actual resolution of the problem stays the same: we want to determine the vector $\underline{a}$ such as $\|M \times \underline{a}\|^2$ be minimized.

With respect to the interpretation of the disruptive terms $e(n)$ and $w(n)$, a further work could be done, introducing the *state-space model theory* that accounts for either deterministic and stochastic input. It would probably be the way to have the most complete theoretical approach. Looking further in this theory, the general problem would be the following: we have a state-space model with a deterministic input (an impulse) added to a state noise (stationary white noise input), and a measurement noise added at the output (a white noise background). Indeed, it seems clear that the measurement noise (error of the model and actual noise added to the signal -bad recording, ...), and the state noise (stochastic noisy nature of the sound -breathing in flute sounds, ...) should be treated separetly. So far in our algorithm, we use a deterministic impulse input, no state noise, and a measurement noise that we want to minimize.

3.1.2 Computation of the AR parameters (vector a)

The step prior to the determination of the Prony parameters (frequencies, damping factors) is the determination of an Auto Regressive (AR) model's coefficients: $\underline{a} = (a(0) \cdots a(p))$. This is done by finding the solution that minimizes $\|M \times \underline{a}\|^2$.

If $x(n)$ corresponded exactly to the Prony model (no modeling error and no measurement noise), then the matrix M would be singular of rank $2L$. One could take any vector $\underline{a}$ in the null-space of M, in that case $M \times \underline{a} = \underline{0}$.

In the general case of real signals, the goal is to find $\underline{a}$ such as the modeling error be minimized (subsequently to a measurement noise compensation algorithm described in paragraph 3.1.1.2).

The method we eventually choose for the minimization of $\|M \times \underline{a}\|^2$ is the truncated SVD algorithm introduced by KUMARESAN and TUFTS in [Kum./Tufts]. This method lies on the fact that M is non singular (because of the modeling error), but is "forced" to be so. Given the order $2L$, one can determine what would have been the null-space in the no disruptive terms' case, this subspace could be called the "forced null-space". This method intends to find a vector $\underline{a}$ in this forced null-space, solution to the minimization problem, that would be minimum norm, under the constraint $a_0 = 1$ (otherwise, the solution would be $\underline{0}$). More explanations regarding the relevance of the truncated SVD for our particular problem are given in Appendix 3 page 94.

One can define M' such as $M = (\underline{x} \mid M')$ where $\underline{x} = (x(N) \cdots x(p))^t$, and $\underline{a'}$ such as $\underline{a} = (1, \underline{a'}^t)^t$

The computation of the Singular Value Decomposition of M' yields:

$$[U, S, V] = SVD(M') \Longrightarrow M' = U \times S \times V^t$$

Where V is the matrix of the eigenvectors $\underline{v_k}$ of $M'^t M'$ and U is the matrix of the eigenvectors $\underline{u_k}$ of $M' M'^t$.

$S = \begin{vmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \\ 0 & & \sigma_p \end{vmatrix}$ is the diagonal matrix of the singular values of M'. (NB: in the noiseless case of a $2L$-Prony signal, the diagonal terms $\sigma_{2L+1}, \cdots, \sigma_p$ are equal to 0)

$M'\underline{a'} = \underline{x} \Leftrightarrow USV^t\underline{a'} = \underline{x}$

$$\Rightarrow \underline{a'} = -\sum_{k=1}^{2L} \sigma_k^{-1} * \left[\underline{u_k^t} * \underline{x}\right] * \underline{v_k}$$

The truncation takes place in the index of that sum: it stops at $2L$. This method makes the use of the $2L$ principal eigenvectors of M^tM and MM^t which are more robust to the noise perturbation in the data. Eventually, $\underline{a}$ is of dimension p. The parameters a_k are zeros of the polynomial introduced in equation (3.2). Because p is chosen to be greater than $2L$ (for accuracy reasons), there are clearly more than $2L$ zeros, indeed the $p-2L$ extraneous zeros stand as artifacts of the method, and we need to get rid of them. It can be shown (in [Kum./Tufts]) that the $p-2L$ extraneous zeros will always fall *in the unit circle*. Thus, a method for identifying the signal zeros from the extraneous zeros could be to use the data in the *reverse time direction*. If the signal corresponds to the equation (3.1), the backward data does consist of growing exponentials, i.e. corresponds to an unstable filter. That way, it can be shown (see [Kum./Tufts]) that the signal zeros will always fall *outside the unit circle*, which makes them more easy to discriminate.

3.1.3 Computation of the frequencies and damping factors

As defined in section 3.1.1, $Z_m = e^{-\alpha_m} * e^{i2\pi\widetilde{f_m}}$.

We can visualize it on a figure of the unit circle:

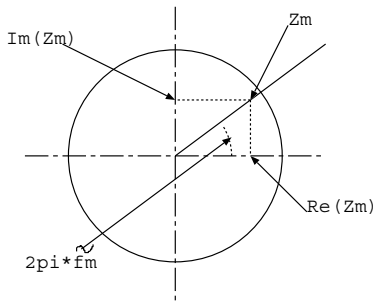


Figure 3.12: Unit circle, representation of a pole

As introduced in equation (3.2), the Z_m are roots of the polynomial $A(Z)$ formed by the parameters a_k estimated in the previous section. We've seen that a choice must be made within the p roots computed, and also that the signal poles (the ones we want to keep) fall outside the unit circle. Thus for each pole which module is greater than 1, we compute α_m and f_m as:[9]

- $\alpha_m = \ln(|Z_m|)$
- $f_m = \tan^{-1}\left(\frac{F_s}{2\pi} * \frac{Im(Z_m)}{Re(Z_m)}\right)$

The final number of zeros (frequencies and damping factors) is $2L$.

3.1.4 Spectral resolution, choice of parameters L, p, N

Let's remind the reader that N is the number of samples of an analysis frame, L is the number of sinusoids in the definition of the Prony model, and p is the order of the corresponding AR model.

We will discuss in this section the reasons for the ones N is chosen relatively small (400 samples), L is given by an order estimation scheme, and p is chosen such as $p = \frac{N}{2}$ or $p = \frac{N}{3}$.

3.1.4.1 Estimation of L (number of sinusoids)

Order estimation methods can give us a clue on the number of sinusoids in the signal, but obviously these methods are not absolutely exact, therefore it may be interesting to first focus on the influence of the value of L on the analysis of synthesized signals (indeed it is a better way to have relevant conclusions than using real signals for the ones L is unknown).

We've tested several choices of L on signals composed by a known number of sinusoidal components.

As we will see, it is still possible to obtain good analyses without an exact determination of L. Eventually, our tendency will be to choose L systematically greater than its estimate.[10]

Overestimated L If one chooses L greater than the actual number of sinusoids, then the Prony algorithm will choose the vector $\underline{a}$ in the null-space[11] of M^tM, which dimension is smaller when L is chosen greater than its actual

[9] A simple look at the figure 3.12 justifies these formulas.

[10] It is by the way currently assumed that an overevaluation of any model's order is preferable to an underevaluation.

[11] This is in the basic theoretical model case, we've seen in page 40 that the scheme is to minimize $\|M \times \underline{a}\|^2$ in the real cases. The former is clearer for this explanation.

value. Thus the vector $\underline{a}$ is chosen in a sub-space of the actual null-space of M^tM; as it still pertains to this latter null-space, its corresponding polynomial still possesses among its roots the $2L$ zeros associated to the L sinusoids, solely the extraneous poles number has decreased.

Results of tests regarding this issue have shown that the overestimation of L (in a reasonable scale) doesn't yield damaging artifact on the estimation of the polynomial's true zeros.

Underestimated L On the other hand, if L is smaller than the actual number of sinusoids, the theory doesn't guarantee that the zeros (of the polynomial associated with the vector $\underline{a}$ computed) are going to correspond to the true sinusoids. Experiments have shown that the results of the poles estimations depend mainly on the nature of the signal.[12] Indeed, the underestimation of L can yield errors on the number of poles as well as huge errors on their values. As the nature of the signal is not a parameter one could choose, it seems obvious that an underestimation of the order L is not worth accepting.

Laroche's method The estimation method (introduced in [Laroche2]) of the order L is the following:

[12] The notion of conditionning introduced on page 54 gives theoretical explanations for the relation between the results of the estimation scheme and the number of sinusoids.

- Sort the eigenvalues σ_i^2 by decreasing order
- Find the minimum value of i for which the ratio $20\log\left(\frac{\sigma_0^2}{\sigma_i^2}\right)$ is less than $-DdB$, then take $2L = i$

Its meaning is that the musical signal will be assumed to contain a certain number $L(D)$ of sinusoids[13] in the amplitude range of $[0dB, \ -DdB]$, where D is a fixed floor. D can be chosen differently according to the signal; a value of $-80dB$ seems to be reasonable. This means that any possible sinusoid which power is $80dB$ below the power of the most prominent sinusoid will be neglected as a sinusoid, and will be considered part of the noise term.

Akaike criterion One could think about applying the Akaike criterion in order to find the best order for the model. This should be part of some future work.

3.1.4.2 Determination of N and p

N and p determine the size of the signal matrix (see page 38).

It is shown in [Laroche1] that N is not a vital parameter for the estimation scheme performances. Moreover, one must understand that it is not a parameter over the one we can have a wide choice, indeed, it corresponds to the size of the analysis window, which must be small in order for the stationarity asumption to make sense. The power of Prony method is most remarkable when the analysis is performed over short windows, indeed it shows much better spectral resolution than other spectral estimation techniques (as shown in chapter 3.2). For the purpose of modeling transients, it is vital to consider very short windows so that the signal can be considered stationary over them. The value $N = 400$ seems to be relevant.[14]

In [Henderson], it is shown that p must be chosen such as $2L \leq p \leq N - 2L$. We will see in the next paragraphs that there are several reasons to choose $p \gg 2L + 1$, around $\frac{N}{2}$.

First of all, it is shown in [Kum./Tufts] that using a large order p has as consequence that the $p - 2L$ extraneous pole of $A(Z)$ tend to be less perturbed, thus the discrimination is easier between the $2L$ meaningful poles one wants to keep and the $p - 2L$ other ones that are artifacts of the method. We will also see in the following that using an order $p \gg 2L + 1$ increases the accuracy of the poles locations estimates and improves the conditioning and the spectral resolution.

Moreover, one should notice that choosing an order $p \gg 2L + 1$ introduces the case of long AR model. Long AR models are presented in [Kay] and in

[13] Real signals do contain the modeling error term introduced in equation (3.1), thus the proper term would be "pseudo-sinusoids".

[14] With a sampling frequency $F_s = 44.1kHz$, this corresponds to approximatively $9.1ms$.

GRENIER's work as ways to estimate ARMA models. Thus, as we already know that an ARMA(p,p) process is the true model for a noise corrupted AR(p) process, *increasing the order of the AR model leads to better estimations of the poles.*[15]

3.1.4.3 Conditioning of the signal

If M^tM is non-singular, then one can define the conditioning of the signal as the ratio between the greatest and the smallest eigenvalue of M^tM:

$$\kappa = \frac{\sigma^2_{max}}{\sigma^2_{min}}$$

If M^tM is singular of rank $2L$, then one can define the pseudo-conditioning:[16]

$$\kappa = \frac{\sigma^2_1}{\sigma^2_{2L}}$$

An ill conditioning means a high number κ, and a good conditioning means a value of κ near 1.

LAROCHE has shown in [Laroche1] that the conditioning depends on p as well as on the number of sinusoidal components of the signal and their parameters.[17] According to him:

- For a value of p fixed, κ increases with the number of sinusoids.
- For a value of p fixed and a number of sinusoids fixed ($L = 2$), κ increases as the difference of frequencies decreases (i.e. as the sines get closer).
- For a fixed signal parameters, κ decreases as p increases.
- κ has a limit when p increases, which is the dynamic range of the signal (which depends on all its parameters).

Say one has to analyze a given signal and determines a value of p, to those corresponds a value of the conditioning κ. The least squares problem one has to solve in order to find the vector $\underline{a}$ depends on the conditioning κ of the matrix M^tM. If κ is high, then it is shown in [Laroche1] that small changes in the signal matrix (for example small additions of random noise on the signals samples) can yield important variations in the solution $\underline{a}$, and thus on the Prony parameters.

Thus, the notion of conditioning is very important regarding the performances of the Prony algorithm.

[15] For the estimation of the true ARMA process parameters, a better scheme than the long AR one is presented in 3.1.1.2.2.

[16] The eigenvalues are sorted out in decreasing order.

[17] Frequencies, damping factors, amplitudes and phases

3.1.4.4 Frequency precision

The spectral resolution of Prony's method depends on the values of the parameter p as well as on the nature of the signal itself and therefore cannot be described as simply as in the case of the FFT method.

An important work by LAROCHE (see [Laroche1]) has been done regarding the spectral resolution of the Prony technique. It emphasizes the relation between the value of the conditioning κ and the critical level of noise in the signal that permits to separate two sinusoids (meaning analyze them as two -and not solely one- sinusoids with different frequencies). He showed that κ contains enough information on the signal to be able to determine this critical SNR.

His conclusion is the following: For a certain value of κ, one can determine[18] the minimal value of SNR (i.e. a maximal value of noise) acceptable that still allows to separate two sinusoids, this is *independent of their respective frequencies and of the number p*. It is here important to focus on the fact that a same value of κ can correspond to different values of p and different sinusoidal components. The dependence between κ and the critical SNR is shown to be linear: if κ increases, then the acceptable noise level decreases (i.e. the critical SNR increases).

Unfortunately, the conditioning is not a parameter one could determine. However, for a given signal, κ decreases (i.e. gets better) as p increases. Thus one has the power to improve the spectral resolution of Prony method by setting a high parameter p. Yet, p suffers a limit that depends on the size of the window which has to be chosen small (as we've seen in paragraph 3.1.4.2).

Eventually, we've seen that the frequency precisison depends on the signal itself, and the way one has to improve it is not very powerful. Besides, for a given value of the conditioning, one can determine the maximal level of noise allowed for keeping track of all the sinusoidal components. But unfortunately, the level of noise is fixed by the sound source of the analyzed signal and the conditions of recording. Therefore, it seems that the overall attitude one may adopt towards the goal of a good spectral resolution should be to first find ways to *reduce the effect of the noise on the analysis algorithm*. Then one may choose a high parameter p, and systematically choose a parameter L a little greater than its estimate.

The next section gives some experimental results of the Prony method, as well as comparisons with Fourier techniques. The analyses of noisy signals are achieved without algorithm of additional noise's compensation so that one can focus on the performances of the basic Prony method. For explanations and results concerning the noise issue, one should see subsection 3.1.1.2.

[18] On a graph displayed in his thesis. It is not necessary to show it here.

3.2 Comparison with Fourier analysis

3.2.1 Theoretical point of view

First of all, it is important to remember that Prony is a parametric *model* of the signal, whereas Fourier techniques don't model the signal but *represent* it in a practical way for spectral analysis issues. The spectral representation of Fourier techniques, unlike the Prony modeling, allows to visualize what in the signal is not purely sinusoidal; however, it doesn't allow to parameterize it. Section 2.1 focuses on this basic difference.

In the theoretical definition of the Fourier transform,[19] the index in the sum is infinite, whereas it is finite in Prony's basic definition.

However one may notice that, in practice, analysing the signal in term of a sum of sinusoids recalls the Discrete Fourier Transform (DFT).[20]

The differences between DFT and Prony stand in:

- The sum is indexed by frequencies in Fourier, whereas in Prony it is indexed by the number of sinusoids one assumes are components of the signal.
- The frequency precision in the Fourier techniques is directly linked to the windowing of the signal, it implies that the frequencies one may find are multiples, unlike in Prony.
- There is no damping factor in Fourier techniques, nor is there a phase parameter.

Moreover, from a resynthesis point of view, Prony method is better than Fourier's. Indeed, one possesses more parameters, very relevant about the signal, that are directly usable, whereas using an algorithm based on Fourier technique, one has to process a subsequent peak detection scheme (interpolations algorithm for example can be quite complex).

NB: However, one should always keep in mind that a very restrictive asumption is being made in the use of any parametric method (see section 2.1).

3.2.2 Comparison over synthesized data

Here, we are not dealing yet with the relevance of our method for the characterization of musical data, the point is to compare it with Fourier techniques, which justifies the use of synthesized data.

[19] $x(t) = \int_{-\infty}^{+\infty} X(f) * e^{(+i2\pi f t)} df$

[20] $x_k \equiv \sum_{l=0}^{N-1} {}_l X_l * e^{\left(+i2\pi \frac{k*l}{N}\right)} \ \forall k = 0 \cdots N-1$

Hence, the signals we want to compare Prony and Fourier techniques on are synthetic signals, generated with a MATLAB program (a very straightforward example is given in figure 3.2).

It seems relevant to account for different measurement noise levels. Indeed, possible noise may be indicative of real signals, thus the analysis technique must be robust to it.

Moreover, it is assumed that real transient signals are very unstable, in the sense that "a lot of things may happen in a short time", and it is sensed that the frequential components may evolve in extreme ways. Thus, over one short frame of signal (considered stationary), one may want to be able to recover frequential components that die fast, as well as components which frequencies are very close. It seems relevant to account for these features in the synthesized signals.

In all the following examples, $F_S = 44.1$kHz, and $N = 400$ samples. Which leads in the Fourier framework to a frequency precision of $\Delta f \simeq 110$Hz (sic!). However, in order to have a better precision, we computed the Fourier transforms using zero-padding. It is very important for our particular problem to account for *very short windows length* as we want to be able to consider the signal stationary over these windows.

3.2.2.1 Non-noisy signals

3.2.2.1.1 Close frequencies issue An important feature of the signal we built lies in the presence of very close frequencies. It may be important to account for this particular case as it is known to be an important feature of acoustic signals. Indeed, the presence of close frequencies in a sound yield an effect called beating effect[21] which is very characteristic of certain instruments. Particularly, piano and cymbals sounds contain beating effects that turn out to be essential to the natural and live character of those sounds. If they are artificially suppressed, the sound becomes obviously synthetic to one's ears.

In figure 3.13, one can see the illustration of the data detailed in table 3.1. The frequencies found by the Prony method are called "Prony frequencies". The damping factors estimations are all equal, all a little bit overestimated. In this example, two pairs of frequencies are very close (differences of 2 Hz).

One can see that the spectral peaks issued from the Fourier analysis of the signal are very broad, and that it is impossible to figure if they correspond to one or several components. The comparison with Prony techniques shows that Prony can have very accurate results where Fourier lacks precision.

[21] What is heard is a beating in amplitude. One actually hears an intermediate sinusoidal component which amplitude vary according to a frequency equal to the difference of the two close frequencies.

	original frequencies (Hz)	Prony frequencies (Hz)
Sine #1	320	320.00
Sine #2	500	500.93
Sine #3	502	503.11
Sine #4	570	569.99
Sine #5	710	710.00
Sine #6	790	789.99
Sine #7	950	950.00
Sine #8	952	952.00

Table 3.1: Prony performances: 8 different frequencies, equal damping factors

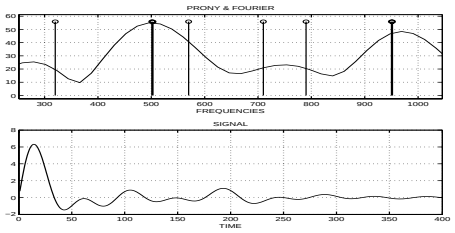


Figure 3.13: Prony performances: comparison #1 with Fourier. 8 different frequencies, equal damping factors

3.2.2.1.2 High damping factors issue Another case one wants to account for is the presence of very high damping factors.[22] In other words, it seems important to be able to detect sinusoidal components that die fast.

	original damping factors	Prony damp. fact.
Sine #1 (1000Hz)	0.008	0.0115
Sine #2 (3000Hz)	0.005	0.0072
Sine #3 (3200Hz)	0.03	0.0433
Sine #4 (3800Hz)	0.08	0.1154
Sine #5 (4100Hz)	0.004	0.0058
Sine #6 (4305.3Hz)	0.004	0.0058
Sine #7 (4500Hz)	0.006	0.0087
Sine #8 (8000Hz)	0.009	0.0130

Table 3.2: Prony performances: 8 different frequencies, different damping factors

The next figure (3.14) displays these results.[23]

[22] A high damping factor corresponds to a component that dies fast.

[23] The number on the Y axe are not directly readable for the damping factor, the level 50 corresponds to a factor equal to 0. The important in this figure is whether the component is detected.

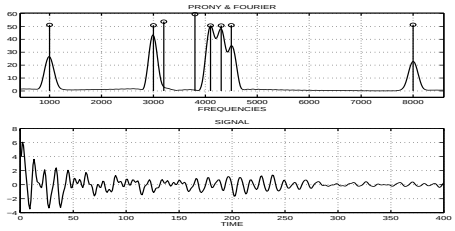


Figure 3.14: Prony performances: comparison #2 with Fourier. 8 different frequencies, different damping factors

One can see that the frequencies are very well estimated, and that the damping factors are a little bit overestimated, but stay in a reasonable scale. One can notice that components that die fast (the third and forth sinusoids) are not located by a Fourier analysis.

Eventually, the general conclusion one can make is that, in absence of measurement noise, the frequencies and damping factors are very well estimated, and the accuracy (specially in the case of very close frequencies and high damping factors) of the results is much better than the one with Fourier techniques.

3.2.2.2 Noisy signals

In the following examples, two different SNR have been tested and lots of experiments have been done with those levels of noise. The results displayed in the tables correspond to average values computed over 15 realizations, whereas the figures correspond to single realizations chosen for their representativity.

It is important to emphasize that for those particular examples *no algorithm of noise compensation* has been added to the Prony method described above. As ways to deal with the measurement are numerous (see subsection 3.1.1.2), it seemed relevant to focus on the results yielded by "raw" Prony analyses.

3.2.2.2.1 High damping factors issue The parameters of the signal used for conducting experiments relating to the issue of high damping factors are detailed in table 3.3. Likewise, the estimations of frequencies and damping factors are displayed in the same table. An illustration is given in figure 3.15.

Analysing these results yields to the conclusion that an important level of noise[24] doesn't permit the Prony algorithm to find components that die fast. Apart from that, the number of components doesn't seem to decrease, however, the parameters' estimations deteriorate with the amount of noise.

[24] high SNR = -40dB and low SNR = -5dB

	original damp. factors	Prony damp. fact.: snr = -40dB | -5dB
Sine #1	0.008	snr1 0.0115 | snr2 0.0080
Sine #2	0.005	0.0071 | 0.0079
Sine #3	0.03	0.0479 | none
Sine #4	0.08	0.0777 | none
Sine #5	0.004	0.0059 | 0.0053
Sine #6	0.004	0.0059 | 0.0059
Sine #7	0.006	0.0086 | 0.0079
Sine #8	0.009	0.0130 | 0.0061

	original frequencies (Hz)	Prony freq.: snr = -40dB | -5dB
Sine #1	1000	snr1 999.5 | snr2 986.2
Sine #2	3000	2999.4 | 2966.8
Sine #3	3200	3208.4 | none
Sine #4	3800	3804.0 | none
Sine #5	4100	4099.8 | 4106.5
Sine #6	4305.3	4304.7 | 4316.5
Sine #7	4500	4499.0 | 4509.8
Sine #8	8000	7999.7 | 8154.9

Table 3.3: Prony performances: 8 different frequencies, different damping factors. Averages of estimations for a high SNR (-40dB), and a low SNR (-5dB)

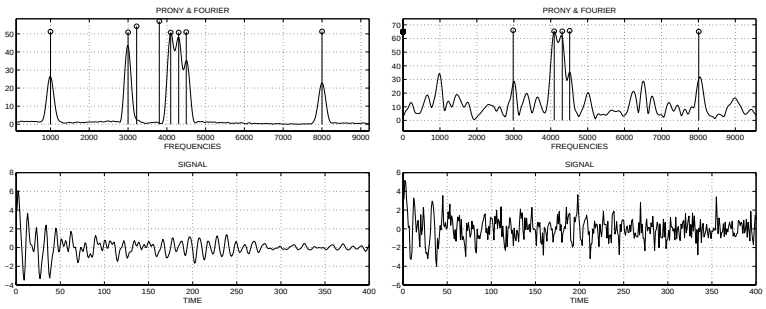


Figure 3.15: Prony performances: comparison #3&4 with Fourier. 8 different frequencies, different damping factors.

3.2.2.2.2 Close frequencies issue What one can see in figure 3.16 are two realizations of the signal, one for SNR=-40dB, and the other for SNR=-5dB. The signal is made of eight sinusoids which frequencies are given in table 3.4, with equal damping factors (= 0.008).

	original frequencies (Hz)
Sine #1	320
Sine #2	500
Sine #3	502
Sine #4	570
Sine #5	710
Sine #6	790
Sine #7	950
Sine #8	952

Table 3.4: Prony performances: 8 different frequencies, same damping factors

For this example, it is not relevant to display the results of the estimations in a table anymore. Indeed, experiments show that in the case where SNR=-40dB, the algorithm yields:

- A first frequency around 300 Hz
- One frequency around 525 Hz
- Another one around 750 Hz
- A last one around 950 Hz

In the case where SNR=-5dB, the algorithm yields:

- One frequency around 525 Hz
- One around 950 Hz
- In approximatively one realization over three, one around 750 Hz and 300 Hz

Hence, in this example, it seems that the estimated number of components deteriorates with the amount of noise. Moreover, the values of their parameters seem to suffer an averaging effect. Indeed the estimated frequency around 525 Hz may be an average between the sine #4 (570 Hz) and both sines #2&3 (500 & 502 Hz), likewise the estimated frequency around 950 Hz may be an average between the sine #7 (950 Hz) and the sine #8 (952 Hz).

In figure 3.16, the frequency axis of the low SNR plot is wider than in the previous figures. That is in order to focus on the fact that at a high level of noise-corruption of the signal, the Fourier analysis yields results that are not readable

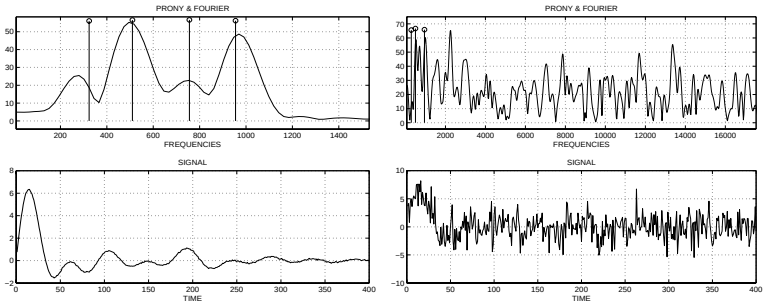


Figure 3.16: Prony performances: comparison #5&6 with Fourier. 8 different frequencies, same damping factors.

because of the height of the noise peaks. Prony also yields bad results (the close frequencies are averaged, moreover, the number of components decreases), however, the frequencies that are found correspond approximatively to actual components of the signal that wouldn't stand out against the background of noise peaks in a Fourier analysis.

3.2.2.3 Conclusions

Analysing these results, one can see that there are big differences between the performances of the Prony analyses depending on the presence and the level of measurement noise. When the signal doesn't suffer an addition of measurement noise, the Prony analysis gives very good results that are much more accurate than the ones given by a Fourier analysis.[25] However, even a small amount of noise causes uneven results, depending on the realizations of the noise.

An important conclusion to make is that there is a need for a reliable and efficient algorithm that would permit to reduce the effect of the noise on the estimation method.

These examples cover only parts of all the indicative particularities of signals one may want to analyze, however, they are adequate to conclude that the Prony technique seem very promising and much more appropriate for our purpose than a Fourier technique, but it calls for an efficient way to handle the noise problem in order to reach the level of precision required by our purpose.

We've seen in subsection 3.1.1.2 that several ways to reduce the effect of the noise on the analysis are conceivable. Some more work must be achieved regarding this issue, particularly on the estimation of ARMA models parameters (as explained in paragraph 3.1.1.2.2), which seems to be much more efficient that a simple noise compensation algorithm (as the one used in [Laroche1]).

[25] As it was sensed in section 2.1.

Chapter 4

Prony applied to non-stationary cases

As explained in chapter 3 introduction, here we want to deal with the entire transient regions of musical signals. An important feature of these regions is their strong non-stationarity. It can be understood as the fact that the descriptive parameters of the signal are very unstable. In order to analyze this non-stationarity and to be able to reproduce it in relevant parameters, one can extend the previous Prony method in different ways.

As introduced in the preceding chapter, one can see the signal as the output of a linear system disrupted by a measurement noise. The determination of the Prony parameters in the stationary case standed in the estimation of the filter's coefficients. One could assume that the process is stationary over short windows of signal, and then use directly the model described in the previous chapter. Then, one would be using a *discrete method* (see [Quat./McAul.]) as described in section 4.1. A way to "unstationarize" the process could be to apprehend the filter as a time-varying system, i.e. which coefficients are varying with time, this would be an *evolutionary model* (see [Molin./Castanie], [Grenier1] and [Grenier2]). This is the approach we will describe in section 4.2. Another way could be to have a non-stationary input running into a linear non-varying system (see [Yvetot] and [Atal]), this type of method is introduced in section 4.3. In this "input→system→output" framework, another way to handle the problem could be to have both non-stationarities (an evolutionary model with a non-stationary input).

4.1 Discrete methods

Representing the parameters' evolution could be done in dividing the transient region in K several shorter windows on which one will achieve an analysis -just

like described in the previous chapter-, assuming that the signal is stationary enough over each window. In this case the transient is eventually represented by time-varying parameters which values are known at instantaneous discrete index t_i $\forall i = 1, \cdots, K$.

This type of method, often called *discrete methods*, are usual in speech processing and music processing. In this respect an important paper is the one by Mc-Aulay & Quatieri ([Quat./McAul.]), which gives clues on the difficult problem of the tracking of different partials. In the computer music world, the way the frequency partials are handled in a SMS analysis is an example of discrete method. Figure 4.1 shows roughly how partials are linked between successive windows according to sequential decisions, and how death and birth of partials are considered.

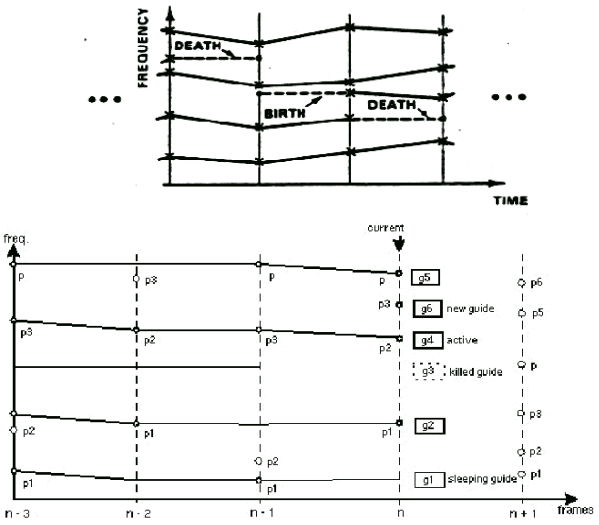


Figure 4.1: Non-stationarity: Examples of a partials tracking method in a discrete approach of the evolution of frequency components (both issued from a SMS analysis)

This type of method has been applied to Prony's model in Laroche's work (see [Laroche1] and [Laroche2]). His results indicate that it could be worthwhile pursuing research in this way.

The modeling of the entire transient could be done on successive frames with possible overlap.

Working in this framework and considering that it should give an indication on the non-stationarity decreasing,[1] here are the constraints on the Prony model one should work with:

[1] In order to consider interacting detection and modeling (see section 1.3)

1. the goodness of fit between the model and the signal within a frame
2. the order of the model for a frame
3. the continuity between the different frames of modeling

The trade-off between 1 and 2 would correspond to the Akaike criterion, it would result in an optimization of the model for each frame. The trade-off between 3 and [1 and 2] would be the parameter that would tell whether the signal has reached stationarity or not. Indeed, if one can have a good continuity between the frames that is not achieved to the detriment of the modeling within each frame, let's assume that a relatively stationary area of the signal has been reached.

However, the work by CASTANIE and MOLINARO concerning time-varying Prony models (see [Molin./Castanie]) motivated the consideration of other ways of handling the non-stationarity.

4.2 An evolutionary Prony model

In this paragraph, we intend to introduce the extension of time-invarying methods represented most prominently in the work of GRENIER. We will also introduce the application of this method to the specific Prony model.

In time-invarying methods, one assumes that solely a single realization of the process is available. In the case of stationary signals, one then uses the ergodicity of the process in order to estimate the statistical data (the correlation coefficients) through time averages. In the non-stationary case, this is not possible because the process is not ergodic.[2] Therefore, one has to make other hypotheses on the kind of non-stationarity encountered, or to use other information about the process. The use of a linear decomposition of the time-dependent parameters in a function basis was introduced by GRENIER in [Grenier1] and [Grenier2], and is detailed in subsection 4.2.1. Arguably a stationary vector-AR process can be associated to a non-stationary process.

This issue and the application to the Prony model make up the entire transient modeling scheme illustrated in figure 4.2:[3]

4.2.1 Estimating time-varying parameters of an AR process

Wold's decomposition states that any stochastic stationary process can be obtained as the output of a causal linear filter driven by a white noise. This

[2] See Appendix 5 on page 102 for definitions.
[3] One could compare this illustration to the one for the stationary case on page 35.

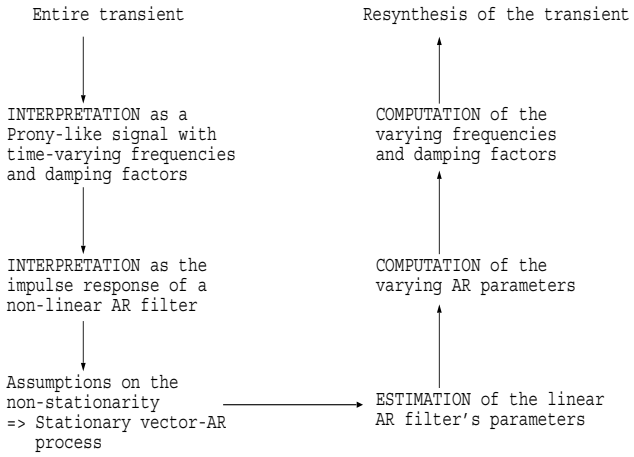


Figure 4.2: Illustration of the Prony algorithm in non-stationary case

is theoretically true whether the output of the filter is thought as a scalar or as a vector. GRENIER showed that a non-stationary process can still be represented by a causal *linear* filter, the key point of his construction is that *by parameterizing the non-stationarity, one turns a non-stationary scalar-AR process into a stationary vector-AR process.* This is done by assuming that the *time-varying coefficients of an AR model can be approximated satisfactorily by a weighted combination of a small number of known functions.* This assumption determines the new hypothesis one makes on the kind of non-stationarity encountered.

4.2.1.1 Theory

In the case of a non-varying parameter, the AR model is:

$x(n) = -\sum_{k=1}^{p} a_k * x(n-k) + \epsilon(n)$

Whereas in the case of a time-varying parameter:

$$x(n) = -\sum_{k=1}^{p} a_k(n-k) * x(n-k) + \epsilon(n) \tag{4.1}$$

$a_k(n)(\forall k = 1, \cdots, p)$ are approximated by weighted combinations of a small number of known functions. The basis of functions $\{f_1(n), \cdots, f_m(n)\}$ implicitly allows specific evolutions of the coefficients.

The time-dependent parameters are expressed as:

$$a_i(n) = \sum_{j=1}^{m} a_{ij} * f_j(n)$$

$$\text{Let } \underline{X}(n) \equiv \begin{vmatrix} f_1(n) * x(n) \\ \vdots \\ f_m(n) * x(n) \end{vmatrix} \text{ and } \underline{A_k} \equiv \begin{vmatrix} a_{k1} \\ \vdots \\ a_{km} \end{vmatrix}$$

Then $a_k(n-k) * x(n-k) = (\underline{X}(n-k))^t \times \underline{A_k}$

Thus equation (4.1) is equivalent to:

$$\epsilon(n) = x(n) + \left[(\underline{X}(n-1))^t, \cdots, (\underline{X}(n-p))^t\right] \times \underline{\theta} \tag{4.2}$$

$$\text{Where } \underline{\theta} \equiv \begin{vmatrix} \underline{A_1} \\ \vdots \\ \underline{A_p} \end{vmatrix}$$

Eventually, the output of our filter (linearized by the introduction of the functions' basis) is now a *vector*.

Let's remind the reader that in the previous stationary case, we were dealing with:

$$\epsilon(n) = x(n) + [x(n-1), \cdots, x(n-p)] \times \begin{vmatrix} a_1 \\ \vdots \\ a_p \end{vmatrix}$$

In equation (4.2), the row representing the output of the filter is m times longer than in the last equation; likewise, the vector representing the parameters one wants to estimate is m times longer too.

Similarly as in paragraph 3.1.1.1.2, one can intend to minimize the modeling error $\epsilon(n)$.[4] Thus the method based on the truncated SVD developed before is still usable. (See section 3.1.2 and Appendix 3 on page 94)

Measurement noise issues As linearity is recovered, one could intend to estimate the parameters of a noise corrupted time-varying AR process using the method described in paragraph 3.1.1.2.2, on page 45. Theoretically, there should be a exact correspondence.

Choice of the basis One can make choices such as Legendre, Fourier or prolate spheroidal basis. It is also possible to incorporate into this framework many other functions, taking advantage of any a priori information one could get, such as the presence of a jump of a coefficient at a known instant.

The issue of orthogonalization of the basis should be taken into account.

The choice of the basis for our specific problem will be discussed in subsection 4.2.2.

[4] An overview of the stochastic and deterministic approaches is given on page 40.

4.2.1.2 Implementations

The algorithm we built first generates an AR process, given the value of the parameters. Then it uses the samples of one realization of the process to estimate back the parameters. This seems a good way to have relevant opinions on the goodness of the results.

Using a MATLAB program, different orders of AR processes have been tested. The performances of the estimations of time-varying as well as time-invarying AR parameters have been measured; a residual between the actual process and the one computed after the parameters' estimation is used to apprehend the performances. Different types of basis have been tried. Eventually, different inputs $\epsilon(n)$ have also been tried (white noise or a simple impulse) so that we could obtain the process itself or the impulse response of the filter.

1. In the next figure one can see a realization of an AR(1) stationary process, then the actual and estimated parameters a_1 and $\widehat{a}_1$ of this process, and then the residual between the process and its reconstruction from the estimated parameter (at the same scale as the process so that the visual comparison be relevant).

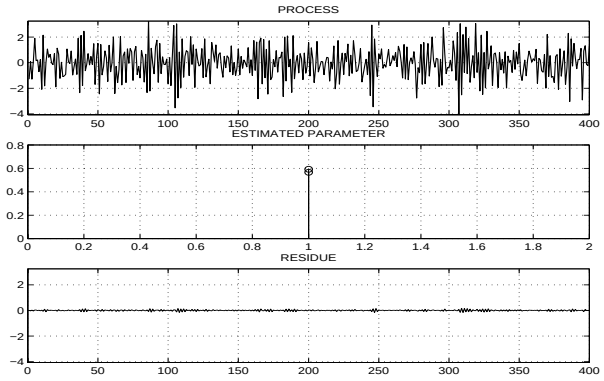


Figure 4.3: Stationary AR(1) process, actual and estimated parameter, residual.

2. In a second example, the basis includes two functions:

   - $f_1(n) = 1$
   - $f_2(n) = n$

   In figures 4.4 and 4.5, one can see realizations of different AR(1) non-stationary processes, then the actual and estimated time-varying parameters $a_1(n)$ and $\widehat{a}_1(n)$ of this processes, and finally the residual (still at the same scale as the process so that the visual comparison be relevant).[5]

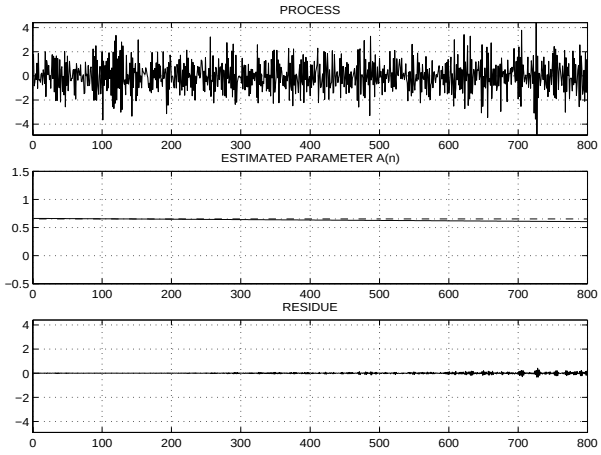


Figure 4.4: Stationary AR(1) process, actual and estimated parameter, residual.

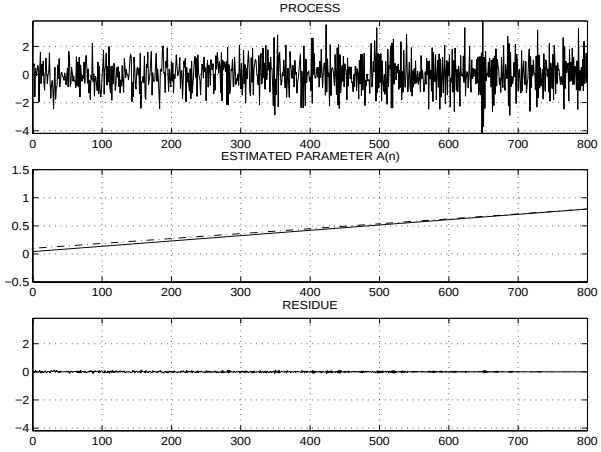


Figure 4.5: Non-stationary AR(1) process, actual and estimated parameter (vary over time with a constant slope), residual.

3. Here, the basis includes three functions:

   - $f_1(n) = 1$
   - $f_2(n) = n$
   - $f_3(n) = n^2$

   In figures 4.6 and 4.7, one can see AR(2) and AR (1) processes which time-varying parameters are defined regarding to these three functions. But the estimated parameters are computed differently in each case. In the former case, the estimator is computed using the three functions of the basis; whereas in the latter, it is computed using only two of the three functions.

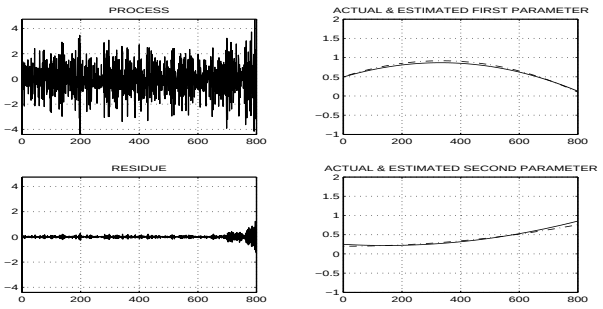


Figure 4.6: Non-stationary AR(2) process, actual and estimated parameters, residual.

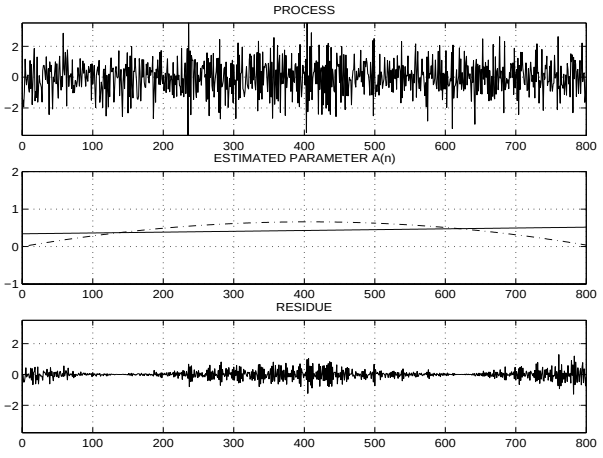


Figure 4.7: Non-stationary AR(1) process, actual and estimated parameter, residual. The search for parameters' variation is done using only $f_1(n)$ and $f_2(n)$.

The example in figure 4.7 shows us that it is important to estimate the variation of the parameter over time using a relevant basis of functions.

4. In this last example, the basis includes these three functions:

   - $f_1(n) = 1$
   - $f_2(n) = \sin(n * \pi N)$
   - $f_3(n) = n^2$

   The second function is set to a sinusoid. The usefulness of it is not obvious, but it is another way to test the efficiency of the estimation algorithm over time-varying parameters.

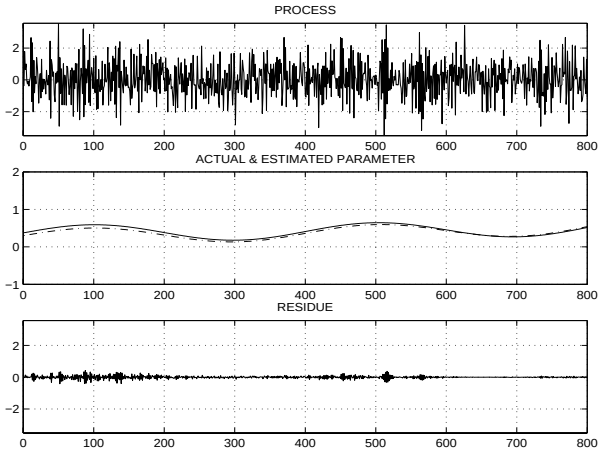


Figure 4.8: Non-stationary AR(1) process, actual and estimated parameter, residual.

Seeing the relatively good results of the estimation scheme over non-stationary AR processes, we tried to estimate the parameters of a Prony model. Let's remind the reader that a Prony-like signal can be interpreted as the impulse response of an AR filter (as introduced in paragraph 3.1.1.1.1 and detailed in Appendix 4 on page 96). However, the tests over Prony parameters' estimation cannot be performed without further theoretical work.

Indeed, one has to estimate the corresponding AR parameters prior to the determination of the frequencies and damping factors; and at this point, it is not clear yet what the actual correspondence between the time-varying parameters of the two models is. (See subsection 4.2.2 concerning this issue)

However, at this stage of the work, we can use the following algorithm to generate time-varying Prony-like signal:

[5] In dotted lines, one can see the time-evolution of the actual parameter (in this case it's constant), and in plain, the estimated parameter.

- run an algorithm that yields the *non-varying* AR parameters corresponding to some known *stationary* Prony-like signal.
- given these parameters, build a slightly different filter made up with parameters that are slightly *changing over time*, according to a given basis (here it is n, n^2, n^3).
- generate a signal as the output of the latter filter which input is an impulse
- one can also add a disruptive white noise at the output of the filter (measurement noise)

Thus we have a way to deal with *non-stationary* Prony-like signals, without actually having to focus on how the frequencies and damping factors evolve, but solely focusing on the estimation of AR parameters.

With very low levels of measurement noise, the algorithm permits to estimate the AR parameters corresponding to a Prony-like signal quite well. Figure 4.9 displays one of these experiments where $SNR = -60dB$.

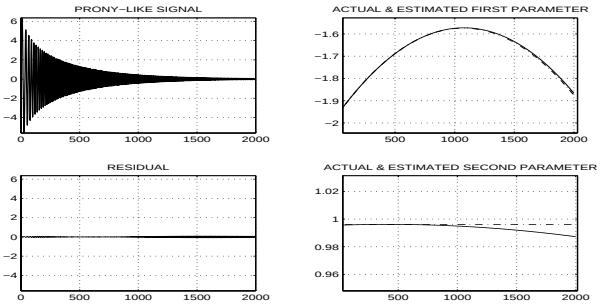


Figure 4.9: Non-stationary Prony signal (1 time-varying damped sine), actual and estimated corresponding AR parameters, residual. $SNR = -60dB$

With higher levels of measurement noise, the results are relatively bad, as shown in figure 4.10. In dotted lines, one can see the time-evolution of the actual parameters, and in plain, the estimated parameters.

As we have already seen several times in this work, the robustness of the estimation scheme to possible measurement noise is a vital issue. We've seen in subsection 3.1.1.2 that several ways to reduce the effect of the noise on the analysis are conceivable. Some more work must be achieved regarding this issue, particularly on the estimation of ARMA models parameters (as explained in paragraph 3.1.1.2.2).

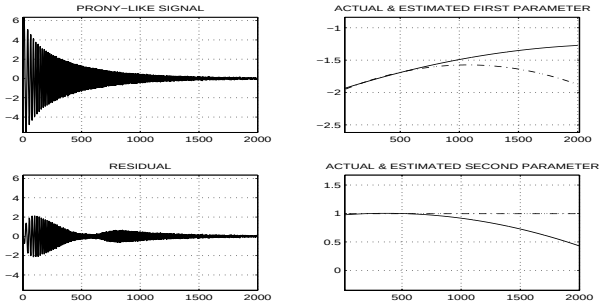


Figure 4.10: Non-stationary Prony signal (1 time-varying damped sine), actual and estimated corresponding AR parameters, residual. $SNR = -40dB$

4.2.2 Estimating the time-varying parameters of a Prony model

As introduced in the previous paragraph, we are assuming that a Prony-like signal can be interpreted as the impulse response of an AR filter; however, the estimation of a Prony model requires additional steps to the estimation of AR parameters. Indeed, one must figure out how poles trajectories (movements of the frequencies and damping factors of the sinusoids) relates to movements of AR parameters so this may be reflected in the decomposition basis.

Here is what we know, at this stage, of the correspondence between the parameters of the Prony model (the frequencies and damping factors) and the parameters of an AR filter:

$$Z_m(n) = \exp(-\alpha_m(n)) * \exp(j2\pi\widetilde{f_m}(n))$$

the $\{Z_m(n)\}$ are the time-varying poles

$$A(Z) = \begin{cases} \sum_{k=1}^{p} a_k(n) * Z^{-k} \\ \prod_{m=1}^{p} (1 - Z_m(n) * Z^{-1}) \end{cases} \tag{4.3}$$

Eventually, the algorithm will use one realization of the process (i.e. the transient signal) to estimate the vector $\underline{\theta}$ (of dimension $m * p$) that generates the non-stationary AR process that fits best the signal. Here p is the order of the model, and m is the length of the basis of functions that the time-varying AR parameters are projected over.

Thus the actual basis that will be used for the computations is the *AR parameters' basis*. However, for the purpose of our modeling scheme, it seems more natural to have relevant basis of functions for the *frequencies* as well as for the *damping factors* in order to characterize the kind of non-stationarities of the *poles*. Indeed, parameterizing the non-stationarity in terms of evolutions of AR parameters obviously lacks the purpose of meaningfulness of the modeling.

We know how to deal with time-varying AR parameters (thanks to GRENIER), but we'd rather deal with time-varying frequencies and damping factors. Hence our goal is to figure out in what way the choice of certain basis decompositions for frequencies and damping factors will affect the poles' trajectories; as well as figure out how poles trajectories relates to movements of AR parameters.

Let's keep in mind that:

1. We want to have relevant basis to decompose the frequencies and the damping factors over.
2. We want these basis to lead to a decomposition of the corresponding AR parameters that would correspond to GRENIER's scheme.

Roughly speaking, we need to devise a practical way to parameterize the variations of the frequencies and damping factors that would *also* allow a simple decomposition of the AR parameters.

Some more theoretical work must be done regarding these issues. In particular, one should answer the following questions:

- Is it possible to find a systematic formula from equation (4.3) that would relate a decomposition of the poles over a certain basis of functions to a decomposition of the AR parameters over another basis?
- Would it really be better to first choose the kind of basis functions for frequencies and damping factors, and then determine the corresponding basis it yields in terms of AR parameters? Or would it be better to start from a decomposition of the AR parameters (that we want to choose so that the GRENIER's scheme would be easily computable), and then see what it yields to in terms of poles, and eventually in terms of frequencies and damping factors?
- If the former philosophy is eventually adopted, then what kind of assumptions should be made on the way the frequencies and the damping factors are supposed to evolve? Let's say we decompose them as sums of weighted functions. What kind of functions are relevant for our purpose?

For this purpose, the reference [Molin./Castanie] may be insightful.

4.3 Prony multipulse

The processes that were designed to model the signal in stationary cases can be "unstationarized" for the current purpose using a different scheme. Staying in the "input→system→output" framework, instead of using time-varying filter, one could think about using non-stationary input to a linear filter.

NB: No implementations nor theoretical work has been done regarding this issue, thus no conclusions will be made. This paragraph only stands as a very elementary presentation of an seemingly interesting related research topic.

Multipulse modeling has been initiated in the speech coding domain by ATAL (see [Atal]). It consists in determining the series of impulses which, being put in the input of a linear invarying filter (of impulse response $h(n)$), would best describe the signal $x(n)$ by:

$$x(n) = \sum_{i=1}^{N_{imp}} A_i * h(n - n_i) + b(n)$$

Where $b(n)$ is the modeling error.

It stands basically as the same estimation problem as before, the difference is that one must first estimate the instants of impulses and also their amplitudes.

Associating the Prony model to this multi-impulses philosophy yields to the following model:

$$x(n) = \sum_{i=1}^{N_{imp}} A_i * \left\{ \sum_{m=1}^{2L} B_m * Z_m^{(n-n_i)} \right\} * u(n - n_i) + b(n)$$

This type of modeling is developed in YVETOT's work (see his PhD thesis [Yvetot]). He also details the use of *a different Prony model for each impulse*. This type of model seems to be interesting for musical concerns. Particularly in the case of polyphonic signals where the data come usually *almost* simultaneously from very different sources. Indeed, it makes sense to analyze as *one non-stationarity* a precise problem region of the signal[6] resulting from the addition of different transient events (issued from separate instruments, differing regarding their spectral characteristics).

4.4 Conclusions regarding the non-stationary case

4.4.1 Comparison of the methods

Discrete methods implicitely require a compromise between the accuracy that can be achieved with a short data segment and the faithfulness with which one can ensure the continuity between frames.

[6] Such problem regions can be seen in figure 1.6.

Using an evolutionary method should be better for several reasons:

1. The analysis would be done on the entire transient region, what should have good consequences regarding the robustness to measurement noise. Indeed, as shown in paragraph 3.1.4.2, the more N is important, the more the estimation algorithm is noise-resistant. It is also shown that it improves the frequency precision.

2. It is more meaningful to apprehend continuous evolutions to the frequential components than discrete ones.

4.4.2 Connection with the detection scheme

Let's remind the reader that the quality of the modeling of non-stationarities is not the only goal one wants to achieve. Indeed, as introduced from the very beginning, it is essential to think about the link between the modeling scheme and the detection scheme. As stated in the introduction, we eventually want to consider interacting schemes for detection and modeling of nonstationarities.[7]

The use of a discrete method facilitates this issue. Indeed, as detailed in section 4.1, devising a criterion for the evolution of the non-stationarity is easily conceivable in this case.

While an evolutionary method should yield better results than a discrete one concerning the modeling, it seems less easy to connect it to the detection scheme. Indeed, unlike in the discrete method, one must know the length of the entire transient in order to set a size to the analysis window. This seems to be in antinomy with what has been introduced in section 1.3: "The model of transient must give an indication of the non-stationarity decreasing. Depending on this indication, the detection scheme starts again".

A way to reconcile the issues of good modeling and relevant connection with the detection could be the following:

- Settle a high number of samples to achieve the modeling on, using an evolutionary method. This region must be chosen so that it is assumed to contain more data that solely the transient.

- Compute a modeling of the same region using a sines + noise model (like SMS)

- Comparing these two modelings. One could use an algorithm based on a measure of divergence between these two models to achieve this comparison (see section 6.2 for details regarding the comparison between two models).

[7] The philosophy of unifying the characterization and the segmentation issues has been justified in chapter 1.

- Re-settle the size of the analysis window (i.e. reduce it) according to a decision made over what rate of error in the modeling is considered acceptable. This step would be directly linkable to compression concerns.

NB: This method is inspired by Levine's transient detector philosophy (see page 23). Indeed, the information contained in the difference between two modelings of the signal (one which takes into account the modeling of transients and the other which assumes a stationary signal) should be used.

Part II

Applications and conclusions

Chapter 5

Applications

This chapter proposes some applications that might follow the completion of the project we developed in this document. Provided that a unification of the characterization and segmentation issues in musical signals can be achieved; possessing a powerful and flexible tool to detect and represent nonstationarities might allow the user to deal with monophonic, as well as polyphonic signals.

What is called "the algorithm" in the following is obviously an algorithm "to be"; future work that has to be done is described in the next chapter.

Sines+noise+transient: First of all, in a sines + noise + transient framework, this algorithm is meant to be linked to existing analysis/synthesis methods that currently cannot handle certain difficult regions of musical signals. Its very nature might permit a more flexible and meaningful parameterization of signals. The use of the Prony model allows for an *additive synthesis of the transients*, which is another advantage for the connection to existing sines + noise existing methods.

Timbres classification: In a monophonic context, a very interesting application could stand in the classification of timbres. Let's assume that an analysis (very accurate in time as well as in frequency)[1]of a transient in terms of a sum of sinusoids is proven relevant. Then, the way the junction is made between the transient region and the stationary region may be insightful with respect to the very nature of the source. Indeed, the user possesses a measure of the *coherence* between the two regions. It might be worthwhile thinking that different timbre classes can be separated with a certain relevance according to such a criterion.

Audio compression: As it has been shown in [Verma1] and [Levine], the modeling of signal is very much related to compression goals. This algorithm, based on a criterion measuring the degree of non-stationarity, would permit the user to monitor a tuning parameter: one could choose

the trade-off corresponding to a particular application between the goodness of the modeling and the compression rate. The extreme boundaries are: "very good compression rate / modeling as sines + noise solely", and "small compression rate / a lot of regions are modeled as transients".

Scale modifications: This algorithm would naturally find use in *time and pitch-scale modifications without artifacts.*[2] Possessing a good model for attack-transients would consist in an important improvement of these current modifications. Indeed, in order to produce a perceptually-natural scaled signal, it is well-known that one must conserve intact the duration and spectral characteristics of the attack-transients regions.

Synthesis: Being able to parameterize transients in a *meaningful* way is certainly useful for a synthesis goal. Indeed, new textures of sound can be derived from the alteration of the model's parameters.

Sound content description: The low level representation this algorithm would yield might serve as a first step for a sound content description of the signals at a higher, cognitive level. Particularly for polyphonic sounds. Indexation of audio and synchronization of audio and video are also conceivable.

Rhythm detection: Likewise, a rhythm detection scheme could be based on the accurate detection of the attack-transients provided by the algorithm.

Hybridization of several sounds could be considered as a possible extension of the classification of timbre introduced above.

Music performance: Eventually, the definition and automatic extraction of *performance parameters* from a particular musical performance could be considered.

[1] Which should be allowed by the use of the Prony model.

[2] Time-scale modifications alter the playback speed of audio without changing the pitch. Similarly, pitch-scale modifications alter the pitch of the audio without changing the playback speed.

Chapter 6

Conclusions and future work

6.1 Conclusions

In this document, musical signals are considered to be series of stationary regions between which are located transient regions. Our topics of interest are: the detection of transients and the modeling of the signal in the transient portions. An introductory chapter gives a review of onset detection in the context of analysis/synthesis of digital musical signals and justifies the approach of considering jointly the segmentation (detection) and the characterization (modeling) issues in the sequential processing of a signal.

The latter topic has been developed in the first part of this report. The use of Prony's model is justified with respect to theoretical issues; it is also justified in the work that has been done on the stationary case, as well as on the non-stationary case. Implementation work is highlighted and discussed. The segmentation issue has not been treated in this document. However, the modeling issue is treated with a focus on the essential feature that should be accounted for: some measure of the decrease in the non-stationarity of the transient regions.

The necessity of handling the measurement noise problem has been emphasized on several occasions. We also introduced a particular scheme with respect to a possible reduction of the measurement noise effect on the estimation of the Prony model's parameters.

In conclusion, some additional work must be done in order to realize the "Chicken and egg" project and to complete an algorithm that would serve as an improvement of current analysis/synthesis methods. The next section gives an overview of the directions in which subsequent work should be pursued.

6.2 Future work

- Improve the reduction of the measurement noise effect on the estimation of the Prony model's parameters. That is try the different methods developed in [Kay] with respect to ARMA models parameters estimation; and improve their application to the Prony model. The importance of this issue has been shown in subsection 3.1.1.2, 3.1.4.4, 3.2.2.3 and also 4.2.1.2.

- Handle the amplitude and phase estimations of the Prony model in the non-stationary case. [Laroche1], [Laroche2] and [Molin./Castanie] are the references related to this issue.

  Even in a theoretical non-noisy framework, studying ARMA parameters' estimation should be worthwhile. Indeed, as it is explained in Appendix 4 (on page 96): a Prony-like signal can be interpreted as the impulse response of an ARMA filter. The asumption that it can be interpreted as the impulse response of an AR filter is shown to be sufficient for the computation of the frequencies and damping factors; however, the zeros (i.e. the MA part) could account for phases and amplitudes concerns.

- Develop a better estimation scheme of the order L. This issue has been shown to be very important in subsection 3.1.4.1. Further research regarding the Akaike criterion could find use.

- Pursuing research relating to the different methods considered for the handling of the non-stationary Prony model. Particularly the Prony multipulse model that should find use in the handling of polyphonic signals (see section 4.3).

- In the framework of the evolutionary Prony model, we need to devise a practical way to parameterize the variations of the frequencies and damping factors that would *also* allow a simple decomposition of the AR parameters. This issue has been introduced in subsection 4.2.2 where precise questions are emphasized. The work developed in [Molin./Castanie] might find use for this concern.

- Combine the work regarding the modeling issue to an essential research concerning the detection, so that the modeling and detection scheme consist in a single program. The activation of the modeling scheme might eventually be monitored by an indication regarding the presence of a non-stationary region. Here is a very basic way to introduce the theoretical framework regarding the detection issue:

  Abrupt change detection Musical signals are considered to be series of stationary regions between which are located transient regions. The spectral characteristics are assumed to change abruptly between stationary segments and transients that are highly non-stationary by definition.

The goal is to determine where and if a change occur in the nature of the signal. For this purpose, the data is processed sequentially, the algorithm accesses only present and past data. The basic philosophy is to use a modeling of the signal; the meaningful changes detection is given by the measure of a statistical distance between two models of the signal. The following general steps must be followed:

- Adopt a choice of model
- Define a statistical test between change and no change
- Make a trade-off between temporal precision and false-alarm rate (thresholds tuning)

The main hypothesis is that stationary parts are well described by an AR model. As in BASSEVILLE's method for detecting abrupt changes (see [Bass./Nik.]), we want to compare two AR models: the first is computed within a frame that *grows* with the number of samples, the second keeps the same size but is *sliding* over the signal.

Basically, the detection scheme should be based on sequential tests, *for every sample* a likelihood ratio test might be performed to decide between hypotheses: "no change" or "change at this point". This gives the answer to whether a change occurs. But the instant of detection and the actual instant of change can differ, due to the thresholds tuning which determines the decision. ANDRE-OBRECHT's work (see [Andre-Obrecht] and [Lep./Obrecht]) shows that it is wortwhile considering a *backward* algorithm for improving the accuracy of the abrupt change location.

The following references may serve as a basis to pursue this work:

[Andre-Obrecht], [Lep./Obrecht], [Bass./Nik.] and finally ongoing work by THORNBURG.

- Regarding the modeling and detection issues, one could extend the methods to subbands in an attempt to account for the perception of signals. It has also been shown to serve as an improvement of the Prony modeling accuracy (see [Laroche1]).

- As it has been introduced in subsection 3.1.1.3, it may be worthwhile to consider the state-space model theory. The input of the modeling filter would account for either a stochastic part (state noise) and a deterministic part (series of impulses). This would permit a handling of the approximation made in this work regarding the correspondence between the Prony model and the AR model (see Appendix 4 on page 96). Accordingly, a separate handling of both noises (the measurement noise added at the output of the filter and the state noise) would be possible.[1]

- Eventually, a deeper interest regarding the lattice filters framework would be worthwhile. It is known that this framework accounts for estimations of

a model parameters that are associated with indications on the goodness of the modeling, which may serve as a good way to obtain better order estimations.

[1] Indeed, it seems clear that the measurement noise (error of the model and actual noise added to the signal -bad recording, ...), and the state noise (stochastic noisy nature of the sound -breathing in flute sounds, ...) should be treated separetly.

Part III

Appendices

Glossary of symbols and notational conventions

- $\underline{a} = (a_0, \cdots, a_p)$: general notation for a vector
- $\widehat{h}$: estimation of h
- Capital letters ($X,\ W, \cdots$) : stochastic processes
- Capital letters ($X(f),\ X(Z), \cdots$): transforms of a signal (Fourier transform, Z-transform,$\cdots$)
- Small letters ($x(n),\ w(n), \cdots$): realizations of stochastic processes; signals
- $P_{XX}(Z)$: Power Spectral Density (PSD) of the process X
- $\delta(n) = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n \neq 0 \end{cases}$: impulse
- $\star$: convolution operator
- $\times$: matrix product
- $*$: product
- N: number of samples
- p: order of a model
- L: number (estimated) of sinusoidal components in a signal
- F_s is the symbol for the sampling frequency which has been chosen equal to $44.1kHz$ in this work.
- SNR: stands as Signal over Noise Ratio

Appendix1: Yule-Walker

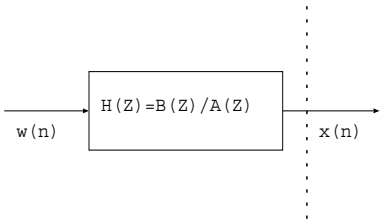


Figure 6.1: Yule-Walker: ARMA model

Say $X(n)$ is an ARMA(p,q) process, $x(n)$ a realization of X, and $w(n)$ a realization of the stationary white noise $W(n)$ of variance σ^2.

$h(n)$ is the impulse response of the filter (assumed causal).

$H(Z)$ is the Z-transform of $h(n)$. $H(Z) = \frac{b_0 + b_1 * Z^{-1} + \cdots + b_q * Z^{-q}}{1 + a_1 * Z^{-1} + \cdots + a_p * Z^{-p}}$.

$$X(n) = -\sum_{k=1}^{p} a_k * X(n-k) + \sum_{k=0}^{q} b_k * W(n-k)$$

$R_X(m) \equiv [X(n) * X(n-m)]$

$\Longrightarrow R_X(m) = E[(-\sum_{k=1}^{p} a_k * X(n-k) * X(n-m)) + (\sum_{k=0}^{q} b_k * W(n-k) * X(n-m))]$

$\Longrightarrow R_X(m) = -\sum_{k=1}^{p} a_k * E[X(n-k) * X(n-m)] + \sum_{k=0}^{q} b_k * E[W(n-k) * X(n-m)]$

$$\Longrightarrow R_X(m) = -\sum_{k=1}^{p} a_k * R_X(m-k) + \sum_{k=0}^{q} b_k * R_{WX}(m-k) \qquad (6.1)$$

$X(n) = h(n) \star W(n)$ where $\star$ is the convolution operator

$\Longrightarrow X(n-m) = \sum_{l=0}^{\infty} h(l) * W(n-m-l)$ (l starts at index 0 for causality)

$$E[W(n-k) * W(n-m-l)] \equiv \begin{cases} \sigma^2 \text{ if } l = k-m \\ 0 \text{ if } l \neq k-m \end{cases}$$

$$\begin{aligned} \Longrightarrow E[W(n-k) * X(n-m)] &= \sum_{l=0}^{\infty} h(l) * E[W(n-k) * W(n-m-l)] \\ &= h(k-m) * \sigma^2 \end{aligned}$$

Thus:

$$R_X(m) = -\sum_{k=1}^{p} a_k * R_X(m-k) + \sigma^2 * \sum_{k=m}^{q} b_k * h(k-m) \text{ for } m = 0 \cdots q \quad (6.2)$$

$$R_X(m) = -\sum_{k=1}^{p} a_k * R_X(m-k) \text{ for } m \geq q+1 \tag{6.3}$$

(NB: k starts at index m in the second sum of equation (6.2) for causality)

It should be noted that the relationship between the parameters of an ARMA process and the autocorrelation function is a nonlinear one. Indeed, in the term $\sum_{k=m}^{q} b_k * h(k-m)$, the impulse response $h(n)$ of the filter contain the parameters a_k and b_k.

Appendix 2: ARMA parameters estimation methods

Several ARMA parameters estimation methods are detailed in [Kay], the algorithms we did test are the Two-stage and Three-stage (also called Mayne-Firoozan) Least Squares (LS) methods. They pertain to a class of ARMA estimation algorithms that rely on the estimation of the driving white noise.

In the ARMA framework, the non-linear nature of the Yule-Walker equations is due to the unknown cross-correlation between the input and the output (see appendix on Yule-Walker page 90). Since the input is unobservable, the cross-correlation cannot be estimated. If however we knew the input, the ARMA parameters could be estimated as a solution of a set of linear equations.

The final goal is to estimate the parameters of an ARMA(p,q) model. This is used in paragraph 3.1.1.2.2 as a way to estimate the true AR parameters of a noise corrupted AR process.

The signal is modeled as shown in figure 6.2:

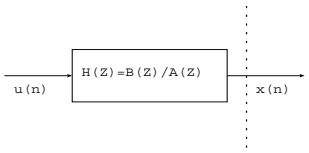


Figure 6.2: ARMA parameters estimation: ARMA filter

The Three-stage LS estimator algorithm is the following:

1. Fit a large order AR model to $x(n)$. Let $\widehat{a}_k^L$ be the estimated parameters.

2. Find the estimate of $u(n)$ as the output of the whitening filter which input is the signal (NB: $x(n) = 0\ \forall n < 0$).

$$\widehat{u}(n) = \sum_{k=0}^{L} \widehat{a}_k^L * x(n-k)\ \forall n = -q \cdots N-1$$

3. The true ARMA parameters verify:

   $x(n) = -\sum_{k=1}^{p} a_k * x(n-k) + \sum_{k=0}^{q} b_k * \widehat{u}(n-k)$

   Thus one can use $\widehat{u}(n)$ to generate estimates of the AR and MA parameters, this can be done in solving the following Least Squares problem: One wants to find the vector $\underline{\theta}$ that minimizes $\|H\underline{\theta} + \underline{h}\|^2$.

Where $\underline{\theta} = [-a_1, \cdots -a_p, b_0, b_1, \cdots b_q]^t$, $\underline{h} = [x(N) \cdots x(1)]^t$ and

$$H = \begin{vmatrix} x(N-1) & & \cdots & x(N-p) & \widehat{u}(N) & \widehat{u}(N-1) & & \cdots & \widehat{u}(N-q) \\ & & & \vdots & & & & & \vdots \\ x(p) & & \cdots & x(0) & \widehat{u}(q+1) & & & \cdots & \widehat{u}(0) \\ & & & 0 & \widehat{u}(q) & & \cdots & \widehat{u}(0) & 0 \\ & x(0) & & \vdots & & & \widehat{u}(0) & & \vdots \\ x(0) & 0 & \cdots & 0 & \widehat{u}(1) & \widehat{u}(0) & 0 & \cdots & 0 \end{vmatrix}$$

$\widehat{\underline{\theta}}$ is the estimate of $\underline{\theta}$, let $\widetilde{b}_k$ be the estimates of the MA parameters.

4. Form the following sequences:

   - $y(n) = -\sum_{k=1}^{q} \widetilde{b}_k * y(n-k) + x(n)$
   - $z(n) = -\sum_{k=1}^{q} \widetilde{b}_k * z(n-k) + \widehat{u}(n)$

5. Determine the final estimates of the ARMA parameters by solving the same Least Squares problem as in step 3, where $y(n)$ takes the place of $x(n)$ and $z(n)$ the one of $\widehat{u}(n)$ in the construction of the matrix H.

Steps 1 to 3 correspond to the Two-stage LS estimator (which is known to be inefficient), while steps 4 and 5 attempt to generate an efficient estimator.

NB: Results and discussion regarding this method are given on page 45.

Appendix 3: Truncated SVD

It is known that a *Singular Value Decomposition* (SVD) of a matrix A can highlight its null-space and singular values, thus it is a way to verify whether it is a singular matrix. It also yields eigenvectors of A^tA and AA^t.

As introduced in [Kum./Tufts], the particular *truncated SVD algorithm* is a robust solution to *Least Squares (LS) problems*, that is very well adapted to our particular problem where we possess an estimate of the order of the model and thus of what should be the size of the null-space of the covariance matrix.

Say one wants a solution to a LS problem (i.e. find a solution $\underline{c}$ to the minimization of $\|A \times \underline{c} + \underline{h}\|^2$). It is known that the pseudoinverse $(A^tA)^{-1}A^t$ of the matrix A gives a vector $\underline{c} = -(A^tA)^{-1}A^t \times \underline{h}$ that is the unique solution to this minimization. However, A must be non-singular in order to be able to compute its pseudoinverse. When M is singular, the pseudoinverse can't be computed, there is no unique solution but an infinite family of solutions. One of them is the minimum norm vector that lies in the null-space of the matrix A; it is known that it can be computed as a linear combination of the eigenvectors of A^tA, "weighted" by the eigenvalues. An SVD can be used for highlighting the eigenvalues and singular values.

In our particular case, $x(n)$ is a Prony-like signal (sum of damped sines) and M and $\underline{a}$ are defined as on page 38.

In the case of noiseless data, we've seen that M is singular and $\underline{a}$ lies in the null-space of M and M^tM,[2] thus

$$M\underline{a} = 0$$

We can define M' such as $M = (\underline{x} \mid M')$ where $\underline{x} = (x(N) \cdots x(p))^t$, and $\underline{a'}$ such as $\underline{a} = (1, \underline{a'}^t)^t$, thus $M'\underline{a'} = -\underline{x}$.

NB: $dim(M') = (p-1) \times (N-p+1)$ and $dim(\underline{a'}) = (p-1) \times 1$.

In the case where $x(n)$ doesn't fit exactly to a Prony-like signal, one may want to minimize the least squares error, thus the goal is:

$$\text{find } \underline{a'} \text{ that minimizes } \|M'\underline{a'} + \underline{x}\|^2 \tag{6.4}$$

One could compute the pseudoinverse of M', indeed, the effect of the noise (i.e. the modeling error) is that M'^tM' is non-singular, thus $(M'^tM')^{-1}$ does exist.[3]

However, the fact that M'^tM' is non-singular is due to a disruptive term (the modeling error), we'd rather have a non-noisy (thus singular) matrix, and find

[2] M and M^tM have the same null-space .

[3] In the case where M' is not square, and M' and M'^tM' are singular, M'^{-1} and $(M'^tM')^{-1}$ don't exist.

the minimum norm vector $\underline{a'}$ pertaining to the infinite family of solutions that fits the model. That is what the truncated SVD method does, by *forcing the matrix to be singular and computing the minimum-norm vector that lies in its "forced" null-space.*

The computation of the Singular Value Decomposition yields:

$$[U, S, V] = SVD(M') \Longrightarrow M' = U \times S \times V^t$$

Where V is the matrix of the eigenvectors of $M'^t M'$ and U is the matrix of the eigenvectors of $M' M'^t$.

$S = \begin{vmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \\ 0 & & \sigma_p \end{vmatrix}$ is the diagonal matrix of the singular values of M'. (NB: in the noiseless case of a $2L$-Prony signal, the diagonal terms $\sigma_{2L+1}, \cdots, \sigma_p$ are equal to 0)

$dim(U) = (N - p + 1) \times (N - p + 1)$, $dim(S) = (N - p + 1) \times (p - 1)$ and $dim(V) = (p - 1) \times (p - 1)$.

$M'\underline{a'} = \underline{x} \Leftrightarrow USV^t \underline{a'} = \underline{x}$

$$\Rightarrow \underline{a'} = -\sum_{k=1}^{2L} \sigma_k^{-1} * \left[\underline{u_k^t} * \underline{x}\right] * \underline{v_k} \tag{6.5}$$

Where $\underline{v_k}$ are the eigenvectors of $M'^t M'$ and $\underline{u_k}$ are the eigenvectors of $M' M'^t$.

In equation (6.5), we have made use of only the $2L$ principal eigenvectors of $M'^t M'$ and $M' M'^t$ which are more robust to the noise perturbations in the data, *this is where the truncation takes place.* The least squares solution to equation (6.4) can be written in the same way as equation (6.5), but it would include all the singular values of M' and the corresponding eigenvectors of $M'^t M'$ and $M' M'^t$. Thus the stability in the coefficient vector $\underline{a'}$ is achieved by dropping from the least square solution to equation (6.4) the less robust eigenvectors of $M'^t M'$ and $M' M'^t$.

The effect of using a truncated SVD algorithm is to increase the SNR in the data prior to obtain the solution vector $\underline{a'}$.

In the case of noiseless data, since the rank of M' is $2L$, only $\sigma_1 \cdots \sigma_{2L}$ will be nonzero and $\underline{a'}$ will be the minimum norm solution desired.

Appendix 4: the Prony/AR correspondence

One may want to test the pertinence of the basic asumption we are making in paragraph 3.1.1.1.1: "*a Prony-like signal can be interpreted as the impulse response of an AR filter*".

As we are not testing the robustness of the model, but the relevance of the AR/Prony correspondence, there is no noise (neither state noise nor measurement noise) at this stage. Besides, as we can run experiments with a synthetic signal, the error of model is equal to zero.

Here is a way to have a better comprehension of the goodness of our basic asumption. One can:

1. generate a Prony-like signal (L sinusoids), compute its corresponding AR parameters (there are $2L$) as in subsection 3.1.2, and build the AR filter
2. put in input of this filter an impulse. Thus, one can compare its output with the signal.
3. build the corresponding whitening filter (i.e. inverse), and put the signal in input. Thus one can compare the output to what it should be (i.e. an impulse)
4. a way to check if this computation of the AR parameters works (without testing the pertinence of their meanings) is to put the output of the whitening filter in input of the AR filter, the final output must be the signal.

Here are the parameters one wants to focus on:

- L = number of sinusoids
- p = order of the AR filter, p must be $> 2L$ (see paragraph 3.1.4.2)

Impulse response of the AR filter

The first logical way to apprehend the goodness of the asumption "A Prony-like signal can be interpreted as the impulse response of an AR filter" can be to build the all-pole filter from the knowledge of $A(Z)$ (which is given by our SVD-based method explained in paragraph 3.1.2), put an impulse in input, look at the output, and compare it to the signal itself, this corresponds to the steps 1 and 2 enumerated above.

One can synthesize a signal made by an exponentially damped sinusoid[4], as shown in figure 6.3:

[4] $L = 1$, $f = 500$ Hz, $\alpha = 0.008$

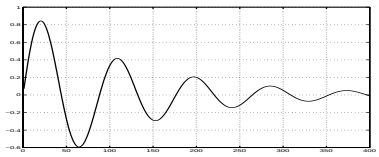


Figure 6.3: Prony/AR correspondence: One damped sinusoid

From this signal, we can get the parameters a_k of this filter.

If we use these parameters a_k in a program that builds a filter[5] and yields its ouput given an impulse input, it yields a process shown in figure 6.4:

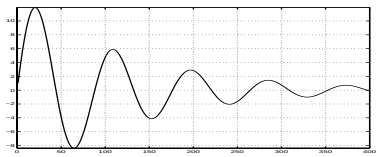


Figure 6.4: Prony/AR correspondence: Impulse response of the all-pole filter

In figures 6.3, 6.4 and 6.5, one can compare the original synthetic signal and the impulse responses of two different AR filters.

With L fixed, how can p vary?

L must be greater or equal to the actual number of sinusoids (see paragraph 3.1.4.1).

Here, with a synthesized signal, we can consider that we know exactly this number of sines L. In the truncated SVD algorithm, the size of the columns of the matrix V determines the number p (see subsection 3.1.2), hence the length of the vector $\underline{a}$, thus it determines how much a_k we have that make up the all-pole filter.

Figure 6.4 showed a first example where L is the exact number of sinusoids, and $p = 2L + 1$

Figure 6.5 is an example where $p > 2L + 1$:[6]

[5] $[a_0 = 1, a_1 = -1.9790, a_2 = 0.9841]$. Number of sinusoids = 1, $L = 1$, $p = 3$.

[6] $[a_0 = 1, a_1 = -0.7764, a_2 = -0.4929, a_3 = -0.2022, a_4 = 0.0943, a_5 = 0,3950]$. Number of sinusoids in the synthetic signal = 1, $L = 1$, $p = 6$

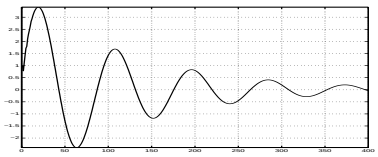


Figure 6.5: Prony/AR correspondence: Impulse response of the all-pole filter

With L moving (but greater than the actual number of sines), how can p vary?

Let's asume that we don't know the exact number of sinusoids, and that we use an estimation scheme to estimate it. We don't want to underestimate it,[7] so we choose L a little bit greater than the result of the estimation. Figure 6.6 is an example where L is overestimated.[8]

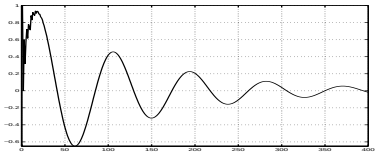


Figure 6.6: Prony/AR correspondence: Impulse response of the all-pole filter built with 11 parameters a_k.

In conclusion of this paragraph, it is shown that provided that it is always greater than two times the actual number of sinusoids, different values of p (i.e. different orders for the filter) can yield very similar results in term of impulse response. At this stage one could think that the asumption is justified. But let's deal with more complex signals in the next paragraph.

Whitening filter

Another way to measure the goodness of our basic asumption could be to put the signal itself in input to the all-zeros whitening filter corresponding to the previous AR filter, and comparing its output to what it should be: an impulse (this corresponds to step 3.).

Figure 6.7 shows the original signal (sum of sinusoids), the impulse response of the computed AR filter, the output of the inverse filter (i.e. the whitening filter) when its input is the signal, and eventually the signal issued from the AR filter which input is the output of the whitening filter (obviously it must yield the original signal), this is step 4. This is useful in order to check the goodness

[7] As discussed in 3.1.4.1.

[8] Number of sinusoids in the synthetic signal = 1, $L = 5$, $p = 12$

of the AR parameters' computation without taking into account the pertinence of their meanings.

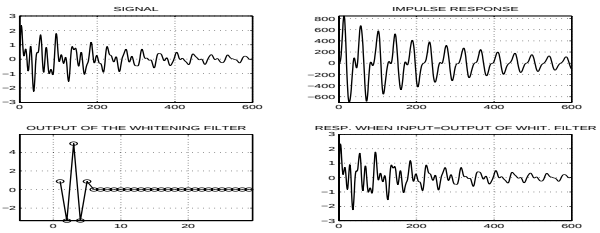


Figure 6.7: Prony/AR correspondence: whitening filter

We can see in figure 6.7 that:[9]

- The output of the AR filter is similar to the signal when its input is the the ouput of the whitening filter. Thus the computation of the AR parameters is valid.
- The impulse response looks different than the original signal.
- The output of the whitening filter is more complex than an impulse (what we thought it would be), and it is equal to zero only after the index p.

Conclusions

We've seen that the computation of the AR parameters is valid, however, there is a difference between the impulse response of the AR filter and the Prony signal. To understand why this is not a big problem, let's look a little further in our main algorithm. The algorithm we built to estimate the AR parameters achieves this task as an *intermediate stage*, in order to compute the corresponding poles and then the frequencies and damping factors of the corresponding Prony model. Given those, it is possible to resynthesize the original signal. In figure 6.8, we show the original signal,[10] the output of the whitening filter (which input is the signal), the impulse response of the AR filter, and the signal generated with the computed frequencies and damping factors.

- Beside the fact that the damping factors are underestimated a little bit, the original signal and its resynthesis are quite similar. It is much better than the impulse response of the filter.

[9] $L = 3$ (3 sinusoids), $p = 7$ (AR(7))
[10] $L = 3$, $p = 30$ (AR(30))

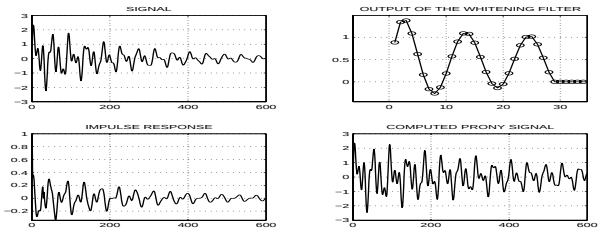
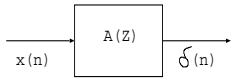


Figure 6.8: Prony/AR correspondence: whitening filter

The asumption we are making corresponds to:

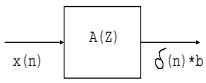
$$\delta(n) = x(n) + \sum_{k=1}^{p} a_k * x(n-k)^{11} \Leftrightarrow$$



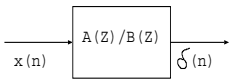
What we actually get is:

$$\sum_{q=0}^{p} b_q * \delta(n-q) = x(n) + \sum_{k=1}^{p} a_k * x(n-k)$$

$\Leftrightarrow$



$\Leftrightarrow$



In these two different cases, *only after the order p*, one can see that we eventually respect the equation (3.2) (page 38).

$$x(n) = -\Sigma_{k=1}^{p} a_k * x(n-k) \ (\forall n \geq p+1)$$

But before the order p, the behaviors of these two models (the one we're asuming we have and the one we actually have) differ.

Let's remind the reader that the algorithm based on the Prony/AR asumption achieves the AR parameters' estimation as an *intermediate stage*, in order to compute the corresponding poles. The output of the whitening filter shows us the *zeros* part. Therefore, our basic asumption is not quite exact, we are actually simplifying the problem, the correct asumption to make would be: "*a Prony-like signal can be interpreted as the impulse response of an ARMA filter*" (which still corresponds to the asumption made by Grenier in [Grenier2]: "$x(n)$ can be considered as the output of a linear system with input zero stochastic initial conditions").

But at this stage of the algorithm, we only want to deal with the *poles of the modeling filter*, only them (and not the zeros) account for the computation of the frequencies and the damping factors. The zeros (i.e. the MA part) would account for the phases and amplitudes computations.

---

[11] NB: $\delta(n) = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n \neq 0 \end{cases}$

Thus the asumption we are making is sufficient for our purpose at this stage of our method. However, one could still want to store in memory the p coefficients yielded by the whitening filter, and use them for phase and amplitudes concerns (see section 6.2).

Appendix 5: Stationarity and ergodicity of a stochastic process

If n is the variable representing time, let $Y(n)$ be a *stochastic process*. $y(n)$ is a realization of $Y(n)$.

For a precise value n_i of n, $Y(n_i)$ is a *random variable*, and $y(n_i)$ a realization of this variable.

$Y(n) \equiv (Y(n_1), \cdots, Y(n_N))^t$ and $y(n) \equiv (y(n_1), \cdots, y(n_N))^t$

To represent the the statistical behavior of a random variable and processes, one may want to use the following descriptors:

- Probability laws that describe the simultaneous belongingness of several random variables to a domain:
  - Probability Density Function (PDF)[12]:

    $$P\{Y(n_1) \in [y(n_1), y(n_1) + \Delta(y(n_1))[; \cdots; Y(n_N) \in [y(n_N), y(n_N) + \Delta y(n_N)[\}$$

  - Repartition Function (RF): $P\{Y(n_1) \in]-\infty, y(n_1)[; \cdots; Y(n_N) \in]-\infty, y(n_N)[\}$

    $$\Leftrightarrow P\{Y(n_1) < y(n_1); \cdots; Y(n_N) < y(n_N)\}$$

- Momentum:
  - Expectation: $E[X] = \int x * P\{X < x\}\, dx$ (momentum of order 1)
  - Variance: $Var[X] = E\left[(X - E[X])^2\right] = \int (x - E[X])^2 * P\{X < x\}\, dx$ (momentum of order 2)

Stationarity

In the case where **all** probability functions are independant of a shift of time origin, the stochastic process is said to be *stationary in the wide sense*.

In the case where $E[Y(n_i)]$ can be written as a quantity that doesn't depend on time, and $E[Y(n_1)] = E[Y(n_2)] = \cdots = E[Y(n_N)]\ \forall i$, then one usually write

$$E[Y(n_i)] = m_Y$$

[12] Δy represents the infinitesimal variation of y.
An example of a PDF (for a single random variable for a better readability) is the Gaussian distribution:

$$P[Y \in]y + \Delta y]] = \frac{1}{\sigma^2\sqrt{2\pi}} \exp\left(-0.5 * \frac{(y - E[Y])^2}{\sigma^2}\right)$$

This is the stationarity at the order 1 in the sense of the momentum.

In the case where $E[Y(n_i) * Y^*(n_j)]$ can be written as a quantity that depends only on the time difference $(n_i - n_j)$, and $E[Y(n_1) * Y^*(n_{1+i-j})] = \cdots = E[Y(n_N) * Y^*(n_{N+i-j})] \ \forall i, j$, then one usually write

$$E[Y(n_i) * Y^*(n_j)] = R_Y(n_i - n_j)$$

This, assuming first the stationarity at the order 1, is the stationarity at the order 2 in the sense of the momentum.

In this thesis, the meaning of stationarity is restricted to the *stationarity at the order 2 in the sense of the momentum.*

Ergodicity

From a stationary process, one can show the ergodicity:

$E[Y(n_1)] = \cdots = E[Y(n_i)] \ \forall i = 1 \cdots N.$

Let $Z_N = \frac{1}{N} * \sum_{i=1}^{N} Y(n_i)$ be a new stochastic process,

and $z_N = \frac{1}{N} * \sum_{i=1}^{N} y(n_i)$ a realization of Z_N.

$$\text{Ergodicity of } Z_N \Longleftrightarrow Z_N \to E[Y(n_i)] \text{ when } N \to \infty$$

In other words, it means that all the realizations z_N of Z_N converge torwards the same limit when $N \to \infty$, this limit is the expectation of any random variable $Y(n_i)$ of the process $Y(n)$.

NB: The opposite is not that Z_N diverge when $N \to \infty$, but that each realization of Z_N converges towards a different limit.

Bibliography

[Andre-Obrecht] Andre-Obrecht: "Segmentation et parole" document d'habilitation IRISA 1993 (french)

[Atal] Atal: "High-quality speech at low bit rates: multi-pulse and stochastically excited linear predictive coders" IEEE 1986 (english)

[Bass./Nik.] Basseville & Nikiforov: "Detection of abrupt changes: theory and application" (english)

[Chabert] Chabert: "Detection et estimation de ruptures noyees dans un bruit multiplicatif" PhD thesis INP Toulouse 1997 (french)

[Grenier1] Grenier: "Time-frequency analysis using time-dependent ARMA models" IEEE 1984 (english)

[Grenier2] Grenier: "Time-dependent ARMA modeling of non-stationary signals" IEEE 1983 (english)

[Henderson] Henderson: "Geometric method for determining system poles from transient response" IEEE 1981 (english)

[Kay] Kay: "Modern spectral estimation" Prentice Hall (english)

[Klapuri] Klapuri: "Sound onset detection by applying psychoacoustic knowledge" (english)

[Kum./Tufts] Kumaresan & Tufts: "estimating the parameters of exponentially damped sinusoids and pole-zero modeling in noise" IEEE 1982 (english)

[Laroche1] Laroche: "Etude d'un systeme d'analyse et de synthese utilisant la methode de Prony - Application aux instruments de type percussif" Master's thesis ENST 1988 (french)

[Laroche2] Laroche: "A new analysis/synthesis system of musical signals using Prony's method" IEEE 1989 (english)

[Lep./Obrecht] Lepain & Andre-Obrecht: "Micro-segmentation d'enregistrements musicaux" (french)

[Levine] Levine :"Audio Representations for Data Compression and Compressed Domain Processing" PhD thesis Stanford University 1998 (english)

[Molin./Castanie] Molinaro & Castanie: "A time-varying Prony model" 1992 (english)

[Quat./McAul.] Quatieri & McAulay: "Speech transformations based on a sinusoidal representation" IEEE 1986 (english)

[Rossignol] Rossignol, Rodet, Soumagne, Collette & Depalle: "Features extraction and temporal segmentation of acoustic signals" ICMC 1998 (english)

[Schloss] Schloss: "On the automatic transcription of percussive music" CCRMA 1985 (english)

[Serra/Bon.] Serra & Bonada: "Sound transformations based on the SMS high-level attributes" DAFX 1998 Proceedings (english)

[Shahwan] Shahwan: "An adaptive procedure for the optimization of an acoustic onset detector" technical report San Jose University 1994 (english)

[Verma1] Verma & Meng: "Time scale modification using a sines + transient+noise signal model" (english)

[Verma2] Verma, Levine & Meng: "Transient modeling synthesis: a flexible analysis/synthesis tool for transient signals" ICMC 1997 (english)

[Yvetot] Yvetot: "Analyse de Prony multi-modele de signaux transitoires" PhD thesis INP Toulouse 1996 (french)