Internship report

Nicolas Vinuesa

To cite this version:

Nicolas Vinuesa. Internship report. Signal and Image Processing. 2013. hal-00830202

HAL Id: hal-00830202
https://inria.hal.science/hal-00830202
Submitted on 25 Aug 2013

Internship report
Inria Bordeaux - Team GEOSTAT

NICOLAS VINUESA

10/2012 - 04/2013

Contents

1 Inria

1.1 Inria in brief

The information in this section is based on the official site of INRIA: http://www.inria.fr.

Established in 1967 at Rocquencourt, near Paris, INRIA (Institut National de Recherche Computer Science and Control) is a public establishment of Science and Technology (EPST) under the dual supervision of the Ministry of Research and the Ministry of Economy, Finance and Industry.

The main tasks of INRIA are described by the decree of 2 August 1985 on the organization and functioning of the institute. INRIA's ambition is to be globally, a research institute in the heart of the information society. INRIA National Institute for Research in Computer Science and Control, under the joint supervision of the Ministries of Research and Industry, aims to undertake basic and applied research in the areas of science and information technology and communication (ICT).

The institute also provides a strong technology transfer giving attention to the research training, dissemination of information scientific and technical development, expertise and participation in international programs.

INRIA is a major player in the development of ICT in France. INRIA has 3800 workers in its eight research centers in Rocquencourt, Rennes, Sophia Antipolis, Grenoble, Nancy, Bordeaux, Lille and Saclay, while 2800 of them are scientists from INRIA and partner organizations (CNRS, universities, colleges) who work in more than 150 project teams joint research. Many INRIA researchers are also teachers and their students (about 1000) are preparing their thesis in project teams of INRIA research. INRIA develops many partnerships with industry and fosters technology transfer and the creation companies in the field of ICT.

1.2 Bordeaux Sud-Ouest center / GEOSTAT Team

The Inria BordeauxSud-Ouest Research Centre was created in 2008 and is located on the Bordeaux and Pau university campuses. The Centre, together with its academic and industrial partners, carries out research activities in computational sciences and technologies (IT/mathematics), boosted by technology transfer and innovation in a particularly stimulating and dynamic region.

My internship took place in GEOSTAT team, an INRIA project located at INRIA Bordeaux Sud-Ouest (INRIA BSO), which deals with: applied mathematical computation and simulation, optimization, learning and statistical methods. The project makes fundamental and applied reasearch on new nonlinear methods for the analysis of complex systems, turbulent and natural signals, using paradigms and tools coming from the notions of scale invariance, predictability, universality classes and nonlinear physics.

2 Introduction

For the last decades Matching Pursuit has been widely used in audio (and particularly speech) coding. The efforts for finding a better representation of audio signals led researchers towards different dictionaries used in the atomic decomposition and the implementation of different psycho-acoustical models to code the signal accordingly to the perceived audio.

During my internship in GEOSTAT team under the supervision of Khalid Daoudi we studied the influence of different dictionaries and psycho-acoustical models in the coded audio quality. Taking as a starting point Smith & Lewicki paper of 2005 we compared Gammatone dictionary with other widely used dictionaries (such as Gabor and Dumped Sinusoids). Later psycho-acoustic models where implemented and added to the algorithm to verify its influence in perceived quality when using different dictionaries. Finaly, the coding efficiency of MP using Gammatone dictionary is compared to Gabor dictionary.

This report is organized as follows:

3 Background

3.1 Matching Pursuit

Block-based approaches to signal representation are not suited for modeling non-stationary and time-relative acoustic structures because the arbitrary alignment of the processing blocks results in different representations. To overcome this issue, a sparse, shiftable kernel method of signal representation was used in [4]:

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_i^m \phi_m(t - \tau_i^m) + \epsilon(t), \tag{1}$$

where τ_i^m and s_i^m are the temporal position and weight of the ith instance of kernel ϕ_m, respectively. The notation n_m indicates the number of instances of ϕ_m, which need not to be the same across kernels, and M indicates the number of different kernels. Gammatone kernels are used in this model.

In order to find the optimal values of τ_i^m, s_i^m and ϕ_m for a given signal which minimize the error $\epsilon(t)$, the matching pursuit algorithm (MP) can be used. Matching pursuit is a greedy algorithm which approximates a signal by a finite expansion into time-limited functions, or atoms [2]:

$$R_x^m(t) = \langle R_x^m(t), \phi_\theta \rangle \phi_\theta + R_x^{m+1}(t), \tag{2}$$

with $R_x^0(t) = x(t)$ at the first iteration $m = 0$. At each iteration m, a single atom ϕ_m is selected such that:

$$\phi_m = \arg \max_{\phi_\theta \in D} |\langle R_x^m(t), \phi_\theta \rangle| \tag{3}$$

These atoms are iteratively selected from a redundant dictionary.

3.2 Time-frequency atomic decompositions

Basis expansions have been used widely as a signal model. The main drawback is that a given basis is not well-suitedfor modeling a wide variety of signals [1]. Contrary to this approach, selecting the expansion functions that best match the signal from a an overcomplete set of time-frequency atoms will derive in a better signal-dependent model.

A family of time-frequency atoms is defined by translating, modulating and scaling a single window function $w(t) \in L^2(R)$:

$$\phi_\theta(t) = \frac{1}{\sqrt{s}} w\left(\frac{t-u}{s}\right) e^{i\xi t}, \tag{4}$$

where the factor $1/\sqrt{s}$ normalizes to 1 the norm of $\phi_\theta(t)$, u is the translation, ξ is the frequency modulation and the index $\theta = (s, u, \xi)$ is an element of the set $\Theta = R^+ \times R^2$. The energy of each atom is mostly concentrated around time u and frequency ξ. The family defined by $(\phi_\theta(t))_{\theta\in\Theta}$ is extremely redundant and we must select an appropriate countable subset of atoms.

As argued in [4], the shift-invariance property of the kernels in eq. 1 is motivated by the fact that it shares many properties with cochlear models. Thereby, subsets of atoms in our dictionaries should allow atoms to be positioned at any time index, i.e. $0 \leqslant u \leqslant N$, where N is the length of the signal. In the wavelet signal model, time-frequency atoms are scaled depending on their center frequency. This interesting property is shared by Gammatones, as will be explained later, and along with ξ (or central frequency of each atom) sets another constraint on the subset that will be used as dictionary. The last constraint is the window function $w(t)$, which is different for each dictionary.

Here we employ three types of dictionaries: Gammatones, Gabor and Damped Sinusoids. Gammatone filters can be seen as gamma-modulated sinusoids and are defined as:

$$\gamma(t) = Kt^{n-1} e^{-2\pi b ERB(f_c)} e^{i2\pi f_c t}, \tag{5}$$

where n is the filter order, b is a parameter that controls the bandwith of the filter proportional to the ERB of a human auditory filter (ERB stands for equivalent rectangular bandwith, $ERB(f_c) = 24.7 + 0.108 f_c$) and f_c is the central frequency. Commonly assumed values $n = 4$ and $b = 1.019$ where derived for human auditory filterbank.

As mentioned earlier, we use atoms that are scaled depending on their center frequency. Moreover, we use Gabor and Damped Sinusoid atoms that have the same center frequencies as Gammatones (with linear steps in ERB-rate scale) and therefore the same scale s. For Gabor atoms the window function $w(t)$ is a Gaussian window, and they are defined as:

$$g(t) = Ke^{-\pi\left(\frac{t-u}{s}\right)^2} e^{i2\pi f_c t}, \tag{6}$$

And Damped Sinusoids:

$$d(t) = K\lambda^{(t-u)} e^{i2\pi f_c t}, \tag{7}$$

The main motivation for choosing these three dictionaries is that most natural sounds are asymmetric in time, and Gabor atoms exhibit pre-echo artifacts when used for decompositions of this type of signals.

3.3 Psychoacoustically-based matching pursuits

Eq. 3 does not take into account important psychoacoustical effects produced in the auditory system. To overcome this issue, a perceptually relevant norm shown in [5]:

$$\|x\|^2 = \sum_f \hat{\alpha}(f) |\hat{x}(f)|^2 \tag{8}$$

this norm is induced by the inner product:

$$\langle x, y \rangle = \sum_f \hat{\alpha}(f) \hat{x}(f) \hat{y^*}(f), \tag{9}$$

where $\hat{\alpha}(f)$ is a real and positive weighting function designed to represent the distortion produced by previously selected atoms. The inner product in 3 can be replaced with the one defined in 9, selecting the perceptually best matching atoms. We will refer to his method as pshychoacoustical matching pursuit (PAMP).

The method described above uses a perceptual model based on masking patterns created by each previously selected atom. When the residual equals the signal, i.e. in the first iteration, $\hat{\alpha}(f)$ is the inverse of the absolute threshold of hearing, and later the simultaneous masking effect generated by the selected atom is taken into account. An improved perceptual model is introduced in [3] which not only adds a temporal masking pattern but also defines a stop criterion for the algorithm once there is no audible part left in the residual. We will refer to this method as perceptual matching pursuit (PMP).

While in PAMP the L^2-norm is replaced by a perceptually relevant norm and therefore adjusting the atoms selection, in PMP the matching pursuit algorithm is used without any modification. A time-frequency masking pattern is created progressively to determine a masking threshold at all time indexes and frequencies. The initial level for the masking pattern is set to the absolute threshold of hearing, and add the masking effects caused by the extracted kernel after each iteration. Before finding the maximum value and position of the cross correlation of the residual signal and each kernel, all its values are set to zero if they are below the masking threshold.

4 Implementations

For the first stage of the internship, Matching Pursuit with a set of dictionaries was implemented on Matlab and later validated with the Matching Pursuit Toolkit (MPTK), a well-known implementation of MP in C++. Several types of dictionaries where also developed for MPTK (using XML definition files) for which the MPTK documentation is very useful.

The implementation of MP in Matlab was needed for two reasons. First because running Matlab scripts allows variable dumping at debugging stage, which is very useful for correctly understanding how the algorithm works. Second and most important, the Matlab code for MP is modular which allows us to easily add perceptually relevant selection criteria.

All the documentation for the Matlab implementation as well as basic usage of MPTK can be found in the annex.

5 An analysis of psychoacoustically-inspired matching pursuit decompositions of speech signals

An analysis of psychoacoustically-inspired matching pursuit decompositions of speech signals

Nicolas Vinuesa, Khalid Daoudi

INRIA Bordeaux Sud-Ouest (GEOSTAT team)
200, avenue de la Vieille Tour, 33405 Talence, France
nicolas.vinuesa@inria.fr, khalid.daoudi@inria.fr, http://geostat.bordeaux.inria.fr/

Abstract

Matching pursuit (MP), particularly using the Gammatones dictionary, has become a popular tool in sparse representations of speech/audio signals. The classical MP algorithm does not however take into account psychoacoustical aspects of the auditory system. Recently two algorithms, called PAMP and PMP have been introduced in order to select only perceptually relevant atoms during MP decomposition. In this paper we compare this two algorithms on few speech sentences. The results suggest that PMP, which also has the strong advantage of including an implicit stop criterion, always outperforms PAMP as well as classical MP. We then raise the question of whether the Gammatones dictionary is the best choice when using PMP. We thus compare it to the popular Gabor and damped-Sinusoids dictionaries. The results suggest that Gammatones always outperform damped-Sinusoids, and that Gabor yield better reconstruction quality but with higher atoms rate.

Index Terms: Matching pursuit, Time-frequency decomposition, Sparse representation, Gammatones, Perceptual models.

1. Introduction

During the last two decades, the Matching pursuit (MP) algorithm [1] has been widely used as a powerful tool for spare representation of signals using redundant dictionaries of time-frequency functions (atoms). MP is a greedy algorithm which iteratively approximates a signal $x(t)$ by a projecting it onto an overcomplete dictionary D of atoms ϕ_θ:

$$R_x^m(t) = \langle R_x^m(t), \phi_\theta \rangle \phi_\theta + R_x^{m+1}(t), \tag{1}$$

with $R_x^0(t) = x(t)$ at the first iteration $m = 0$. At each iteration m, a single atom ϕ_m is selected such that:

$$\phi_m = \underset{\phi_\theta \in D}{\arg\max} |\langle R_x^m(t), \phi_\theta \rangle| \tag{2}$$

where $\langle \cdot, \cdot \rangle$ is (generally) the Hermitian inner product. The signal $x(t)$ can be thus decomposed as:

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_i^m \phi_m(t - \tau_i^m) + \epsilon(t), \tag{3}$$

where τ_i^m and s_i^m are the temporal position and weight of the ith instance of the atom ϕ_m, respectively. The notation n_m indicates the number of instances of ϕ_m, which need not to be the same across atoms, and M indicates the number of different atoms.

In the field of speech/audio coding, it has been argued in [2, 3] that a relatively small dictionary of Gammatone atoms allow efficient coding of natural sounds using MP. The motivation behind this work is that early psychoacoustic experiments used Gammatone functions as a model of basilar membrane displacement [4] and where found to approximate cochlear responses of the cat [5]. Later it was stated in [6] that Gammatone functions also delineate the impulse response of human auditory filters. Real-valued Gammatone filters can be seen as gamma-modulated sinusoids and are defined as:

$$\gamma(t) = t^{n-1} e^{-2\pi b ERB(f_c)} cos(2\pi f_c t), \tag{4}$$

where f_c is the central frequency distributed on ERB (equivalent rectangular bandwidth) scales, n is the filter order, and b is a parameter that controls the bandwidth of the filter. $ERB(f_c) = 24.7 + 0.108 f_c$, $n = 4$ and $b = 1.019$ are commonly used as parameters.

On the other hand, classical MP (used in [2, 3]) focus on minimizing the energy of residuals at each iteration. Thus, it does not take into account psychoacoustical aspects of the auditory system which are crucial in any codec development. To address this issue, a psychoacoustically-adaptive inner product, considering frequency masking effects in sinusoidal decompositions, was presented in [7] and later refined with a perceptual model in [8]. The resulting algorithm is called PAMP. More recently a new perceptual model, called PMP, was introduced in [9], taking into account both temporal and frequency masking effects. Similar to the perceptual model embedded in MPEG coders, the goal of psychoacoustically-based MP algorithms is to discard the perceptually irrelevant structures of the input signal and therefore increase the coding efficiency.

In this article we first experimentally compare these two algorithms when using a dictionary of Gammatone atoms. Our experiments suggest that PMP, which also has the strong advantage of including an implicit stop criterion, always outperforms PAMP as well as classical MP. We then raise the question of whether the Gammatones dictionary is the best choice when using PMP. We thus compare it to the popular Gabor and damped-sinusoids dictionaries. The results suggest that Gammatones always outperform damped-sinusoids, and that Gabor yield better reconstruction quality but with higher atoms rate.

The paper is organized as the following. In section 2, the two psychoacoustically-based MP algorithms are briefly described. Section 3 presents and analyzes the results of comparison between classical MP, PAMP and PMP. In section 4 we present an evaluation of PMP when using different dictionaries. Finally, we draw our conclusion and perspectives in section 5.

2. Psychoacoustically-inspired matching pursuits

In this section we give a short description of PAMP and PMP. Details about these algorithms can be found in [7, 8] and [10], respectively.

2.1. PAMP

PAMP [7, 8] relies on a perceptual model which predicts masked thresholds for sinusoidal distortions in audio coding. This model exploits the simultaneous masking effect (frequency masking) in order to determine what distortion level can be allowed such that it is perceptually not detectable. The model is based on a perceptual distortion measure [11] which estimates the probability that subjects can detect a distortion signal in the presence of a masking signal. This distortion measure defines a norm:

$$\|x\|^2 = \sum_f \hat{\alpha}(f)|\hat{x}(f)|^2 \tag{5}$$

where $\hat{x}(f)$ is the Fourier transform of the signal x and $\hat{\alpha}(f)$ is a real and positive weighting function representing the inverse of the masking curve for sinusoidal distortions. The norm is induced by the inner product:

$$\langle x, y \rangle = \sum_f \hat{\alpha}(f)\hat{x}(f)\hat{y}^*(f), \tag{6}$$

This inner product is then used in Eq. 2 instead of the classical Hermitian inner product in order to select the sinusoidal components of the signal in a MP decomposition with a dictionary of sinusoids. At the first iteration, i.e., when the residual equals the signal, $\hat{\alpha}(f)$ is set as the inverse of the threshold-in-quiet. Then, at each iteration $\hat{\alpha}(f)$ takes into account the sinusoidal distortion caused by the atom selected at the previous iteration.

2.2. PMP

PMP [10] relies on a perceptual model which take into account both temporal and frequency masking effects (as opposed to PAMP which considers frequency masking only). In PMP, a dictionary of Gammatone atoms is used and a masking pattern is created (and progressively updated) to determine a masking threshold at all time indexes and atom central frequencies.

At the first iteration, the masking pattern is set to the threshold-in-quiet, as in PAMP, for all time indexes. Then, all the inner products in Eq. 2 which are below the masking pattern are set to zero, meaning that projections that are below he threshold-in-quiet are ignored. Once the first atom has been selected, which will act as a masker, the masking pattern is elevated in a time interval around the atom temporal position and in the two adjacent critical bands. The updated masking pattern is then again used as a threshold, setting to zero all the inner products below it, thereby avoiding the search of atoms that would be masked by previously selected ones, i.e. perceptually irrelevant. This process is repeated until no inner product is above the masking threshold, meaning that there is no audible part left in the residual, and the algorithm stops. This implicit and perceptually-motivated stop criterion is a strong advantage over classical MP and PMP.

3. Comparison between MP, PMP and PAMP

In this section, we experimentally compare the performance of the two psychoacoustically-based and the the classical matching pursuit algorithms. Since Gammatones have become popular waveforms in sparse speech/audio representations, we perform this comparison in the setting of MP using Gammatones dictionaries. The main idea is to analyse the behavior of MP when (Gammatone) atom selection is performed by the two perceptual models. We recall however that, while PMP has been introduced in this setting, PAMP have been developed in the framework of sinusoidal decompositions. By doing such a comparison, we are thus evaluating the behavior of PAMP when distortions are generated by Gammatone components.

We use four sentences from the TIMIT database [12] for the experiments. We selected these excerpts such that both speaker and phonetic variability is achieved: two male (sx54 and sx221) and two female (sx23 and sx136) speakers from different geographic regions are used in this study. The following speakers were used: mbma1 (sx54), fdrw0 (sx23), fgcs0 (sx136) and mcre0 (sx221). All files are sampled at 16 kHz with 16-bit quantization.

While signal to noise ratio (SNR) is a valid measure for *waveform* reconstructability, for audio coding problems this does not necessary reflect the perceived quality of the reconstruction. Therefore we use the well-known perceptual quality assessment measure PESQ [13] to estimate mean opinion scores (MOS). PESQ gives a continuous grading scale from 1 (very annoying) to 5 (no perceptual difference between original and reconstruction).

Since PMP is the only algorithm which implicitly has a perceptual stop criterion, we use the latter as an operating point to compare the 3 algorithms. We first run PMP and then compute the atoms rate per sample when it stops. Then, MP and PAMP are stopped when they reach this atoms rate. The experimental results of this process are are shown in Table 1, for Gammatones dictionaries with 32, 64 and 128 atoms. A first observation is that PMP always yield a PESQ value above 3.5. Moreover, our listening tests confirm that the reconstruction quality is good without being perceptually transparent. This shows that the perceptual model of PMP achieves the desired goal. A second observation is that PMP always slightly outperforms MP and significantly outperforms PAMP. Finally, these results suggest that a good choice for the number of Gammatones in the dictionary is 64.

Figure 1 displays the evolution of the 3 algorithms, iteration after iteration, until they they reach the atoms rate given by PMP. The figure corresponds to only one sentence, but the behavior is very similar for the 4 sentences. Because the masking pattern is updated with the masking effect caused by each new atom, PMP behaves exactly like MP until most of the masker atoms have been extracted; only then (around atoms rate of 0.05) the newly selected atoms are perceptually relevant and the difference can be appreciated. From this rate, PMP starts selecting atoms which are perceptually relevant and thus yields higher PESQ values, while MP selects atoms which minimize the residual energy and thus yields higher SNR values. The weak performances of PAMP are most probably due to the fact that distortions generated by Gammatone decompositions do not satisfy the hypothesis made on distortions obtained in sinusoidal modeling.

In Table 2, we provide the atoms rate required by MP and PAMP to achieve the same PESQ value as PMP (at stopping atoms rate), for 64 Gammatones. It is clear that PMP exhibits the highest efficiency among the three algorithms, as MP requires up to 40% and PAMP up to 80% more atoms to achieve the same perceived quality. All these experimental results suggest that PMP is a very good algorithm for efficient

File	Dictionary	PESQ-PMP	PESQ-MP	PESQ-PAMP	Atoms Rate
sx54					
	32 Gammatones	3.66	3.56	3.17	0.09
	64 Gammatones	3.70	3.61	3.22	0.08
	128 Gammatones	3.70	3.61	3.21	0.08
sx23					
	32 Gammatones	3.77	3.45	3.07	0.15
	64 Gammatones	3.84	3.62	3.08	0.14
	128 Gammatones	3.85	3.67	3.15	0.14
sx136					
	32 Gammatones	3.60	3.43	2.98	0.09
	64 Gammatones	3.64	3.42	3.05	0.08
	128 Gammatones	3.62	3.47	3.08	0.08
sx221					
	32 Gammatones	3.55	3.41	3.19	0.09
	64 Gammatones	3.58	3.47	3.33	0.08
	128 Gammatones	3.59	3.49	3.35	0.08

Table 1: *PESQ and atoms rate using Gammatone dictionary.*

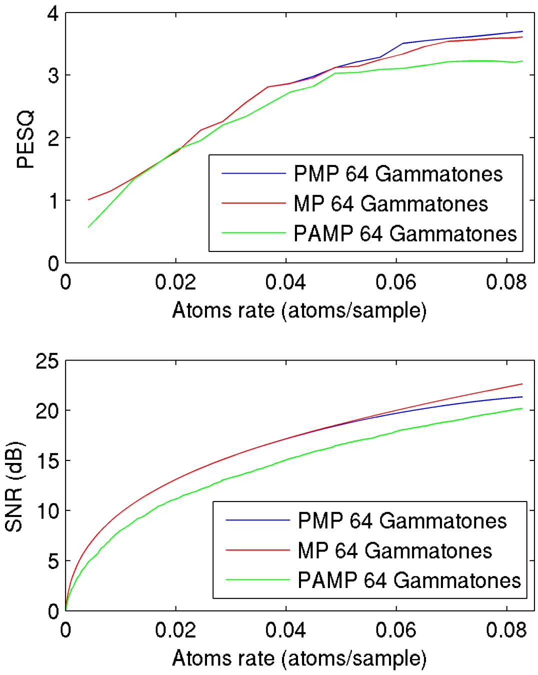

Figure 1: *Comparison of PESQ and SNR vs. atoms rate for sentence sx54 and 64 Gammatones.*

and perceptually-consistent sparse speech representations and coding.

4. Comparison of different dictionaries using PMP

Given the results of the previous section which are in favor of PMP, we now focus on the latter and raise the following question: is the Gammatones dictionary the best choice when using PMP? This question has been indeed central in classical matching pursuit. The original MP algorithm [1] used the Gabor dictionary defined as:

File	Atoms rate PMP	Atoms rate MP	Atoms rate PAMP
sx54	0.08	0.11	0.13
sx23	0.14	0.17	0.23
sx136	0.08	0.10	0.15
sx221	0.08	0.10	0.11

Table 2: *Atoms rate required by MP and PAMP to reach the same PESQ value as PMP.*

$$g_\theta(t) = K_\theta e^{-\pi\left(\frac{t-\tau}{s}\right)^2} e^{i\omega(t-\tau)}, \tag{7}$$

for index $\theta = \{s, \tau, \omega\}$ where s is the scale, τ the time translation , ω the frequency modulation, and K_θ such that $\|g_\theta\| = 1$.

Probably the most known work on this matter is [14], where the authors argued that a dictionary which consists only of atoms that exhibit symmetric time-domain behavior are not well suited for modeling asymmetric events such as transients in audio signals. They proposed the use of structured overcomplete dictionaries of damped sinusoids (DS) defined as:

$$d_\theta(t) = K_\theta \lambda^{(t-\tau)} e^{i\omega(t-\tau)} u(t-\tau), \tag{8}$$

for index $\theta = \{\lambda, \tau, \omega\}$ where λ is the damping factor, τ the time translation, ω the frequency modulation, K_θ such that $\|d_\theta\| = 1$ and $u(t-\tau)$ being the step function.

They showed, in the context of classical MP, that DS are more suited for modeling signals with transient behavior than symmetric Gabor atoms. More recently, the work in [15] proposed a comparison of Gabor atoms, complex exponentials and ”Fonction d’onde Formantique”. The authors argued that the Gabor dictionary performs sufficiently well.

This motivates us to analyse the behavior of PMP when using different dictionaries than Gammatones, within the ERB scale. We thus propose in this section a comparison between Gammatones, damped sinusoids and Gabor atoms.

Table 3 shows the results obtained using 64 atoms per dictionary, within the ERB scale. A first observation is that Gammatones and DS stop at almost the same atoms rate, but Gammatones dictionary outperforms DS. The most important observation is that the Gabor dictionary achieves higher PESQ values

File	Dictionary	PESQ-PMP	Atoms rate
sx54			
	64 Gammatones	3.69	0.08
	64 Gabor	3.85	0.12
	64 D-Sinusoids	3.55	0.08
sx23			
	64 Gammatones	3.84	0.14
	64 Gabor	4.04	0.20
	64 D-Sinusoids	3.54	0.15
sx136			
	64 Gammatones	3.64	0.08
	64 Gabor	3.87	0.12
	64 D-Sinusoids	3.39	0.08
sx221			
	64 Gammatones	3.58	0.08
	64 Gabor	3.85	0.12
	64 D-Sinusoids	3.34	0.09

Table 3: *PESQ and atoms rate using different dictionaries.*

than the other dictionaries, but the atoms rate is also considerably higher. If we rely on atom rates as a measure of coding efficiency, we may consider that the Gammatones dictionary is the best choice, given that PMP requires about 50% more Gabor atoms to achieve a relatively smaller gain in PESQ. However, the best way to assess coding efficiency is to use bits per second, and the best way to asses perceptual quality is MOS. Moreover, the ERB scale may not be the optimal choice for Gabor and DS atoms. Finally, we used only 4 sentences in our experiments. A much larger and richer database should be used in order to have a good evaluation. Thus, all these factors (at least) should be taken into account before drawing final conclusions. The question we raised is then still open, but we may still argue that PMP with Gammatones presents a promising potential.

5. Conclusion

In this paper, we first presented an experimental comparison of two psychoacoustically-based matching pursuit algorithms (PMP and PAMP) as well as the classical MP algorithm. The results suggest that PMP always outperforms both MP as PAMP in term of sparsity and perceived reconstruction quality. In a second experiment, we compared different dictionaries (Gammatones, Gabor and damped-sinusoids) using PMP. The results suggest that Gammatones is the best choice if atoms rate is considered as a measure for coding efficiency. All these results suggest that PMP is a very good algorithm for efficient and perceptually-consistent sparse speech representations and coding. However, further work is required in refining the perceptual model in PAMP in order to take into account the distortions generated by Gammatone decompositions more accurately. A more in-depth study of the coding efficiency, using bits per second instead of atoms rate, is also necessary. Finally, a large and rich speech database should be used. This will be the purpose of future communications.

6. Acknowledgments

The authors would like to thank Ramin Pichevar and Hossein Najaf-Zadeh for the help provided in the implementation and fruitful discussions over PMP.

7. References

[1] S. G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3397–3415, 1993.

[2] E. Smith and M. S. Lewicki, “Efficient coding of time-relative structure using spikes,” *Neural Computation*, vol. 17, no. 1, pp. 19–45, 2005.

[3] E. C. Smith and M. S. Lewicki, “Efficient auditory coding,” *Nature*, vol. 439, no. 7079, pp. 978–982, 2006.

[4] J. L. Flanagan, “Models for approximating basilar membrane displacement,” *The Journal of the Acoustical Society of America*, vol. 32, no. 7, pp. 937–937, 1960.

[5] P. I. Johannesma, “The pre-response stimulus ensemble of neurons in the cochlear nucleus,” in *Proceedings of the Symposium of Hearing Theory*, 1972.

[6] R. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, “An efficient auditory filterbank based on the gammatone function,” *APU report*, vol. 2341, 1988.

[7] R. Heusdens, R. Vafin, and W. B. Kleijn, “Sinusoidal modeling using psychoacoustic-adaptive matching pursuits,” *Signal Processing Letters, IEEE*, vol. 9, no. 8, pp. 262–265, 2002.

[8] S. van de Par, A. Kohlrausch, R. Heusdens, J. Jensen, and S. H. Jensen, “A perceptual model for sinusoidal audio coding based on spectral integration,” *EURASIP Journal on Applied Signal Processing*, vol. 2005, pp. 1292–1304, 2005.

[9] H. Najaf-Zadeh, R. Pichevar, L. Thibault, and H. Lahdili, “Perceptual matching pursuit for audio coding,” *Audio Engineering Society Convetion, Amsterdam, The Netherlands*, 2008.

[10] R. Pichevar, H. Najaf-Zadeh, L. Thibault, and H. Lahdili, “Auditory-inspired sparse representation of audio signals,” *Speech Communication*, vol. 53, no. 5, pp. 643–657, 2011.

[11] S. Van De Par, A. Kohlrausch, G. Charestan, and R. Heusdens, “A new psychoacoustical masking model for audio coding applications,” in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 2. IEEE, 2002, pp. II–1805.

[12] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, and V. Zue, “DARPA TIMIT acoustic-phonetic continuous speech corpus,” U.S. Dept. of Commerce, NIST, Gaithersburg, MD, Tech. Rep., 1993.

[13] “Perceptual evaluation of speech quality (pesq), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs,” ITU-T Rec. P.862, 2001.

[14] M. M. Goodwin and M. Vetterli, “Matching pursuit and atomic signal models based on recursive filter banks,” *Signal Processing, IEEE Transactions on*, vol. 47, no. 7, pp. 1890–1902, 1999.

[15] B. L. Sturm and J. D. Gibson, “Matching pursuit decompositions of non-noisy speech signals using several dictionaries,” in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 3. IEEE, 2006, pp. III–III.

6 Atom coding

Up to now, different algorithms as well as different dictionaries was compared in terms of number of atoms needed in the reconstructed audio to achieve the same quality.

After obtaining the results shown in the paper above, a final set of experiments was carried out in order to compare the Gammatone dictionary with overcomplete dictionaries such as Gabor, Damped sinusoids and “Full” Gammatones using the TIMIT test database now taking into account the “cost” of the code (i.e. the number of bits needed to encode each atom).

In Smith & Lewicki (2005), an objective comparison between classical coding schemes and MP using Gammatones has been made. Our goal here was to extend this objective comparison over the different dictionaries used along the internship. A priori, we expect that the number of bits needed to encode Gammatone atoms should be smaller than all the other dictionaries, because they show a “full” structure and therefore more parameters need to be coded. Rate-fidelity curves will show us if Gammatones are “cheap” enough to achieve a better bitrate vs SNR than “full”-structured dictionaries.

In order to perform this evaluations, the entropy of the code for each type of dictionary had to be estimated and therefore obtain the code bitrate. This is done in two steps. The first step is quantization, and since the only non-integer parameter is the amplitude (for all types of dictionaries) this is the only parameter that needs to be quantized. Secondly the code entropy is computed from the histograms of each parameter (included the quantized amplitude).

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2(x_i) \tag{10}$$

Where $H(X)$ is the entropy of the random variable X, which represents the histogram of each parameter, and it has to be computed separately for each parameter. To obtain the histogram of each parameter, MP decompositions using all the dictionaries were run over the TIMIT database.

It has to be remembered that in Gammatone dictionary the bandwith of each gammatone atom is proportional its central frequency while in full-structured dictionaries (e.g. Gabor) those parameters are not correlated. This property of Gammatone dictionaries allows each atom to be coded with less bits than Gabor dictionaries (for example).

Dictionary	Entropy (bits)
Gabor	26.3
Gammatones	21.3
Dumped Sinusoids	26.8
Full-gammatones	26.6

Table 1: *Entropy of the code of each dictionary.*

The table above shows the final entropy of each type of dictionary, it means, the sum of the entropy of each parameter. The number of bits needed to code Gammatones confirms what was expected, as its lower amount of parameters is reflected in its entropy. In order to obtain the rate-fidelity curves, the total entropy of each dictionary has to be multiplied by the number of atoms in the reconstruction, i.e. multiply the atom-rate (number of atoms per second) by the entropy:

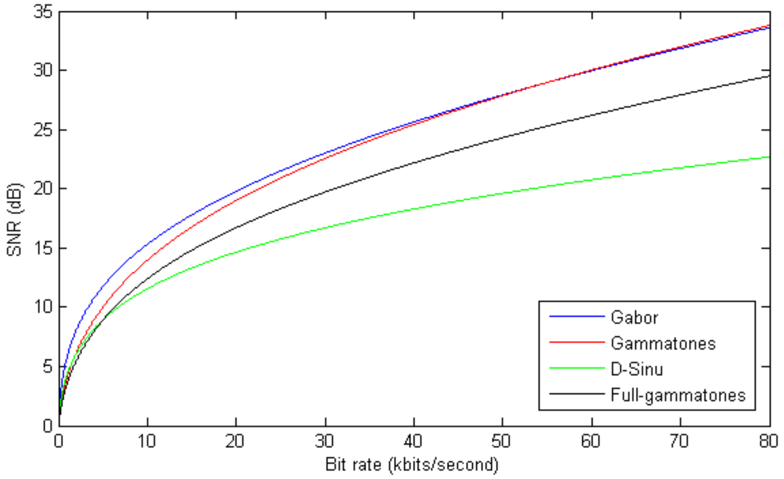

Figure 1: bitrate vs. SNR of different dictionaries.

As can be seen in the figure above, Gabor dictionaries outperforms Gammatones (at least until a bitrate of 60 kbits/second), while Damped sinusoids and Full-gammatones show the worst results.

From this analysis we can conclude that even though Gabor dictionaries are “more expensive” to code than Gammatones, this is compensated by the fact that less Gabor atoms are needed in order to achive the same audio quality. The location of the scripts used for this analysis can be found on the annex.

References

[1] Michael M Goodwin and Martin Vetterli. Matching pursuit and atomic signal models based on recursive filter banks. *Signal Processing, IEEE Transactions on*, 47(7):1890–1902, 1999.

[2] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415, 1993.

[3] Hossein Najaf-Zadeh, Ramin Pichevar, L Thibault, and H Lahdili. Perceptual matching pursuit for audio coding. *Audio Engineering Society Convetion, Amsterdam, The Netherlands*, 2008.

[4] Evan Smith and Michael S Lewicki. Efficient coding of time-relative structure using spikes. *Neural Computation*, 17(1):19–45, 2005.

[5] Steven van de Par, Armin Kohlrausch, Richard Heusdens, Jesper Jensen, and Søren Holdt Jensen. A perceptual model for sinusoidal audio coding based on spectral integration. *EURASIP Journal on Applied Signal Processing*, 2005:1292–1304, 2005.

7 Annex

In this folder all the necessary scripts and files for running Matching Pursuit can be found, both Matlab implementation as MPTK.
MPTK should be installed before running any test, following the User Manual (http://mptk.irisa.fr/) as well as configuring the path for using it with Matlab (all the files for Matlab usage are located in /usr/local/mptk/matlab).
The general usage for mpd (matching pursuit decomposition) function is:

```
mpd -C /usr/local/mptk/path.xml -D dic.xml -E decay.txt -n 1000 foo.wav foo.bin
```

(in this case the algorithm will run 1000 iterations).

```
mpd -C /usr/local/mptk/path.xml -D dic.xml -E decay.txt -s 25 foo.wav foo.bin
```

(in this case the algorithm will run until a SNR of 25dB is reached).

For plotting SNR vs. iterations under Matlab (using the energy decay: decay.txt) this snippet can be used:

```
fid = fopen('decay.txt','r');
booken = fread(fid,inf,'double');
fclose(fid);
SNR=10*log10(booken(1)./booken);
atomsPerSecond=(1:length(SNR))/signalLengthInSeconds; % divide number of atoms
by length of signal (seconds)
plot(atomsPerSecond,SNR);
```

Also, dictionaries created through the Matlab implementation of MP can be used in MPTK via “anywave atoms”. Anywave atoms need the waveform definition of each atom, passed through a binary table which contains all the single atoms in the dictionary. This snippet can be used to create the .bin file from Matlab (remember to replace gamma_erb with the name of your dictionary):

```
for i=1:size(gamma_erb,1)    % where gamma_erb has to be replaced by the name
of the dictionary that we are writing to MPTK
    if i==1
        export=gamma_erb(i,:);
    else
        export=[export gamma_erb(i,:)];
    end
end
fileID = fopen('anywave_table.bin','w');
fwrite(fileID,export,'double');
fclose(fileID);
```

Examples of anywave atom definitions can be found in the folder MPTK/dic/.

Bookplot, bookover and bookedit functions located in /usr/local/mptk/matlab/ can be used in Matlab to edit and analyse .bin “book” files.

Matching Pursuit reconstructions are performed with the mpr function:

```
mpr -C /usr/local/mptk/path.xml foo.bin foo_reconstructed.wav
```

Tests on the full timit_test database can be performed using ./scriptMPTK script (which calls ./mptk_singleFile and ./MPTK_parallel.pl) and modifying the folders and paths in them.

The entropy of the code for each atom can be found using gabor_entropy.m and anywave_entropy.m located in “Matching Pursuit/Entropy”. The folder Quantizer should be added to the path in Matlab.

Important note: Always remember that when working with Matlab scripts of MPTK (located in /usr/local/mptk/matlab/) the script GettingStarted.m should be ran before running any other script (because it loads the configuration), otherwise errors will be shown.

Different dictionaries definition files for MPTK can be found in the folder dic/.
The structure of the discrete Gabor atoms (also for fullgamma and dsinu) is the one in MPTK description paper (found in mptk.irisa.fr) as well as in Sturm & Gibson, 2006 (ICASSP).
For Gammatone dictionaries the structure used is Anywave atoms, provided by a definition table generated by Matlab.

Important note: Always remember that when working with Matlab scripts of MPTK (located in /usr/local/mptk/matlab/) the script GettingStarted.m should be ran before running any other script (because it loads the configuration), otherwise errors will be shown.

This folder contains all the necessary m files for the MP, PAMP and PMP Matlab implementation. A script is provided in script.m which performs a signal decomposition and a 2-D time-frequency representation of the selected atoms, where the dictionary and stop criterion of the algorithms are specified within the file. All subfolders should be added to the path in Matlab. MP algorithm implementation is based on Patrick Mineault's code:
http://xcorr.net/2011/08/05/matching-pursuit-for-one-dimensional-signals-matlab-code/
http://www.mathworks.com/matlabcentral/fileexchange/32426-matching-pursuit-for-1d-signals
A detailed description of the m files for each dictionary can be found in the directory dict/.

Function: CMP.m
Description: Matching pursuit decomposition for 1-D signals. It will stop either at iter, or when snrLimit is reached, which happens first.
Usage: **[ws,r,SnrOut] = CMP(y,D,iter,snrLimit)**
Inputs:
y : Input signal length N. Nx1 vector
D : Dictionary. MxL matrix, M is number of different atoms (without taking into account positions) and L is atom length.
iter : Stop after “iter” iterations.
snrLimit : Stop at desired level of snr (dB).
Outputs:
ws : MxN matrix showing position of each selected atoms. Used to plot the 2-D time-frequency representation.
r : Reconstructed signal. Nx1 vector.
SnrOut: SNR (dB) after each iteration. Length(SnrOut) is equal to the number of iterations ran.

Function: PAMP.m
Description: Psychoacoustical Matching pursuit decomposition for 1-D signals. It will stop either at iter, or when snrLimit is reached, which happens first. Described in [1].
Usage: **[ws,r,SnrOut] = PAMP(y,D,iter,snrLimit)**
Inputs:
y : Input signal length N. Nx1 vector
D : Dictionary. MxL matrix, M is number of different atoms (without taking into account positions) and L is atom length.
iter : Stop after “iter” iterations.
snrLimit : Stop at desired level of snr (dB).
Outputs:
ws : MxN matrix showing position of each selected atoms. Used to plot the 2-D time-frequency representation.
r : Reconstructed signal. Nx1 vector.
SnrOut: SNR (dB) after each iteration. Length(SnrOut) is equal to the number of iterations ran.

Function: PAMP_fixed.m
Description: Psychoacoustical Matching pursuit decomposition for 1-D signals, fixed norm approximation described in [2]. It will stop either at iter, or when snrLimit is reached, which happens first.
Usage: **[ws,r,SnrOut] = PAMP_fixed(y,D,iter,snrLimit)**
Inputs:
y : Input signal length N. Nx1 vector
D : Dictionary. MxL matrix, M is number of different atoms (without taking into account positions) and L is atom length.
iter : Stop after “iter” iterations.

snrLimit : Stop at desired level of snr (dB).
Outputs:
ws : MxN matrix showing position of each selected atoms. Used to plot the 2-D time-frequency representation.
r : Reconstructed signal. Nx1 vector.
SnrOut: SNR (dB) after each iteration. Length(SnrOut) is equal to the number of iterations ran.

Function: PMP.m
Description: Perceptual Matching pursuit decomposition for 1-D signals. It will stop either at iter, or when snrLimit is reached, which happens first. Described in [3]. For PMP only normalized files should be used (e.g sx54_norm.wav).
Usage: **[ws,r,SnrOut,exitIter] = PMP(y,D,iter,snrLimit)**
Inputs:
y : Input signal length N. Nx1 vector
D : Dictionary. MxL matrix, M is number of different atoms (without taking into account positions) and L is atom length.
iter : Stop after “iter” iterations.
snrLimit : Stop at desired level of snr (dB).
Outputs:
ws : MxN matrix showing position of each selected atoms. Used to plot the 2-D time-frequency representation.
r : Reconstructed signal. Nx1 vector.
ExitIter: Iteration number in which the algorithm stops according to the stop criterion of PMP.
SnrOut: SNR (dB) after each iteration. Length(SnrOut) is equal to the number of iterations ran.

References:

[1] van de Par, Steven, et al. "A perceptual model for sinusoidal audio coding based on spectral integration." EURASIP Journal on Applied Signal Processing, 2005.
[2] Heusdens, Richard. "Rate-distortion optimal sinusoidal modeling of audio and speech using psychoacoustical matching pursuits." MPCA-2002, 2002.
[3] Pichevar, Ramin, et al. "Auditory-inspired sparse representation of audio signals." Speech Communication 53.5 (2011): 643-657.

This folder contains all the Matlab scripts needed to compute the entropy of atoms belonging to anywave dictionaries (in our case this are only gammatones) and Gabor dictionaries (Gabor, Full-gamma, d-sinu).

The script **anywave_entropy.m** loads the book (MPTK decomposition) and computes the entropy for anywave structured dictionaries. **The first line including the name and location of the book should be adjusted.**

The script **gabor_entropy.m** loads the book (MPTK decomposition) and computes the entropy for gabor structured dictionaries. **The first line including the name and location of the book should be adjusted.**